

Einführung in die statistische Datenanalyse mit R

2. Auflage

Matthias Kohl

Inhaltsverzeichnis

Abbildungsverzeichnis	iii
Tabellenverzeichnis	iv
Vorwort	v
Vorwort zur ersten Ausgabe	vi
1 Die Statistiksoftware R	1
1.1 R und seine Entwicklungsgeschichte	1
1.2 Der Aufbau von R	2
1.3 Die Installation von R	3
1.4 Das Arbeiten mit R	4
1.5 Markdown in Verbindung mit R	6
1.6 Übungsaufgaben	7
2 Deskriptive Statistik	8
2.1 Grundlagen	8
2.2 Exkurs: Datenimport und -export mit R	12
2.3 Import des ITS-Datensatzes	14
2.4 Exkurs: Erweiterungspakete installieren	19
2.5 Kategoriale Variablen	20
2.5.1 Univariate Analyse	20
2.5.2 Bivariate Analyse	36
2.6 Metrische Variablen	43
2.6.1 Univariate Analyse	43
2.6.2 Bivariate Analyse	62
2.7 Übungsaufgaben	67
3 Farben und Diagramme	69
3.1 Farben	69
3.2 Exkurs: Export von Diagrammen	78
3.3 Diagramme	81
3.4 Übungsaufgaben	88
4 Wahrscheinlichkeitsverteilungen	90
4.1 Diskrete Verteilungen	91

4.2 Stetige Verteilungen	107
4.3 Übungsaufgaben	124
5 Schätzen	127
5.1 Einführung	129
5.2 Punktschätzer	130
5.3 Konfidenzintervalle	160
5.4 Übungsaufgaben	192
6 Statistische Hypothesentests	199
6.1 Einführung	200
6.2 Nominale Merkmale	210
6.3 Ordinale und quantitative Merkmale	224
6.4 Übungsaufgaben	257
7 Multiples Testen	261
7.1 Einführung	262
7.2 Family-wise Error Rate (FWER)	262
7.3 False Discovery Rate (FDR)	271
7.4 Übungsaufgaben	278
Softwareversionen	281
Literaturverzeichnis	283
Sachverzeichnis	293
R Verzeichnis	300

Abbildungsverzeichnis

1.1	Die R GUI (64-bit) unter Windows.	4
1.2	Die RStudio IDE nach der Installation unter Ubuntu Linux.	5
1.3	Die RStudio IDE nach dem Öffnen eines neuen R Skripts unter Ubuntu Linux.	6
2.1	Zusammenspiel von Wahrscheinlichkeitsrechnung, deskriptiver und induktiver Statistik. .	9
2.2	Merkmalsarten und Skalenniveaus.	11
2.3	RStudio Fenster für den Import von Textdateien.	13
2.4	RStudio Fenster <i>Environment</i> mit einem Datenobjekt.	14
2.5	Anzeige der genaueren Struktur der Daten in RStudio.	17
2.6	Installation von R Paketen in RStudio.	19
2.7	Interaktive kontextbezogene Hilfe in RStudio.	23
2.8	Beispiele für Schiefe.	50
2.9	Beispiele für Kurtosis.	52
3.1	Ein Negativbeispiel für den Einsatz von Farben und Diagrammen.	70
3.2	Ein Negativbeispiel mit verbesserter Farbauswahl.	73
3.3	Von einem Negativ- zu einem Positivbeispiel.	74
3.4	RStudio Fenster <i>Plots</i> mit einem Beispiel.	78
3.5	RStudio Fenster zum Abspeichern eines Plots als Bild.	79
3.6	RStudio Fenster zum Abspeichern eines Plots als pdf-Datei.	79
3.7	Sortiere die Kategorien der Größe nach!	81
3.8	Noch einmal: Sortiere die Kategorien der Größe nach!	82
3.9	Und noch einmal: Sortiere die Kategorien der Größe nach!	83
5.1	Veranschaulichung von biasfrei und effizient.	131
5.2	Verhältnis zwischen den 95%-Quantilen der t- und der Standardnormalverteilung. . . .	166
6.1	Fallzahl in Abhängigkeit von der Effektstärke.	206
6.2	Fallzahl in Abhängigkeit von der Standardabweichung.	206
6.3	Baumdiagramm zur Auswahl des geeignetsten Tests im Fall nominaler Merkmale. . . .	211
6.4	Baumdiagramm zur Auswahl des geeignetsten Tests im Fall ordinaler oder quantitativer Merkmale.	225

Tabellenverzeichnis

2.1	Übersicht über einige grundlegende Funktionen zum Datenimport mit R.	12
3.1	Übersicht über Grafikgeräte (devices), die von R unterstützt werden.	80
4.1	Begriffe aus der Statistik und ihre Gegenstücke in der Wahrscheinlichkeitsrechnung . . .	124
6.1	Entscheidungssituation beim statistischen Testen.	202
6.2	Beispiel einer 2×2 -Kontingenztafel.	214
7.1	Testentscheidungen beim multiplen Testen.	271

Vorwort

Mittlerweile sind mehr als fünf Jahre seit der Erstellung der ersten Version dieses Buches vergangen. Da sich die Statistiksoftware R sehr dynamisch entwickelt, funktioniert manches aus der ersten Version des Buches nicht mehr und es haben sich vielfältige neue Möglichkeiten ergeben. Daher war es höchste Zeit, ein entsprechendes Update des Buches und des darin enthaltenen R Codes zu erstellen. Insbesondere verwende ich nun seit einigen Semestern nicht mehr einfache R Skripte für meine Vorlesungen, sondern entsprechende R Markdown Dokumente (siehe Abschnitt 1.5).

Weitere wichtige Updates im Vergleich zur ersten Version des Buches sind:

- Links zu meinem GitHub Repository ESDR
- deutlich mehr Übungsaufgaben
- kurze Videos auf YouTube
- AL- und RMX-Schätzer
- Schätzen von Schwellenwerten
- Bootstrap-Konfidenzintervalle
- mehr Beispiele zur Fallzahlplanung
- statistische Tests für Meßwiederholungen
- neues Kapitel zu multiplem Testen

Der R Code, der in den einzelnen Kapiteln bzw. Abschnitten verwendet wird, findet sich auf GitHub <https://github.com/stamats/ESDR> und ist in Textdateien (R Markdown Dateien) mit der Dateiendung .Rmd enthalten. Der R Code eines jeden Kapitels kann jeweils losgelöst von den anderen Kapiteln ausgeführt werden. Außerdem finden sich auf YouTube unter <https://youtu.be/p1W93veYjN8> Videos mit kurzen Erklärungen.

Auch die zweite Auflage des Buches wurde wieder mit Hilfe des Softwarepakets L^AT_EX und pdfL^AT_EX erstellt. Zusätzlich kam wieder das Erweiterungspaket "knitr" (Xie (2015a)) für die Statistiksoftware R zum Einsatz, welches flexible Möglichkeiten bietet, Erklärungen mit Ein- und Ausgaben von R zu kombinieren.

Villingen-Schwenningen im Oktober 2022

Matthias Kohl

Vorwort zur ersten Ausgabe

Die Statistik hat heutzutage in alle Bereiche des Lebens Einzug gehalten und wir werden stetig bewusst oder unbewusst mit den Ergebnissen statistischer Verfahren konfrontiert. Dazu zählen die Ergebnisse von Internetsuchmaschinen, gezielte Werbeangebote auf besuchten Websites, die Einschätzung unserer Kreditwürdigkeit bei Abschluss von Kauf- oder Mobilfunkverträgen, die angegebenen Normbereiche bei Laboruntersuchungen unseres Blutes, die Wettervorhersage, Wahlprognosen und vieles mehr. Nicht immer werden statistische Verfahren korrekt eingesetzt bzw. die Ergebnisse korrekt wiedergegeben. Statistische Grundkenntnisse sind daher nicht nur im beruflichen, sondern auch im alltäglichen Leben wichtig, um richtige von falschen Informationen besser unterscheiden zu können.

Die Grundlage dieses Buches bildeten meine Vorlesungsskripten zu einer Reihe von Statistikkursen, die ich in den letzten Jahren an der Hochschule Furtwangen, Campus Villingen-Schwenningen, im Rahmen verschiedener Bachelor- und Masterstudiengänge sowie an der Universität Freiburg im Rahmen des International Master Program in Biomedical Sciences (IMBS) abgehalten habe.

Wie der Titel des Buches bereits andeutet, erfolgt die Einführung in die statistische Datenanalyse unter Verwendung der Statistiksoftware R (R Core Team (2022a)), einer freien Software, die für die meisten gängigen Betriebssysteme zur Verfügung steht.

Der R Code, der in den einzelnen Kapiteln bzw. Abschnitten verwendet wird, ist in der Datei www.stamats.de/RCodeDE.zip in Form von Textdateien mit der Dateiendung `.R` enthalten. Der R Code eines Kapitels kann jeweils losgelöst von den anderen Kapiteln ausgeführt werden.

Anmerkung:

Bei der Erstellung des Buches wurden einige von R generierte Meldungen bewusst unterdrückt, um den Blick nicht vom Wesentlichen abzulenken und um Platz zu sparen. Bei den unterdrückten Ausgaben handelt es sich jeweils um Meldungen, die für die vorgestellten Analysen nicht von Bedeutung sind. Umgekehrt heißt dies jedoch, dass Sie mit zusätzlichen Ausgaben rechnen müssen, wenn Sie den im Buch enthaltenen Code ausführen. Dazu gehören auch als harmlos einzustufende Warnmeldungen.

Das Buch wurde mit Hilfe des Softwarepakets \LaTeX und $\text{pdf}\text{\LaTeX}$ erstellt. Zusätzlich kam das Erweiterungspaket "knitr" (Xie (2015b)) für die Statistiksoftware R zum Einsatz, welches flexible Möglichkeiten bietet, Erklärungen mit Ein- und Ausgaben von R zu kombinieren.

Villingen-Schwenningen im August 2015

Matthias Kohl

1 Die Statistiksoftware R

Dieses Kapitel gibt eine sehr kurze Einführung in die Statistiksoftware R, wobei folgende Aspekte behandelt werden:

- die Entwicklungsgeschichte ausgehend von der Statistik-Programmiersprache S
- der modulare Aufbau in Form von Paketen
- die Installation unter verschiedenen Betriebssystemen
- die Installation der integrierten Entwicklungsumgebung (IDE) RStudio

Das konkrete Arbeiten mit R wird in den folgenden Kapiteln zusammen mit der Einführung in die statistische Datenanalyse vorgestellt.

1.1 R und seine Entwicklungsgeschichte

Die Statistiksoftware R (R Core Team (2022a)) ist eine freie, nicht-kommerzielle Implementierung der bei den AT&T Bell Laboratories von Rick Becker, John Chambers und Mitarbeitern entwickelten Statistik-Programmiersprache S. Es handelt sich um eine Entwicklungsumgebung und Programmiersprache für Statistik und Grafik, welche unter GNU GPL-2/3 entwickelt wird und somit jederzeit ohne Einschränkung auf beliebig vielen Rechnern installiert werden kann.

R ist eine auf Funktionen basierende Sprache. Das bedeutet, dass alle Aktionen durch den Aufruf von Funktionen initiiert werden. Dabei werden den Funktionen häufig zusätzliche Parameter mit übergeben, welche die konkrete Ausführung der Funktion steuern. Die Funktion wird anhand ihres Namens identifiziert, die Parameter anhand ihres Namens oder auch ihrer Position. Ein Aufruf hat demnach folgende Struktur (wenn auch nicht immer sofort ersichtlich):

```
Funktionsname(Parameter1 = Wert1, Parameter2 = Wert2, ..., ParameterN = WertN)
```

Wir werden hierzu eine Vielzahl von Beispielen im Buch kennenlernen.

Im Folgenden fassen wir kurz die Entwicklungsgeschichte von S und R zusammen:

05.05.1976: Beginn der Entwicklung von Version 1 von S (Chambers (2008, p. 476))

1980: Veröffentlichung der Version 2 von S (Chambers (2000))

1988: Veröffentlichung der Version 3 von S (S3) (Chambers (2000))

1992: Start des R Projekts durch Ross Ihaka und Robert Gentleman (Hornik (2008))

August 1993: Erste Dateien von R werden auf Statlib veröffentlicht (Ihaka (1998)).

Juni 1995: Veröffentlichung der ersten GPL Version von R (Ihaka (1998))

05.12.1997: Das R Projekt wird offiziell zu einem GNU Projekt (Ihaka (1997)).

1998: Veröffentlichung der Version 4 von S (S4) (Chambers (2000))

29.02.2000: R 1.0.0 veröffentlicht, eine Implementation von S3 (Hornik (2008))

04.10.2004: R 2.0.0 veröffentlicht, eine Weiterentwicklung von S4 (Chambers (2008), Hornik (2008))

22.04.2010: R 2.11.0 veröffentlicht, unterstützt Windows 64bit-Systeme (Dalgaard (2010))

03.04.2013: R 3.0.0 veröffentlicht, unbegrenzter Speicherplatz bei 64bit-Systemen (Dalgaard (2013))

24.04.2020: R 4.0.0 veröffentlicht, Reduktion des Speicherbedarfs, neue Farbpaletten und vieles mehr (Dalgaard (2020))

Generell erscheint im Frühling (März/April) eines jeden Jahres jeweils eine neue Version R x.y.0, wobei dann je nach Bedarf über das Jahr verteilt Patches (R x.y.1, R x.y.2, etc.) veröffentlicht werden (R Core Team (2022c)). Für die Entwicklung des Grundsystems von R zeichnet sich das sogenannte R Core Development Team verantwortlich, welches aktuell aus 20 Mitgliedern besteht (The R Foundation (2022a)). Im Jahr 2002 wurde außerdem die R Foundation (The R Foundation (2022b)) gegründet, an der die R Core Development Team Mitglieder als ordentliche Mitglieder beteiligt sind. Die Ziele dieser Foundation umfassen das Fortführen der Entwicklung von R, die Erforschung neuer Methoden, die Lehre und das Training im Bereich der computergestützten Statistik sowie die Organisation von Zusammenkünften und Konferenzen mit einer Ausrichtung auf computergestützte Statistik.

Zusätzlich wurde im Juni 2015 unter dem Dach der Linux Foundation ein R Consortium gegründet, um R auch von Seiten der Industrie stärker zu unterstützen. Ihm gehören Firmen an wie Microsoft, Google, Oracle und Merck (R Consortium (2022)).

Die Popularität und den Marktanteil von Software für die Datenanalyse versucht Muenchen (2022) abzuschätzen. Die Statistiksoftware R schlägt sich dabei in allen Statistiken sehr gut und spielt heute eine zentrale und in manchen Bereichen sogar führende Rolle.

1.2 Der Aufbau von R

Die Statistiksoftware R ist in Paketen aufgebaut, welche in einer oder mehreren Bibliotheken organisiert sind. Bei den Paketen wird zwischen drei Kategorien unterschieden. Zunächst einmal gibt es die **Grund-pakete** (base packages). Diese Pakete stellen die grundlegende Funktionalität von R bereit und werden vom R Core Development Team gepflegt. Aktuell sind es die folgenden 14 Pakete: "base", "compiler", "datasets", "grDevices", "graphics", "grid", "methods", "parallel", "splines", "stats",

"stats4", "tcltk", "tools", "utils"; für weitere Informationen siehe Abschnitt 5 der FAQs von R (Hornik (2022)).

Die zweite Gruppe von Paketen, die ebenfalls zur Standardinstallation von R gehören, sind die **empfohlenen Pakete** (recommended packages). Diese Pakete stellen in erster Linie weitere komplexe statistische Verfahren zur Verfügung. Im einzelnen sind es die folgenden 15 Pakete: "boot", "class", "cluster", "codetools", "foreign", "KernSmooth", "lattice", "MASS", "Matrix", "mgcv", "nlme", "nnet", "rpart", "spatial", "survival" (Hornik (2022, Abschnitt 5)).

Schließlich gibt es noch die **Erweiterungspakete**, die sogenannten “contributed packages”. Die Offenheit von R, die es erlaubt, dass jeder jederzeit neue Pakete beisteuern kann, ist sicherlich ein wichtiger Grund für den Erfolg und die weite Verbreitung von R. So gibt es eine kontinuierlich wachsende Entwicklergemeinde, die stetig neue Pakete zu R beiträgt, wobei die Anzahl der Erweiterungspakete seit mehr als zehn Jahren exponentiell wächst. Die aktuelle Anzahl an R Paketen ist nur schwer zu bestimmen und liegt weit jenseits von 25 000 (DataCamp Inc. (2022), Howson Ian (2022)). Diese Pakete verteilen sich auf mehrere sogenannte Repositorien. Auf dem offiziellen Repository CRAN (Comprehensive R Archive Network, <http://cran.r-project.org/>) sind es aktuell mehr als 18 000 Pakete. Erweiterungspakete, die sich mit der Analyse und dem Verständnis von genomischen Daten befassen, sind zum großen Teil auf Bioconductor (Gentleman et al. (2004), <http://www.bioconductor.org/>) beheimatet. Es stehen dort mittlerweile mehr als 2 000 Pakete zum Download bereit. Eine stark wachsende Anzahl von R Paketen finden sich auf GitHub (<https://github.com/>) und anderen Git Repositorien.

1.3 Die Installation von R

Die Dateien, die benötigt werden, um die Statistiksoftware R unter Windows, Mac OS X oder Linux zu installieren, können von CRAN (<https://cran.r-project.org/>) oder einem seiner Mirror heruntergeladen werden. Generell unterscheidet sich die Installation von R nicht von der üblichen Installation anderer Software unter diesen Betriebssystemen.

Windows Der Windows Installer für 32- und 64-bit findet sich unter <https://cran.r-project.org/bin/windows/base/>. Weitere Informationen zu Installation, Update oder auch Deinstallation unter Windows finden sich in den FAQs für Windows (Ripley and Murdoch (2022)). Eine kurze Videoanleitung findet sich unter <https://youtu.be/7nVWW-VVhXI>.

Mac OS X Die notwendigen Dateien für Mac OS X sowie eine kurze Anleitung finden sich unter <https://cran.r-project.org/bin/macosx/>. Ähnlich wie im Fall von Windows gibt es auch für Mac OS X eine FAQ-Seite (Iacus et al. (2022)), auf der zusätzliche Informationen zu finden sind.

Linux Es gibt Dateien für

- Debian (<https://cran.r-project.org/bin/linux/debian/>, Ranke (2022))
- OpenSUSE (<https://cran.r-project.org/bin/linux/suse/>, Steuer (2022))

- Fedora, Red Hat Enterprise Linux (RHEL), CentOS, Scientific Linux, Oracle Linux (<https://cran.r-project.org/bin/linux/fedora/>, Úcar (2022))
- Ubuntu (<https://cran.r-project.org/bin/linux/ubuntu/>, Rutter (2022))

Auf den entsprechenden Seiten finden sich jeweils auch kurze Anleitungen zur Installation.

Die offizielle und umfassende Dokumentation zur Installation von R findet sich in dem Manual “R Installation and Administration” (R Core Team (2022d)), welches als HTML- und pdf-Version zur Verfügung gestellt wird. Darin finden sich u.a. Informationen dazu, wie R unter der Verwendung der Quelldateien auf verschiedenen Systemen installiert werden kann.

1.4 Das Arbeiten mit R

Startet man R unter Windows erhält man die in Abbildung 1.1 dargestellte einfache graphische Benutzeroberfläche (GUI). Hier könnte man nun in das Fenster *R Console* der Reihe nach R Befehle eingeben.

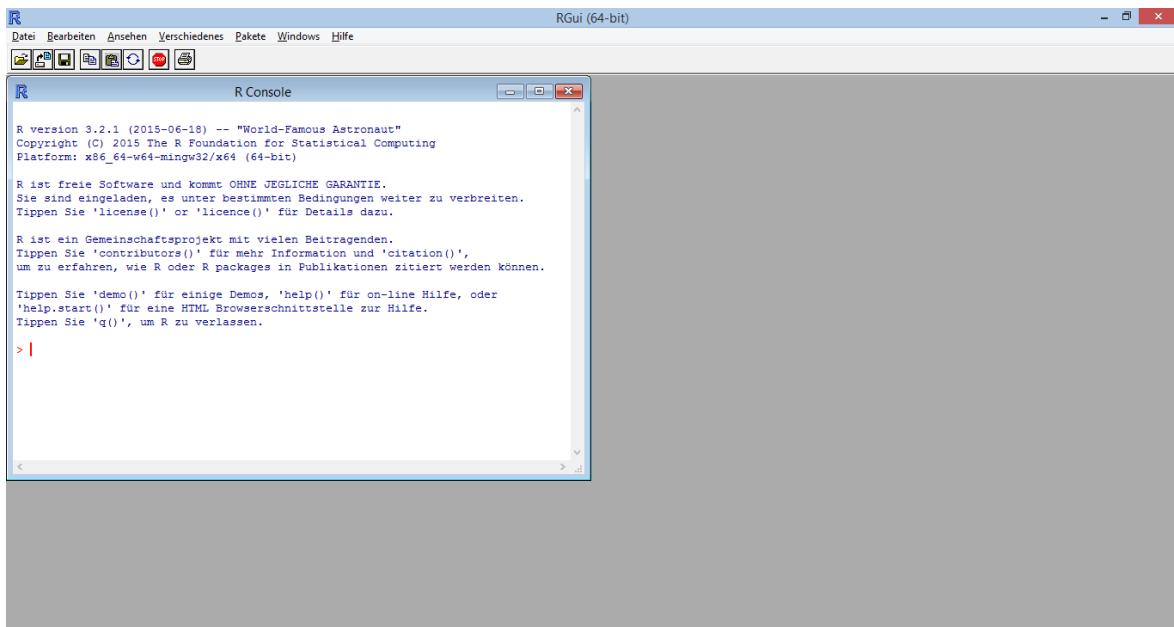


Abbildung 1.1: Die R GUI (64-bit) unter Windows.

Dies mag für einfache Berechnungen pragmatisch sein, aber nicht für Datenanalysen, die man gut dokumentieren sollte und die man unter Umständen in gleicher oder abgewandelter Form zu einem späteren Zeitpunkt wiederholen möchte. In diesem Fall bietet es sich an, eine Textdatei mit den R Befehlen zu erzeugen. Dies kann mit Hilfe irgendeines Texteditors geschehen, wobei es üblich ist, die sich ergebende Datei mit der Endung .r or .R abzuspeichern. Es empfiehlt sich aber noch einen Schritt weiter zu gehen und einen Texteditor mit zusätzlicher Funktionalität oder eine integrierte Entwicklungsumgebung (IDE) zu verwenden. In Abhängigkeit vom Betriebssystem stehen hierfür verschiedene Möglichkeiten zur Auswahl. Den größten Funktionsumfang bietet im Augenblick wohl die freie und open source IDE RStudio (<https://posit.co/>). Diese kann unter Linux, Windows und Mac OS X installiert werden und wird aktuell von mir sowohl im Rahmen meiner Arbeit als auch in meinen Vorlesungen eingesetzt. Eine kurze

Videoanleitung für die Installation unter Windows findet sich unter https://youtu.be/a_xvxutfp0M.

Die Abbildung 1.2 zeigt die RStudio IDE nach der Installation auf einem Ubuntu Linux System. Die Ansicht unter Windows und Mac OS X ist nahezu identisch. Es sind drei der vier Fenster (panes) zu

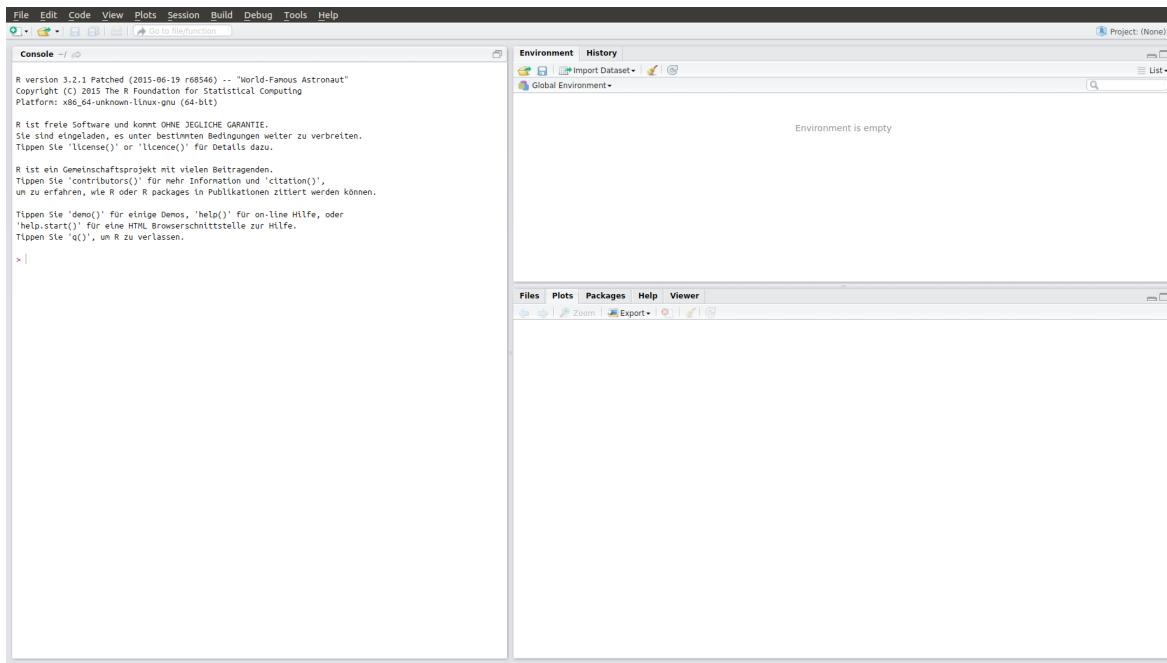


Abbildung 1.2: Die RStudio IDE nach der Installation unter Ubuntu Linux.

sehen. Auf der linken Seite findet sich die *R Console*, in der die aktuelle R Version läuft. Auf der rechten Seite oben werden die beiden Fenster *Environment* und *History* angezeigt. In *Environment* werden alle R Objekte angezeigt, die in der aktuellen R Sitzung geladen oder erzeugt wurden. Da RStudio hier frisch gestartet wurde, ist das *Environment* noch leer. Die *History* enthält die zuletzt ausgeführten R Befehle. Auch diese ist in diesem Fall noch leer. Auf der rechten Seite unten befinden sich die Fenster *Files*, *Plots*, *Packages*, *Help* und *Viewer*. Bei *Files* handelt sich um einen Dateibrowser, der nach dem Start das aktuelle Arbeitsverzeichnis zeigt. Das Fenster *Plots* zeigt die in der aktuellen Sitzung erzeugten Plots an und ist nach dem Start zunächst leer. In *Packages* werden alle auf dem System installierten R Pakete angezeigt und können darüber auch geladen werden. Zusätzlich können auch neue Erweiterungspakete installiert werden. Über das Fenster *Help* können verschiedene Hilfemöglichkeiten (lokal und online) für R und auch RStudio aufgerufen werden. Im *Viewer* lassen sich lokale HTML-Seiten oder auch Webanwendungen anzeigen.

Nach dem Öffnen eines neuen R Skripts über den Menüpunkt *File* → *New File* → *R Script* ist noch ein vierter Fenster zu sehen (vgl. Abbildung 1.3). Es enthält eine leere, noch unbenannte Textdatei – ein sogenanntes R Skript. Wir werden später noch sehen, dass die Texteingabe durch eine Reihe interaktiver Funktionen unterstützt wird, die es insbesondere für Anfänger erleichtern, fehlerfreien R Code zu schreiben. Einzelne Befehle oder auch markierte Befehlsblöcke können über den Menüpunkt *Run* zur Ausführung an die *R Console* geschickt werden. Mit Hilfe des Menüpunkts *Source* kann das gesamte

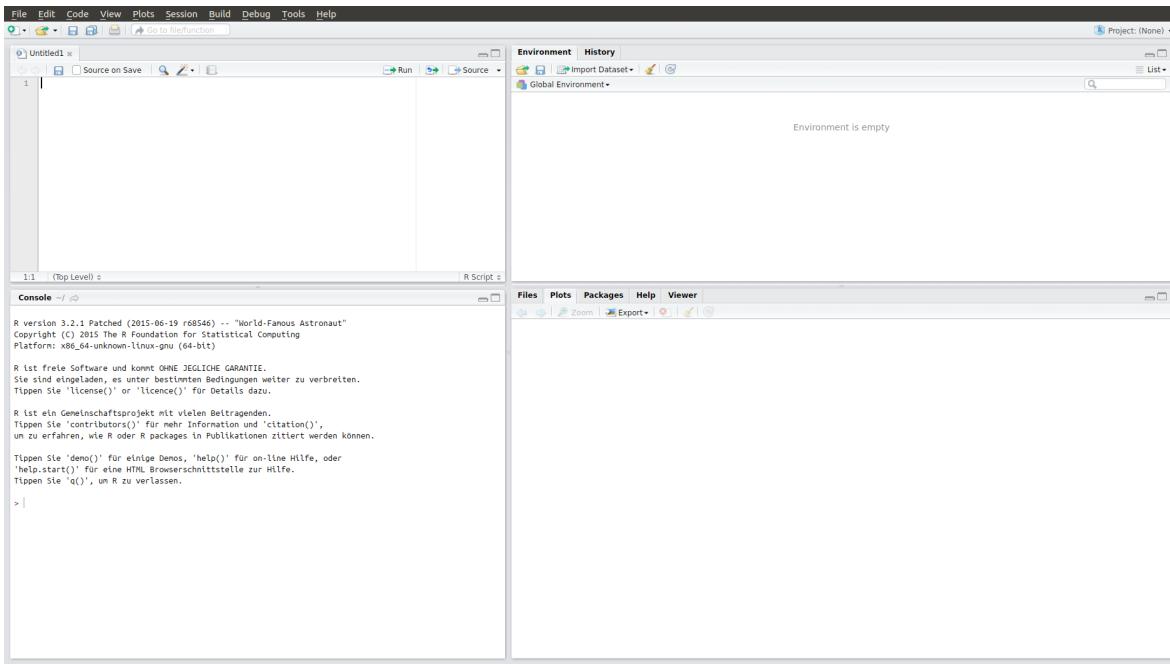


Abbildung 1.3: Die RStudio IDE nach dem Öffnen eines neuen R Skripts unter Ubuntu Linux.

R Skript zur Ausführung an die *R Console* übermittelt werden. Die Anordnung der Fenster kann über den Menüpunkt *Tools* → *Global Options...* → *Pane Layout* nach Wunsch angepasst werden. Auf weitere Details zu RStudio werden wir an gegebenen Stellen im weiteren Verlauf des Buches eingehen.

1.5 Markdown in Verbindung mit R

Neben den klassischen R Skripten wurde in den letzten Jahren das Markdown-System an R angepasst. Markdown ist eine vereinfachte Auszeichnungssprache, deren Ziel es ist, dass bereits die Ausgangsform eines Dokumentes ohne weitere “Übersetzung” lesbar und verständlich ist (Wikipedia contributors (2022e)). Die Umsetzung in R wird R Markdown genannt und wird durch die Firma RStudio (<https://rmarkdown.rstudio.com/>) unterstützt. Eine ausführliche Dokumentation findet sich im Buch von Xie et al. (2018), zu dem es auch eine aktualisierte Online-Version unter <https://bookdown.org/yihui/rmarkdown/> gibt.

Es handelt bei Markdown bzw. R Markdown um ein sehr einfaches und schnell zu lernendes System. Die Ausgangsdateien sind einfach formulierte Textdateien, die in verschiedenste Formate (html, pdf, doc, etc.) übersetzt werden können. Das Übersetzen und die Formatierung der Ergebnisdatei wird durch den Kopf der Datei gesteuert, welcher mit einer Zeile mit dem Eintrag -- beginnt und endet. Die Übersetzung der einfachen Textdateien in die verschiedensten Formate erfolgt meist mit Hilfe der R Pakete "knitr" (Xie (2015a)) und "rmarkdown" (Xie et al. (2018)).

Es gibt zu jedem der folgenden Kapitel dieses Buches eine zugehörige R Markdown Datei (Dateiendung: Rmd), welche den R Code des jeweiligen Kapitels in Form sogenannter R Code Chunks enthält.

R Code Chunks beginnen mit “`r, wobei hier nach dem Buchstaben r zusätzliche Optionen für den entsprechenden R Code angegeben werden können, und enden mit “`. Details hierzu finden sich im Buch von Xie (2015a) sowie auf der zugehörigen Internetseite <https://yihui.org/knitr/>. Zwischen den Code-Chunks kann beliebiger Text ergänzt werden. Jeder Leser des Buches kann auf diese Weise seine eigene Dokumentation der einzelnen Kapitel erzeugen.

1.6 Übungsaufgaben

1. Installieren Sie R und RStudio auf Ihrem Computer, Notebook, etc.
2. Starten Sie RStudio, erzeugen Sie ein neues R Skript und sehen Sie sich anschließend alle zur Verfügung stehenden Fenster inklusive der Menüeinträge genau an.
3. Starten Sie RStudio, erzeugen Sie eine neue R Markdown Datei. Dies erfordert die Installation zusätzlicher Pakete, welche aber automatisch erfolgen sollte. Wählen Sie als Output Format HTML. Sie werden feststellen, dass die erzeugte neue Rmd Datei nicht leer ist, sondern bereits einige Beispiele enthält. Lassen Sie die Datei mit Hilfe von “knit” übersetzen. Sie können sich zur Unterstützung auch das folgende Video auf YouTube <https://youtu.be/p1W93veYjN8> ansehen.
4. Machen Sie sich mit den zur Verfügung stehenden Hilfemöglichkeiten im Fenster *Help* vertraut.
5. Kontrollieren Sie, ob sowohl die Grundpakte als auch die empfohlenen Pakete, auf Ihrem System installiert sind (Fenster *Packages*). Welche R Pakete sind nach dem Start von RStudio angeklickt und somit aktiv – d.h., geladen und können unmittelbar verwendet werden?
6. Machen Sie sich mit R Markdown vertraut.

2 Deskriptive Statistik

Dieses Kapitel behandelt die deskriptive Statistik. Im einzelnen werden folgende Themen behandelt:

- Zusammenspiel von Wahrscheinlichkeitsrechnung, deskriptiver und induktiver Statistik
- Merkmalsarten und Skalenniveaus
- Grundlegende Funktionen für den Datenimport und -export mit R
- Datenimport von Textdateien mit Hilfe von RStudio
- Häufigkeitstabelle, Balken- und Kuchendiagramm
- Modalwert, Quantil, Quartil, Median, Wertebereich, Interquartilsabstand, MAD, Box-und-Whisker Plot
- Kreuztabelle, ϕ -Koeffizient, Pearsons Kontingenzkoeffizient, Cramérs V
- Spearmans ρ , Kendalls τ , Streudiagramm
- Arithmetisches Mittel, geometrisches Mittel, Standardabweichung, Variationskoeffizient, Quartilsdispersionskoeffizient
- Histogramm, Dichteschätzung
- Pearson-(Produkt-Moment-)Korrelation

Es ist nicht nötig, den R Code in diesem Buch abzutippen. Sie finden den R Code für dieses Kapitel in der R Markdown Datei `DeskriptiveStatistik.Rmd`, die Sie von meinem GitHub-Account herunterladen können (Link: <https://github.com/stamats/ESDR/blob/master/DeskriptiveStatistik.Rmd>). Klicken Sie mit der rechten Maustaste auf `Raw`. Dann können Sie das Ziel speichern unter.... Am wenigsten Schwierigkeiten ergeben sich, wenn Sie meine R Markdown Dateien im selben Ordner wie die Daten, welche in den jeweiligen Kapiteln verwendet werden, speichern.

2.1 Grundlagen

Die Abbildung 2.1 gibt einen Überblick über das Zusammenspiel von Wahrscheinlichkeitsrechnung, deskriptiver und induktiver Statistik. Der Ausgangspunkt ist eine **Population** oder **Grundgesamtheit**, welche klar charakterisiert sein muss. Das Ziel ist es, gewisse (neue, wichtige) Erkenntnisse über diese Population zu gewinnen wie etwa, welche Parteien werden bei der nächsten Wahl welchen Stimmanteil erhalten oder welche Erkrankung kommt mit welcher Häufigkeit vor. Eine **Totalerhebung** ist jedoch

Ziel: Neue Erkenntnisse über eine Population

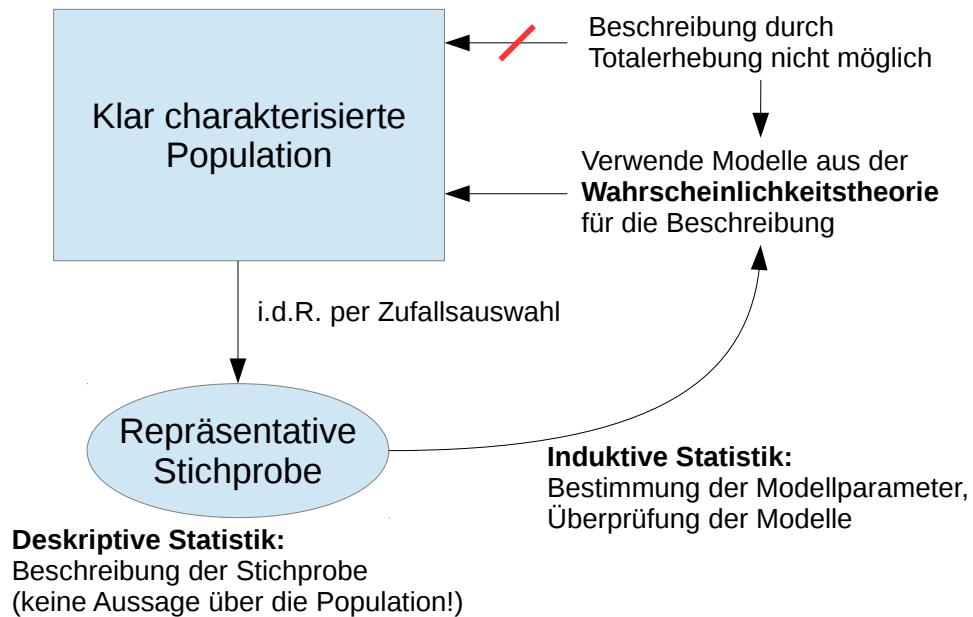


Abbildung 2.1: Zusammenspiel von Wahrscheinlichkeitsrechnung, deskriptiver und induktiver Statistik.

meistens ausgeschlossen, da dies beispielsweise aufgrund der Größe der Population zu teuer wäre oder da sich die Population in einem ständigen zeitlichen Wandel befindet.

Der Ausweg der Statistik besteht darin, **Modelle aus der Wahrscheinlichkeitstheorie** für diese Population aufzustellen, wobei aber die konkreten Modellparameter unbekannt sind und noch bestimmt werden müssen. Zu diesem Zweck wird eine **repräsentative Stichprobe**, meist durch eine Zufallsauswahl, aus der Population „gezogen“. Die Aufgabe der **deskriptiven Statistik** ist es, diese Zufallsstichprobe möglichst gut zu charakterisieren. Die deskriptive Statistik liefert demnach nicht die gesuchten Erkenntnisse über die Grundgesamtheit, sondern beschreibt den hieraus (zufällig) ausgewählten Teil. Hiermit hilft die deskriptive Statistik dabei, die erhobenen Daten kennenzulernen und zu „verstehen“ sowie ungewöhnliche oder fehlerhafte Werte in den Daten ausfindig zu machen. Folglich liefert sie auch einen wichtigen Beitrag für die induktive Statistik, da nur durch die Kenntnis der Daten und der Datenqualität gültige Schlussfolgerungen möglich sind.

Das Ziel der **induktiven Statistik** ist es, ausgehend von einer repräsentativen Stichprobe Rückschlüsse auf die interessierende Population zu ziehen. Dies geschieht, indem die unbekannten Parameter der aufgestellten Wahrscheinlichkeitsmodelle aus den vorliegenden Daten bestimmt (geschätzt) werden. Darüber hinaus können bestehende Modelle mit Hilfe der induktiven Statistik auf ihre Plausibilität hin überprüft werden.

Anmerkung:

Es handelt sich um Modelle; d.h., man sollte nicht davon ausgehen, dass diese Modelle die Wirklichkeit exakt widerspiegeln. Die Modelle liefern nur unter gewissen Voraussetzungen und zu einem gewissen Zeitpunkt eine recht gute Beschreibung der Wirklichkeit. In diesem Sinne ist auch die folgende Aussage des berühmten Statistikers George E.P. Box zu verstehen (Box and Draper (1987, S. 424)):

„Essentially, all models are wrong, but some are useful.“

Das folgende Beispiel zeigt, dass die Wahl des geeigneten Modells entscheidend für das Ergebnis ist und dass identische Daten bei unterschiedlichen Modellannahmen zu sich widersprechenden Ergebnisse führen können.

Beispiel 2.1. Im zweiten Weltkrieg ging es darum, die amerikanischen Bomber besser gegen den Beschuss der Luftabwehr zu schützen. Zu diesem Zweck wurden von den zurückkehrenden Flugzeugen der Ort und die Anzahl der Einschusslöcher analysiert. Das Militär schloss aus diesen Informationen, dass die Stellen, an denen besonders viele Treffer verzeichnet wurden, mit zusätzlichen Panzerungen versehen werden sollten. Dies ist plausibel unter der Annahme, dass die Luftabwehr gezielt diese Stellen der Flugzeuge anvisierte.

Der Statistiker Abraham Wald ging hingegen bei seiner Analyse davon aus, dass sich die Treffer gleichmäßig über die Flugzeuge verteilen müssten (Wald (1980)). Da dies bei den zurückkehrenden Flugzeugen nicht der Fall war, schloss er daraus, dass die nicht zurückkehrenden Flugzeuge gerade an sehr empfindlichen Stellen getroffen wurden und deshalb abstürzten. Folglich gab er die Empfehlung, die Flugzeuge an den Stellen zusätzlich zu panzern, an denen keine oder wenige Treffer bei den zurückkehrenden Flugzeugen festgestellt wurden.

Die Elemente der Population – dies können Personen, Gegenstände, etc. sein – werden durch eine Reihe von **Merkmale** (Variablen) beschrieben. Diese Merkmale wiederum lassen sich in verschiedene **Merkmalsarten** unterteilen wie in Abbildung 2.2 dargestellt. Die Hauptunterteilung erfolgt in qualitative (kategoriale) und quantitative (metrische) Merkmale. Diese beiden Kategorien kann man weiter anhand des sogenannten Skalenniveaus in nominal, ordinal, intervallskaliert und ratioskaliert unterteilen, wobei nominal die niedrigste und ratioskaliert die höchste Stufe darstellt. In Abhängigkeit vom Skalenniveau sind für die Variablen dann gewisse Rechenoperationen erlaubt, wobei die Anzahl der zulässigen Operationen von links (nominal) nach rechts (ratioskaliert) zunimmt. Es ist demnach wichtig, das Skalenniveau der untersuchten Variablen zu kennen, um zum Beispiel im Rahmen der deskriptiven Statistik die erhobenen/gemessenen Werte der Variablen – die sogenannten **Merkmalsausprägungen** – korrekt zu beschreiben.

Anmerkung:

Die Abgrenzungen zwischen den Skalenniveaus sind zum Teil fließend, so wird zum Beispiel ein medizinischer Score mit sehr vielen Ausprägungen in der Praxis oft wie eine metrische Variable behandelt.

Merkmalsart	Qualitativ / Kategorial		Quantitativ / Metrisch	
Skalenniveau	Nominalskala	Ordinalskala	Intervallskala	Ratioskala
Beispiele	Geschlecht, Blutgruppe, Rhesusfaktor	Schulnoten, medizinische Score	Temperatur in °C, Intelligenzquotient	Temperatur in Kelvin, Körpergröße
Hinweise	niedrigstes Niveau	Rangfolge ist definiert	Willkürlicher Nullpunkt, Abstand ist definiert	höchstes Niveau, Natürlicher Nullpunkt, Verhältnis ist definiert
Operationen	$A = B, A \neq B$	$A = B, A \neq B, A < B, A > B$	$A = B, A \neq B, A < B, A > B, d = A - B$	$A = B, A \neq B, A < B, A > B, d = A - B, r = A:B$

Abbildung 2.2: Merkmalsarten und Skalenniveaus.

Mit dem Skalenniveau nimmt auch der Informationsgehalt der Variablen zu. Besteht also bei der Planung einer Studie die Wahl, ein Merkmal mit verschiedenen Variablen zu beschreiben, so liefert die Variable mit dem höchsten Skalenniveau die informativste Beschreibung und sollte daher idealerweise für die Studie ausgewählt werden. Leider ist es in der Praxis oftmals so, dass die Erhebung von informativeren Variablen auch aufwendiger oder teurer ist, weshalb dann unter Umständen doch eine weniger informative Variable für die Studie ausgewählt werden muss.

Wir betrachten ein Beispiel dazu.

Beispiel 2.2. Unser Ziel ist es, die Altersverteilung in einer Stichprobe oder der zugehörigen Population zu charakterisieren. In diesem Fall ist das Geburtsdatum einer Person informativer als das Alter in Jahren oder die Einordnung in eine Altersgruppe, wobei die Erhebung für alle drei Möglichkeiten gleich aufwendig sein dürfte. In diesem Fall empfiehlt es sich folglich, das Geburtsdatum als Variable zu wählen. Diese Auswahl lässt außerdem die Möglichkeit offen, falls die zusätzliche Information nicht benötigt oder später als irrelevant angesehen wird, sich im Rahmen der statistischen Analyse auf das Alter in Jahren oder auf Altersgruppen zu beschränken.

2.2 Exkurs: Datenimport und -export mit R

Bevor wir mit der deskriptiven Analyse beginnen können, müssen wir zunächst einmal eine Studie planen, durchführen und Daten sammeln. Dabei gibt es eine Vielzahl von Dingen zu beachten, auf die wir hier aber nicht näher eingehen können, da es den Rahmen dieses Buches sprengen würde.

In größeren Studien werden die erhobenen Daten meist in eigens dafür angelegten Datenbanken abgelegt, in kleineren Studien in der Regel in einer oder mehreren Dateien eines Tabellenkalkulationsprogramms. Für die statistische Analyse ist die Organisation der Daten in einer Tabelle meist am zielführendsten, wobei hier durchaus auch eine gewisse Redundanz sehr hilfreich sein kann. Für weitere Details verweisen wir auf Bromman and Woo (2018). Eine kurze Übersicht findet sich auch in Kohl and Münch (2022b).

Die meisten Programme bieten heutzutage die Möglichkeit enthaltene Daten zu exportieren, wobei nahezu immer die Möglichkeit besteht, die vorliegenden Daten in eine oder mehrere Textdateien zu exportieren. Wir werden uns daher in diesem Abschnitt nur mit dem Datenimport aus Textdateien befassen. Darüber hinaus bietet R aber eine Vielzahl an weiteren Möglichkeiten, wie etwa das Einlesen von Datendateien von anderen Statistikprogrammen, Tabellenkalkulationsprogrammen oder Schnittstellen zu Datenbanken. Einen Überblick über die vielfältigen Möglichkeiten zum Datenimport und -export gibt das Manual “R Data Import/ Export” (R Core Team (2022b)).

Den Ausgangspunkt für das Einlesen von Textdateien bildet die Funktion `scan`. Mit dieser Funktion können Daten von der Konsole oder einer Textdatei eingelesen werden. Meist ist es jedoch nicht nötig, die Funktion `scan` direkt zu verwenden, sondern man kann die einfacher zu handhabende Funktion `read.table` oder die weiter spezialisierten Funktionen `read.csv`, `read.csv2`, `read.delim` oder `read.delim2` verwenden; siehe auch Tabelle 2.1.

Funktionsname	Beschreibung
<code>scan</code>	Einlesen von Daten von der Konsole oder aus einer Textdatei.
<code>read.table</code>	Liest Daten aus einer Textdatei im Tabellenformat.
<code>read.csv</code>	Spezialfall von <code>read.table</code> mit Dezimalzeichen “.” und Spalten trenner “,” (“englische csv-Datei”).
<code>read.csv2</code>	Spezialfall von <code>read.table</code> mit Dezimalzeichen “,” und Spalten trenner “;” (“deutsche csv-Datei”).
<code>read.delim</code>	Spezialfall von <code>read.table</code> mit Dezimalzeichen “.” und Spalten trenner “\t” (Tabulator).
<code>read.delim2</code>	Spezialfall von <code>read.table</code> mit Dezimalzeichen “,” und Spalten trenner “\t” (Tabulator).

Tabelle 2.1: Übersicht über einige grundlegende Funktionen zum Datenimport mit R.

Der Import von Textdateien kann auch unter Verwendung von RStudio ausgeführt werden. Diese Funktionalität ist vor allem für Anfänger sehr hilfreich. Im Fenster *Environment* findet sich der Menüpunkt

Import Dataset. Nach Auswahl von *From Text ...* öffnet sich ein Fenster für die Auswahl der Textdatei. Nachdem man eine Textdatei ausgewählt hat, öffnet sich das in Abbildung 2.3 gezeigte Fenster. Die

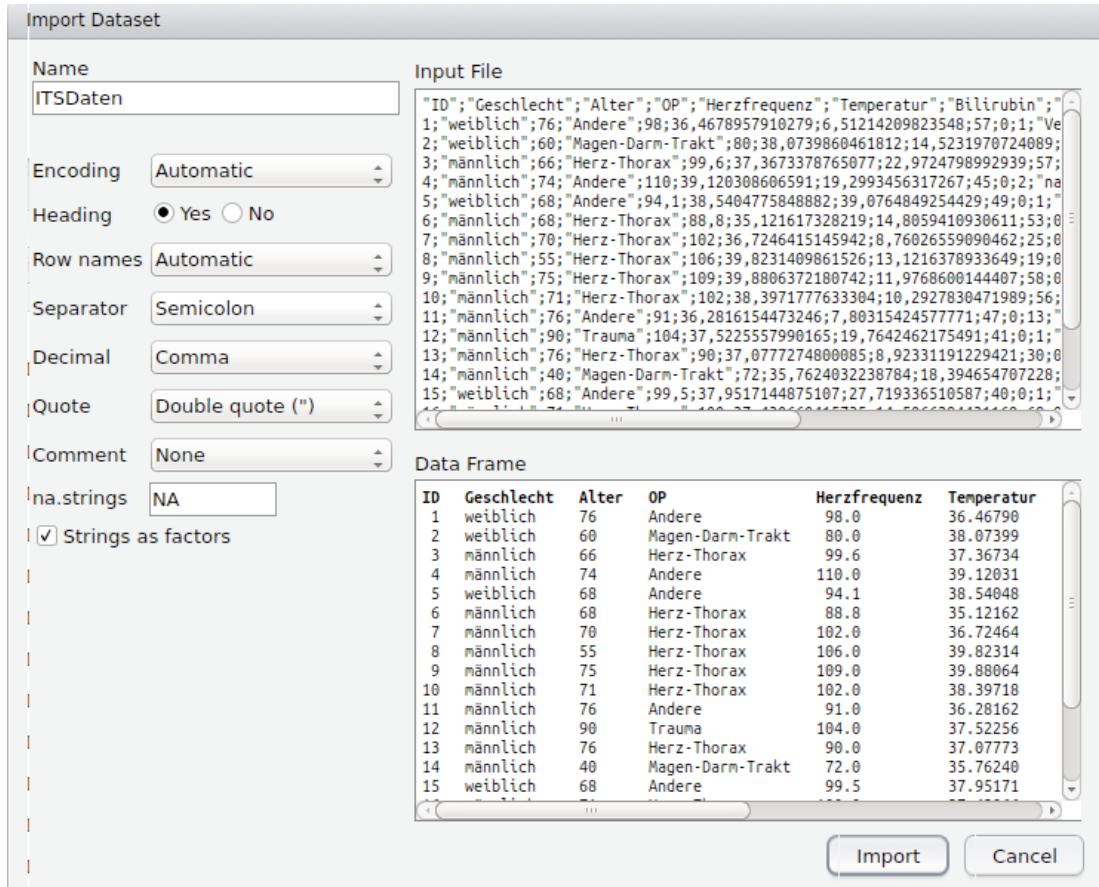


Abbildung 2.3: RStudio Fenster für den Import von Textdateien.

Anpassungsmöglichkeiten dort entsprechen den wichtigsten Optionen der `read.*` Funktionen. Der Import erfolgt unter Verwendung der `read.*` Funktionen, wobei der für den Import verwendete Aufruf anschließend im Fenster *History* zu finden ist. Um die exakte Wiederholbarkeit des Imports sicherzustellen, sollte nach dem Import dieser R Code per *To Source* aus dem Fenster *History* in das aktuelle R Skript übertragen werden.

Anmerkung:

Sollte das Einlesen fehlschlagen, was zum Beispiel bei einem Umlaut oder einem anderen Sonderzeichen im Dateipfad vorkommen kann, wird trotzdem der R Code zum Einlesen im Fenster *History* erzeugt. Indem man diesen R Code wie oben beschrieben in das R Skript überträgt, dort die notwendige Korrektur ausführt (z.B. den Pfad korrigiert) und diesen R Code ausführt, kann die Datei schließlich doch mit den vorgenommenen Einstellungen importiert werden.

Damit das Ergebnis des Einlesevorgangs für spätere Analysen zur Verfügung steht, muss es einer Variable zugewiesen werden. Der Name dieser Variable kann über das Feld *Name* angegeben werden (vgl. Abb. 2.3). Nach dem Einlesen findet sich das Datenobjekt im Bereich *Data* des Fensters *Environment*; siehe Abbildung 2.4. Wird dieses Objekt angeklickt, wird es im Editorfenster zur Ansicht geöffnet.

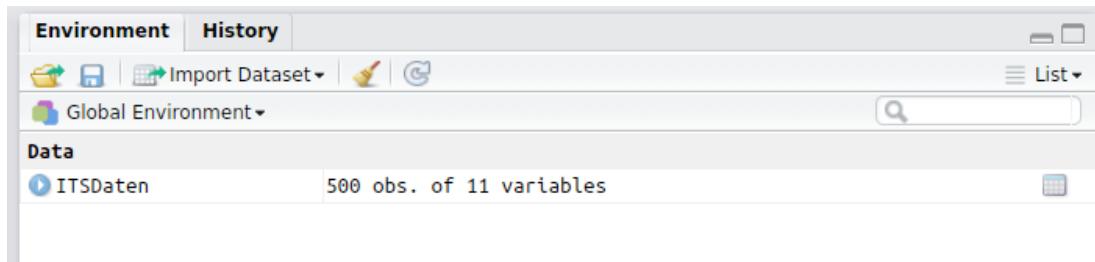


Abbildung 2.4: RStudio Fenster *Environment* mit einem Datenobjekt.

Es handelt sich bei diesem Datenobjekt um einen sogenannten `data.frame`. Der `data.frame` ist die grundlegende Datenstruktur in R für das Abspeichern von Datensätzen. Sie ähnelt einer Tabelle in einem Tabellenkalkulationsprogramm. Die Spalten entsprechen den Variablen (Merkmälern), die Zeilen den Merkmalsausprägungen der einzelnen Beobachtungseinheiten.

Das Gegenstück zu den oben aufgeführten `read.*` Funktionen stellen die Funktionen `write.table`, `write.csv` und `write.csv2` dar. Falls Sie mit deutschen Systemeinstellungen arbeiten, sollten Sie `write.csv2` für den Export verwenden. Die exportierte Datei lässt sich dann in allen gängigen Tabellenkalkulationsprogrammen problemlos öffnen.

Eine weitere Form des Datenimports stellen die R Funktionen `load` und `readRDS` dar. Diese eignen sich dazu, sogenannte `.RData` bzw. `.rds` Dateien einzuladen. Dies sind Dateien, die mit der R Funktion `save` oder `save.image` bzw. `saveRDS` erzeugt wurden. Mit Hilfe dieser Funktionen ist es möglich, einzelne Objekte (`save` oder `saveRDS`) oder den gesamten Speicherinhalt einer R Sitzung (`save.image`) in einer `.RData` bzw. `.rds` Datei abzuspeichern. Dabei kann man zusätzlich wählen, ob die Datei komprimiert werden soll (default) oder nicht.

2.3 Import des ITS-Datensatzes

In diesem Abschnitt importieren wir den Datensatz `ITSDaten.csv`, den wir in diesem Buch auf verschiedene Weise analysieren werden. Es handelt sich um Daten von 500 Patienten auf einer Intensivstation (ITS). Es sind nicht die Daten realer Patienten, sondern die Daten wurden von mir auf der Basis meiner langjährigen Erfahrungen mit den Daten von Intensivpatienten generiert und sind realen Daten hinsichtlich vieler Aspekte sehr ähnlich.

Verwenden Sie für den Import die folgenden Schritte in der hier angegebenen Reihenfolge:

1. Laden Sie den Datensatz von meinen GitHub-Account herunter und speichern Sie ihn auf Ihrem Computer. Vermeiden Sie nach Möglichkeit Umlaute oder andere Sonderzeichen im Pfad zur Datei.

Link: <https://github.com/stamats/ESDR/blob/master/ITSDaten.csv>

Klicken Sie mit der rechten Maustaste auf Raw. Dann können Sie das Ziel speichern unter....

2. Starten Sie RStudio.
3. Wechseln Sie das Arbeitsverzeichnis. Klicken Sie im Fenster *Files* auf ... (am rechten Rand) und öffnen Sie das Verzeichnis, in welches Sie *ITSDaten.csv* gespeichert haben. Anschließend klicken Sie auf *More → Set As Working Directory*.
4. Kontrollieren Sie das Arbeitsverzeichnis. Geben Sie hierzu im Fenster *Console* den folgenden R Code ein

```
1 getwd()
```

Anschließend drücken Sie die *Enter/Return*-Taste. Die Ausgabe sollte dem Verzeichnis entsprechen, in dem die Datei *ITSDaten.csv* gespeichert ist. Falls dies nicht der Fall ist, müssen Sie die obigen Schritte nochmals wiederholen.

5. Speichern Sie die R Markdown Datei *DeskriptiveStatistik.Rmd* im selben Verzeichnis wie den Datensatz und öffnen Sie diese mit RStudio.
6. Importieren Sie den ITS-Datensatz, indem Sie die Zeile mit dem folgenden R Code ausführen. Da der Datensatz unter Linux und mit UTF-8 Encoding erzeugt wurde, setzen wir zusätzlich `fileEncoding = "utf8"`.

```
1 ITSDaten ← read.csv2(file = "ITSDaten.csv", fileEncoding = "utf8")
```

Bewegen Sie dazu im Fenster mit der R Markdown Datei den Cursor in die Zeile mit diesem R Code und klicken sie auf *Run*. Damit wird der R Code in das Fenster *Console* kopiert und ausgeführt. Es sollte keine weitere Ausgabe erfolgen. Sollte eine Fehlermeldung ausgegeben werden, so lautet diese meist

```
Error in file(file, "rt"): kann Verbindung nicht öffnen
```

D.h., es hat entweder das Speichern der Datei oder das Anpassen des Arbeitsverzeichnisses nicht funktioniert. Kontrollieren Sie beides (Schritte 3+4) und führen Sie dann den R Code wie oben beschrieben nochmals aus.

Alternativ können Sie die Daten auch direkt von meiner Webseite importieren.

```
1 Link ← "https://raw.githubusercontent.com/stamats/ESDR/master/ITSDaten.csv"
2 ITSDaten ← read.csv2(file = Link, fileEncoding = "utf8")
```

Auch in diesem Fall sollten Sie Ihr Arbeitsverzeichnis kontrollieren und es auf das Verzeichnis setzen, in dem sich die entsprechende R Markdown Datei befindet.

Als weitere Alternative können Sie auch die Importfunktion von RStudio verwenden wie im Abschnitt 2.2 beschrieben. Achten Sie darauf, dass Ihre Einstellungen den Einstellungen, die in Abbildung 2.3 zu sehen sind, entsprechen.

7. Kontrollieren Sie, ob im Bereich *Data* des Fensters *Environment* das Objekt `ITSDaten` zu sehen ist (vgl. Abb. 2.4). Es muss sich um ein Objekt vom Typ `data.frame` mit 500 Beobachtungen (`obs.`) von 11 Variablen handeln. Falls dies der Fall ist, war der Import erfolgreich.

Anmerkung:

Im Schritt 6 wurde dem Namen `ITSDaten` das Ergebnis des Imports mittels `read.csv2` zugewiesen. Hierbei kam der sogenannte Zuweisungsoperator `<-` zum Einsatz. D.h., die Daten sind damit in einem `data.frame` mit dem Namen `ITSDaten` gespeichert und wir können dieses Objekt für weitere Analysen benutzen.

Trotz eines auf den ersten Blick erfolgreichen Imports, ist es immer noch möglich, dass der Datensatz nicht wie gewünscht eingelesen wurde. Deshalb kann ich nur sehr empfehlen, den Importvorgang genauer zu kontrollieren. Zunächst einmal kann man sich mittels der Funktion `View` den eingelesenen Datensatz genauer ansehen – soweit er nicht zu groß ist.

```
1 View(ITSDaten)
```

Dies kann in gleicher Weise durch Anklicken des Names des Datensatzes im Fenster *Environment* von RStudio erreicht werden. Damit lässt sich zum Beispiel erkennen, ob die Spalten- und gegebenenfalls Zeilennamen korrekt übernommen wurden, ob die Einträge in den Spalten richtig sind und ob es Leerzeilen oder Leerspalten gibt. Da verschiedene Datentypen in dieser Ansicht gleich oder sehr ähnlich aussehen, sollte man die Struktur der Daten noch etwas genauer untersuchen. Hierfür ist die Funktion `str` vorgesehen.

```
1 str(ITSDaten)
```

```
'data.frame': 500 obs. of 11 variables:
 $ ID : int 1 2 3 4 5 6 7 8 9 10 ...
 $ Geschlecht : chr "weiblich" "weiblich" "männlich" "männlich" ...
 $ Alter : int 76 60 66 74 68 68 70 55 75 71 ...
 $ OP : chr "Andere" "Magen-Darm-Trakt" "Herz-Thorax" "Andere" ...
 $ Herzfrequenz : num 98 80 99.6 110 94.1 88.8 102 106 109 102 ...
 $ Temperatur : 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 ...
 $ Leberversagen: int 0 0 0 0 0 0 0 0 0 0 ...
 $ LOS : int 1 2 1 2 1 1 1 1 1 3 ...
 $ Ergebnis : chr "Verstorb" "nach Hause" "Pflege/REHA" "nach Hause" ...
```

Man kann die entsprechende Information in RStudio auch im Fenster *Environment* angezeigt bekommen, indem man im Bereich *Data* auf das blaue Pfeilsymbol vor `ITSDaten` klickt. Das Ergebnis ist in Abbildung 2.5 zu sehen.

Die beiden Anzeigen sind in diesem Fall nicht identisch. Dies liegt daran, dass der Import, der in Abbildung 2.5 zu sehen ist, mit einer älteren Version von R erzeugt wurde. In der R Version 4.0.0 wurde hier eine wichtige Umstellung vorgenommen und das Argument `stringsAsFactors` von `TRUE` auf `FALSE`

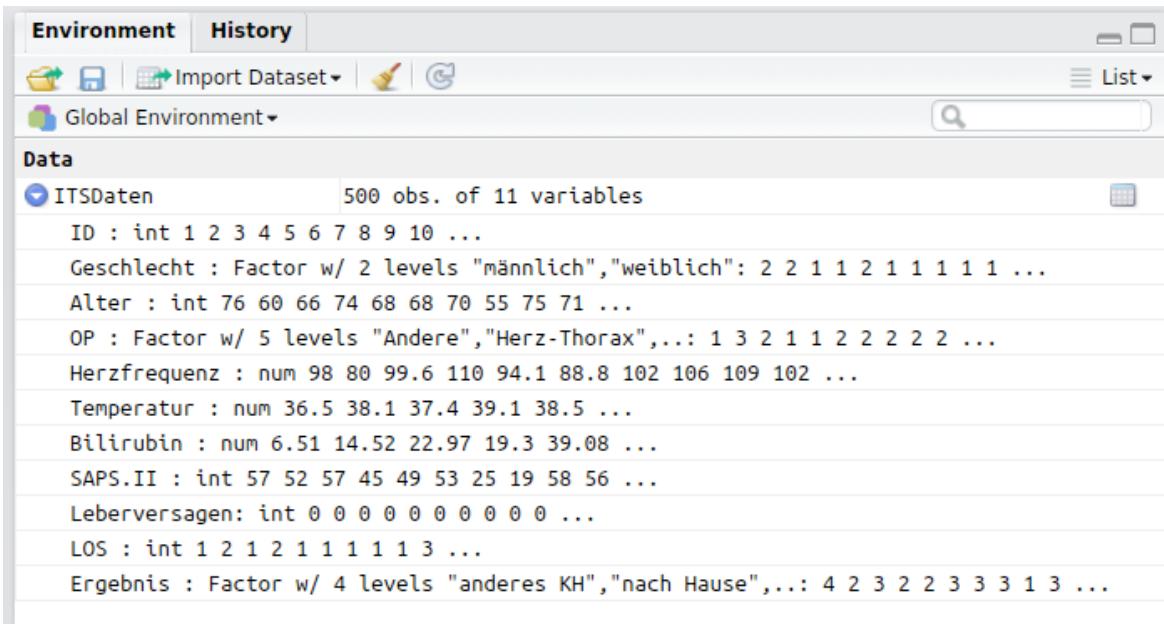


Abbildung 2.5: Anzeige der genaueren Struktur der Daten in RStudio.

verändert. Da es für die Zwecke unserer Analysen aber Sinn macht, die eingelesenen “string” Variablen als Faktoren (kategoriale Variablen) aufzufassen, lesen wir die Daten erneut ein. Dieses Mal mit der zusätzlichen Einstellung `TRUE` für das Argument `stringsAsFactors`.

```
1 ITSDaten ← read.csv2(file = "ITSDaten.csv", fileEncoding = "utf8",
2                         stringsAsFactors = TRUE)
3 ## bzw.
4 Link ← "https://raw.githubusercontent.com/stamats/ESDR/master/ITSDaten.csv"
5 ITSDaten ← read.csv2(file = Link, fileEncoding = "utf8",
6                         stringsAsFactors = TRUE)
```

Wir werfen erneut einen Blick auf die Struktur der eingelesenen Daten.

```
1 str(ITSDaten)
```

```
'data.frame': 500 obs. of 11 variables:
 $ ID : int 1 2 3 4 5 6 7 8 9 10 ...
 $ Geschlecht : Factor w/ 2 levels "männlich","weiblich": 2 2 1 1 2 1 1 1 1 1 ...
 $ Alter : int 76 60 66 74 68 68 70 55 75 71 ...
 $ OP : Factor w/ 5 levels "Andere","Herz-Thorax",...: 1 3 2 1 1 2 2 2 2 2 ...
 ...
 $ Herzfrequenz : num 98 80 99.6 110 94.1 88.8 102 106 109 102 ...
 $ Temperatur : 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 ...
 $ Leberversagen: int 0 0 0 0 0 0 0 0 0 0 ...
 $ LOS : int 1 2 1 2 1 1 1 1 1 3 ...
```

```
$ Ergebnis : Factor w/ 4 levels "anderes KH","nach Hause",...: 4 2 3 2 2 3
 3 3 1 3 ...
```

Der Datensatz enthält demnach die folgenden Variablen:

ID: eine fortlaufende Nummer (`integer` = ganze Zahl) von 1 bis 500 zur Identifikation der Patienten

Geschlecht: eine nominale Variable (`Factor`) mit den Ausprägungen: männlich und weiblich

Alter: das Lebensalter in Jahren (`integer`)

OP: Art der Operation, nominale Variable (`Factor`) mit den Ausprägungen: Andere, Herz-Thorax, Magen-Darm-Trakt, Neuro und Trauma

Herzfrequenz: maximale Herzfrequenz in Herzschlägen pro Minute (`numeric` = reelle Zahl) während des gesamten Aufenthalts auf der ITS

Temperatur: maximale Körpertemperatur in °C (`numeric`) während des gesamten Aufenthalts auf der ITS

Bilirubin: maximaler Bilirubinwert in $\mu\text{mol/l}$ (`numeric`) während des gesamten Aufenthalts auf der ITS. Der rote Blutfarbstoff wird abgebaut und es entsteht als Zwischenstufe Bilirubin, eine gelbliche Substanz. Normal sind Werte unter 21 $\mu\text{mol/l}$, wobei höhere Werte zum Beispiel auf Leberprobleme hindeuten können (Wikipedia contributors (2022b)).

SAPS.II: Wert des SAPS-II Scores (`integer`) zum Zeitpunkt der Aufnahme auf die ITS. Der Score spiegelt den physiologischen Zustand eines Patienten wider und dient zur Einschätzung der Erkrankungsschwere. Je höher der Score, um so schlimmer die Erkrankung. Möglich sind Werte zwischen 0 und 163, wobei den Werten eine Sterbewahrscheinlichkeit (0-100%) zugeordnet werden kann (Wikipedia contributors (2021)).

Leberversagen: Vorliegen eines Leberversagens (`integer`), wobei 0 für nein und 1 für ja steht; d.h., es handelt sich genau genommen um eine numerisch kodierte nominale Variable.

LOS: Aufenthaltsdauer (length of stay) in Tagen auf der ITS (`integer`)

Ergebnis: Art der Entlassung von der ITS (`Factor`). Die Möglichkeiten sind: anderes KH (Krankenhaus), nach Hause, Pflege/REHA und Verstorben.

Anmerkung:

Der Name der Variable `SAPS.II` wurde beim Einlesen angepasst. Die Spaltenüberschrift `SAPS II` enthält ein Leerzeichen und ist daher kein syntaktisch korrekter Variablenname in R. Derartige Anpassungen werden beim Import automatisch vorgenommen. Man kann dies durch Setzen des Parameters `check.names` unterbinden. Der entsprechende Aufruf wäre

```
1 ITSDaten ← read.csv2(file = "ITSDaten.csv", fileEncoding = "utf8",
2                         check.names = FALSE)
```

Die Verwendung von `check.names = FALSE` sollte erst bei mehr Erfahrung im Umgang mit R verwendet werden, da es ansonsten zu unerwünschten Nebeneffekten und Problemen kommen kann.

2.4 Exkurs: Erweiterungspakete installieren

Stellen Sie zunächst sicher, dass Sie eine aktive Internetverbindung haben. Für die Installation verwendet man die Funktion `install.packages`. Wir installieren im Folgenden die Erweiterungspakete "DescTools" (Andri et mult. al. (2022)), "scales" (Wickham and Seidel (2022)), "ggplot2" (Wickham (2009)) und "MKdescr" (Kohl (2022a)), welche wir in diesem Kapitel nutzen wollen.

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

Mit der Funktion `c` (kurz für `concatenate`), werden die vier Paketnamen zu einem Vektor zusammengefügt und können dann zusammen installiert werden.

Anmerkung:

Bei der ersten Installation eines Erweiterungspaketes startet die Installation unter Umständen nicht sofort, sondern es öffnet sich ein Fenster, in dem der Pfad zur Bibliothek, in die das Paket installiert werden soll, abgefragt wird. Hier empfiehlt es sich bei der vorgegeben Standardeinstellung Ihres Betriebssystems zu bleiben; d.h., diese, falls nötig, auszuwählen und zu bestätigen.

Alternativ kann man in RStudio im Fenster *Packages* über den Menüpunkt *Install* ein Fenster für die Installation öffnen; siehe Abbildung 2.6. Die in diesem Fenster von RStudio automatisch vorgenommenen

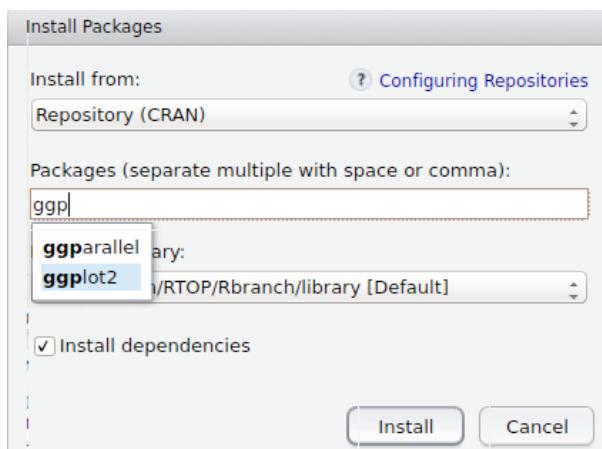


Abbildung 2.6: Installation von R Paketen in RStudio.

Einstellungen sollte man nur verändern, wenn man reichlich Erfahrung im Umgang mit R hat. Insbesondere ist es wichtig, dass *Install dependencies* ausgewählt ist. Die meisten R Pakete benötigen nämlich für ein korrektes Funktionieren andere R Pakete und mit dieser Option wird sichergestellt, dass diese zusätzlichen R Pakete automatisch mit installiert werden.

Anmerkung:

Bitte führen Sie die Installation nur einmal aus. Die Pakete sind damit installiert und müssen nicht bei jedem erneuten Durchlauf einer Analyse erneut installiert werden. Ganz im Gegenteil kann das wiederholte Installieren die vorhandene Installation beschädigen und das entsprechende Paket lässt sich nicht mehr benutzen (laden). In diesem Fall sollten Sie RStudio beenden, ohne den Workspace

zu speichern und anschließend wieder neu starten. Nach dem Starten sollte demnach Ihr Workspace leer sein. Ist dies nicht der Fall, befindet sich in Ihrem aktuellen Arbeitsverzeichnis eine Datei namens `.RData`, welche beim Start automatisch geladen wird. Beenden Sie RStudio erneut und löschen Sie diese Datei. Anschließend starten Sie RStudio nochmals neu. Führen Sie anschließend die Funktion `remove.packages` mit dem Namen des Paketes aus, welches sich nicht mehr laden lässt und installieren Sie das Paket anschließend wieder; z.B.

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

Wie in Abschnitt 1.2 dargestellt, gibt es einige tausend R Pakete. Daher ist es sinnvoll, dass installierte Pakete nicht automatisch geladen werden. Damit würde das laufende System mit zunehmender Anzahl von installierten Paketen immer schwerfälliger und langsamer. Stattdessen müssen alle Pakete mit Ausnahme der Grundpakete (vgl. Abschnitt 1.2) explizit geladen werden. Dies geschieht mit Hilfe der Funktion `library`. Wir laden jetzt die vier eben installierten Pakete.

```
1 library(DescTools)
2 library(scales)
3 library(ggplot2)
4 library(MKdescr)
```

Ein erfolgreiches Laden eines Paketes ist in der Regel daran zu erkennen, dass es keine Fehlermeldung gibt. Manche Pakete geben eine kurze Meldung aus. Manchmal kommt es auch zu Warnmeldungen. Diese sollte man sorgfältig lesen, man kann diese aber in den meisten Fällen ignorieren. Ein Wiederholtes Ausführen von `library` ist unproblematisch. Die Funktion erkennt, dass das angegebene Paket bereits geladen ist und versucht nicht, dieses ein weiteres Mal zu laden.

Anmerkung:

Manchmal kann eine Installation auch fehl schlagen, wenn die Mitinstallation von abhängigen Paketen scheitert. In jedem Fall sollte man sich bei einem Fehlschlagen der Installation oder des Ladens eines Paketes die Fehlermeldung(en) sehr genau lesen und versuchen, die Fehler Schritt für Schritt zu beheben. Durch die weite Verbreitung von R kann bei unbekannten oder unklaren Fehlermeldungen eine Internetsuche meistens helfen, Lösungen zur Behebung eines Fehlers zu finden.

2.5 Kategoriale Variablen

2.5.1 Univariate Analyse

Wir betrachten die Variablen zunächst einzeln (univariat) und beginnen mit nominalen Variablen. In diesem Fall haben wir es also mit einer Variable zu tun, deren Ausprägungen eine Reihe von möglichen Bezeichnungen (Namen) sind, die gleichwertig nebeneinander stehen. Beispiele solcher Merkmale sind Geschlecht, Blutgruppe, Rhesusfaktor oder auch OP und Ergebnis wie im Fall unseres ITS-Datensatzes,

der in Abschnitt 2.3 genauer beschrieben ist.

Falls Sie dies noch nicht getan haben, importieren Sie bitte zunächst den ITS-Datensatz wie in Abschnitt 2.3 beschrieben. Installieren Sie außerdem die Erweiterungspakete "DescTools" (Andri et mult. al. (2022)), "scales" (Wickham and Seidel (2022)), "ggplot2" (Wickham (2009)) und "MKdescr" (Kohl (2022a)) wie in Abschnitt 2.4 beschrieben.

Im Fall von nominalen Variablen besteht die deskriptive Statistik aus der Berechnung und Darstellung von absoluten und relativen Häufigkeiten. Mit dem folgenden R Code berechnen wir die **absoluten Häufigkeiten** für die Operationsarten der ITS-Patienten.

```
1 table(ITSDataen$OP)
```

	Andere	Herz-Thorax	Magen-Darm-Trakt	Neuro
	121	223	79	46
Andere				
Trauma	31			

Die Berechnung erfolgt durch die Funktion `table`. Mit dem Symbol `$` können wir auf die Variablen des Datensatzes (`data.frame`) zugreifen. In diesem Fall greifen wir auf die Variable `OP` zu, welche die Operationsarten enthält. Die **relativen Häufigkeiten** erhalten wir, indem wir diese Werte noch durch die Anzahl der Patienten teilen. Man nennt dies auch die **empirische Häufigkeitsverteilung**. Es ist nicht zu empfehlen, hier die Zahl 500 zu verwenden, auch wenn das natürlich richtig wäre. Besser und allgemeingültiger ist der R Code, wenn wir stattdessen durch die Anzahl der Zeilen des Datensatzes teilen, die wir mit der Funktion `nrow` berechnen können. Damit wird der folgende R Code auch dann noch ein korrektes Ergebnis liefern, wenn der Datensatz abgeändert wird und Zeilen (Patienten) gelöscht oder hinzugefügt werden.

```
1 table(ITSDataen$OP) / nrow(ITSDataen)
```

	Andere	Herz-Thorax	Magen-Darm-Trakt	Neuro
	0.242	0.446	0.158	0.092
Andere				
Trauma	0.062			

Wir sehen also, dass knapp die Hälfte der Patienten sich einer Herz-Thorax Operation unterzogen hat. Man nennt die häufigste Merkmalsausprägung auch den **Modalwert** oder **Modus**. An zweiter Stelle liegen die anderen OP-Arten, gefolgt von OPs am Magen-Darm-Trakt. Die wenigsten OPs waren durch ein Trauma bedingt, etwas mehr waren Neuro-OPs. Wir können die absoluten und relativen Häufigkeiten auch schneller mit Hilfe der Funktion `Freq` aus dem Paket "DescTools" (Andri et mult. al. (2022)) berechnen.

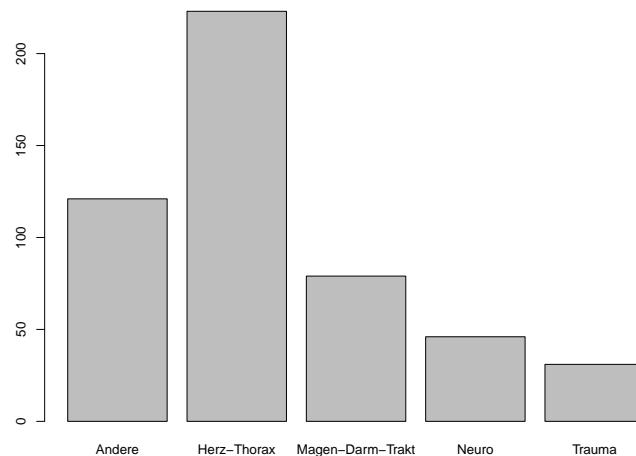
```
1 Freq(ITSDataen$OP)
```

	level	freq	perc	cumfreq	cumperc
1	Andere	121	24.2%	121	24.2%
2	Herz-Thorax	223	44.6%	344	68.8%
3	Magen-Darm-Trakt	79	15.8%	423	84.6%
4	Neuro	46	9.2%	469	93.8%
5	Trauma	31	6.2%	500	100.0%

Die Funktion **Freq** berechnet außerdem die **kumulativen** absoluten und relativen Häufigkeiten.

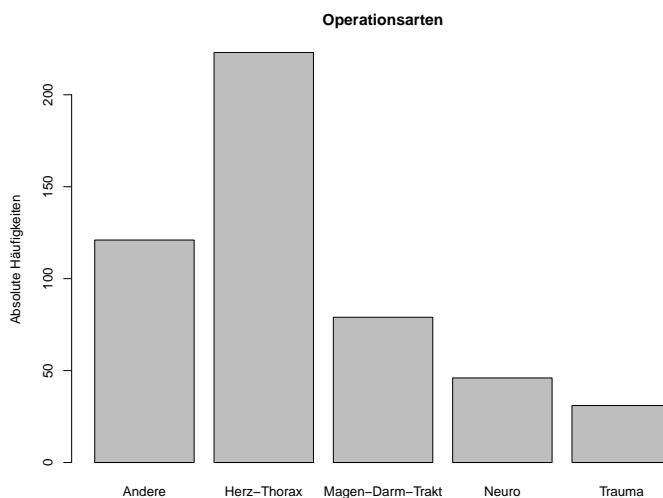
Für die graphische Darstellung der absoluten und relativen Häufigkeiten eignen sich am besten **Balkendiagramme**. Wir stellen zunächst die absoluten Häufigkeiten mit Hilfe der Funktion **barplot** dar.

```
1 barplot(table(ITSDataen$OP))
```



Wir ergänzen das Balkendiagramm um einen Titel (Argument **main**) und beschriften die y-Achse (Argument **ylab**).

```
1 barplot(table(ITSDataen$OP), main = "Operationsarten",
2 ylab = "Absolute Häufigkeiten")
```



Es gibt eine Vielzahl weiterer Parameter, mit denen wir die Grafik zusätzlich nach unseren Wünschen anpassen können. Wir werden noch einige davon im Verlauf des Buches kennenlernen. Verschiedene Beispiele zur Gestaltung von Balkendiagrammen finden sich auch in der Hilfeseite zu `barplot`, die man durch den Aufruf von `?barplot` im Fenster *Help* angezeigt bekommt. Alternativ kann man in RStudio über das Suchfeld im Fenster *Help* nach Hilfe suchen und die Seite so finden und anzeigen lassen.

Aktuelle Versionen von RStudio bieten eine weitere interaktive Hilfe. Beginnt man in einem R Code Chunk mit der Eingabe von Code, so werden automatisch die Namen passender R Objekte und mit etwas Verzögerung passende Hilfe dazu angezeigt; siehe Abbildung 2.7. Durch Drücken der Taste F1 kann

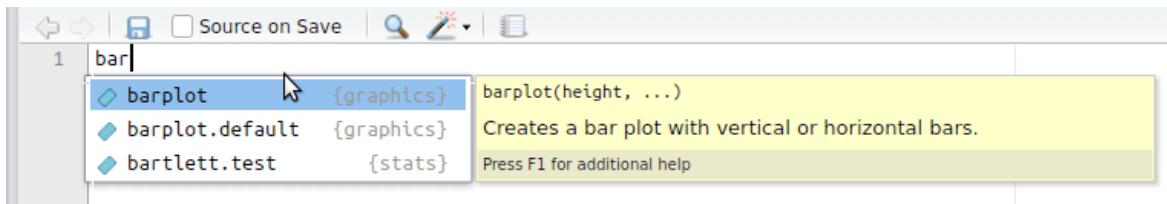


Abbildung 2.7: Interaktive kontextbezogene Hilfe in RStudio.

zusätzlich noch die zugehörige Hilfeseite im Fenster *Help* geöffnet werden.

Ein Balkendiagramm der relativen Häufigkeiten kann mit einem sehr ähnlichen R Code wie im Fall der absoluten Häufigkeit erzeugt werden. Es müssen nur die absoluten durch die relativen Häufigkeiten ersetzt werden. Neben der Standardgrafik sind in R aber auch andere Grafiksysteme implementiert. Das heute neben dem Standardsystem am häufigsten eingesetzte Grafiksystem dürfte die Implementation der sogenannten "grammar of graphics" im Paket "ggplot2" sein (Wickham (2009)). Es gehört zu den Paketen, die einen wichtigen Beitrag zum Erfolg von R geleistet haben. Wir verwenden daher dieses System, um die relativen Häufigkeiten darzustellen.

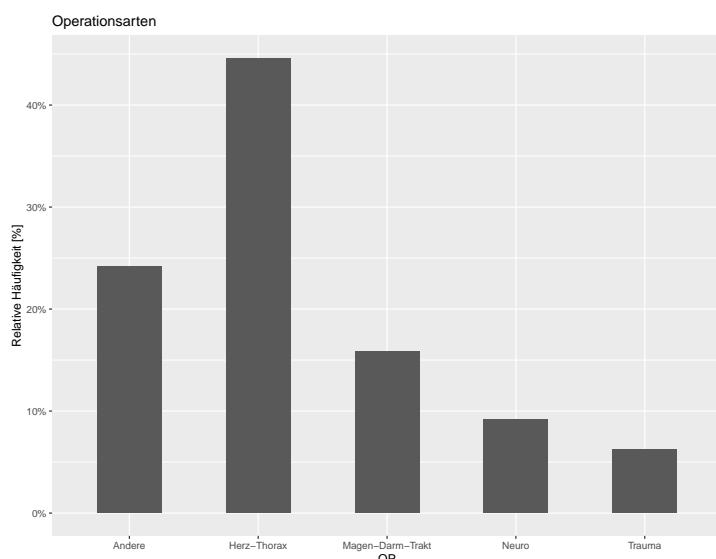
Wir erzeugen ein Balkendiagramm der relativen Häufigkeiten mit Hilfe der Funktionen `ggplot` und

`geom_bar`, wobei wir mit dem Parameter `width` die Breite der Balken steuern können. Mit Hilfe der Funktion `aes` können wir festlegen, wie die Daten für die Darstellung verändert werden sollen. Im vorliegenden Fall gehen wir über zu den relativen Häufigkeiten, welche durch den feststehenden Ausdruck `after_stat(count)/sum(after_stat(count))` berechnet werden. Für die Skalierung der y-Achse gibt es viele Möglichkeiten, die man durch `scale_y_*` erzeugen kann. Im aktuellen Fall verwenden wir eine stetige Skalierung der y-Achse, wobei wir das Beschriftungsformat mit Hilfe der Funktion `percent_format` auf Prozent ändern. Die Funktion `percent_format` ist im Paket "scales" (Wickham and Seidel (2022)) enthalten. Die Funktionen `ggtitle` und `ylab` schließlich verwenden wir für den Titel des Plots und die Beschriftung der y-Achse.

```

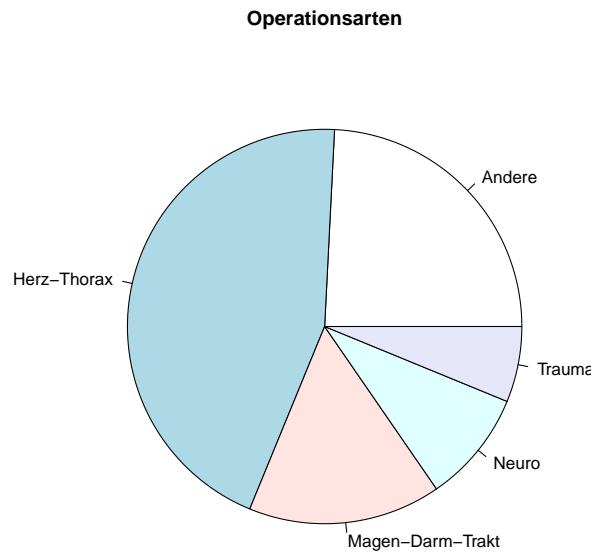
1 ## Anlegen der Daten
2 ggplot(ITSDataen, aes(x=OP)) +
3   ## Hinzufügen der Balken für die relativen Häufigkeiten
4   geom_bar(aes(y = after_stat(count)/sum(after_stat(count))), width = 0.5) +
5   ## Relative Häufigkeiten in Prozent
6   scale_y_continuous(labels = percent_format(accuracy = 1)) +
7   ## Titel und Beschriftung der y-Achse
8   ggtitle("Operationsarten") + ylab("Relative Häufigkeit [%]")

```



Anstelle von Balkendiagrammen werden in der Praxis auch häufig Tortendiagramme (Kuchendiagramme) verwendet. Auch dies ist natürlich mit R möglich. Die Funktion dafür lautet `pie`.

```
1 pie(table(ITSDataen$OP), main = "Operationsarten")
```



Diese Art von Diagramm hat einige Nachteile (vgl. auch Kapitel 3). In der Hilfeseite von `pie` steht dazu:

„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.“

Es ist demnach besser Balken- oder Punktediagramme zu verwenden, da mit diesen das Auge die dargestellte Information schneller und besser erfassen kann.

Anmerkung:

Mit der Verwendung von Farben und der richtigen Verwendung von Diagrammen werden wir uns im Kapitel 3 nochmals genauer beschäftigen.

Im Folgenden nehmen wir an, dass die Kategorien zusätzlich angeordnet sind; d.h., wir haben es mit **ordinalen Variablen** zu tun. Durch die Möglichkeit der Anordnung ergeben sich eine Reihe zusätzlicher Möglichkeiten für die statistische Analyse. Insbesondere können **Quantile** definiert werden, welche vielfache Einsatzmöglichkeiten in der Statistik besitzen.

Definition 2.3 (Quantil). *Gegeben seien Beobachtungen $x_1, x_2, \dots, x_n \in \mathbb{R}$ ($n \in \mathbb{N}$) und wir bezeichnen mit $x_{(1)}, x_{(2)}, \dots, x_{(n)}$ die aufsteigend sortierten Beobachtungen. Dann ist das α -Quantil für $\alpha \in (0, 1)$ definiert als*

$$q_\alpha = \begin{cases} x_{\lceil ceiling(n\alpha) \rceil} & \text{falls } n\alpha \notin \mathbb{Z} \\ [x_{(n\alpha)}, x_{(n\alpha+1)}) & \text{falls } n\alpha \in \mathbb{Z} \end{cases} \quad (2.1)$$

In der folgenden Bemerkung finden sich einige zusätzliche Erklärungen und Ergänzungen zum α -Quantil.

Bemerkung 2.4. (a) Falls $n\alpha$ keine ganze Zahl ist, so ist das α -Quantil also gerade die $\text{ceiling}(n\alpha)$ -te Beobachtung. Hierbei bedeutet “ ceiling ”, dass zur nächst größeren ganzen Zahl aufgerundet wird. In R gibt es hierfür die Funktion `ceiling`; zum Beispiel

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

```
[1] 3 4
```

Im Fall, dass $n\alpha$ jedoch eine ganze Zahl ist, ist das α -Quantil nicht eindeutig und jeder Wert in dem Intervall $[x_{(n\alpha)}, x_{(n\alpha+1)})$ ist ein α -Quantil. Für die Praxis ist das eine nicht zufriedenstellende Situation. Daher gibt es eine Reihe von Vorschlägen, welchen Wert man in diesem Intervall als Repräsentanten des α -Quantils angeben soll. Der naheliegendste Ansatz ist, den Intervallmittelpunkt zu nehmen. In R sind in der Funktion `quantile` neun verschiedene Vorschläge implementiert; siehe auch Beispiel 2.5.

(b) Wichtige Spezialfälle von Quantilen sind **Perzentile** für $\alpha \in \{0.01, 0.02, \dots, 0.99, 1.00\}$, **Quartile** für $\alpha \in \{0.25, 0.50, 0.75\}$ und der **Median** für $\alpha = 0.5$.

Beispiel 2.5. Wir betrachten die Zahlenreihe $2, 4, 6, \dots, 20$ und wollen hiervon das 20-zigste Perzentil berechnen; d.h., $\alpha = 0.2$. Es folgt $n\alpha = 10 \cdot 0.2 = 2$ und damit ist das 20-zigste Perzentil jeder Wert in dem abgeschlossenen Intervall $[x_{(2)}, x_{(3)}) = [4, 6)$. Damit wir die entsprechende Berechnung in R durchführen können, müssen wir zunächst einmal die Daten eingeben. Im vorliegenden Fall eignen sich hierfür die Funktionen `c` (kurz für concatenate) oder `seq` (kurz für sequence).

```
1 ## Aneinanderhängen von Zahlen zu einem Vektor
2 x <- c(2, 4, 6, 8, 10, 12, 14, 16, 18, 20)
3 ## Zahlenfolge: Beginn = 2, Ende = 20, Abstand = 2
4 x <- seq(from = 2, to = 20, by = 2)
```

Das Ergebnis beider Funktionsaufrufe ist der Zahlenvektor `x` mit den entsprechenden Einträgen. Wir wenden die Funktion `quantile` auf den Vektor an.

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

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

```
20%
5.6
```

```
1 ## Von SAS Software verwendet
2 quantile(x, type = 3, probs = 0.2)
```

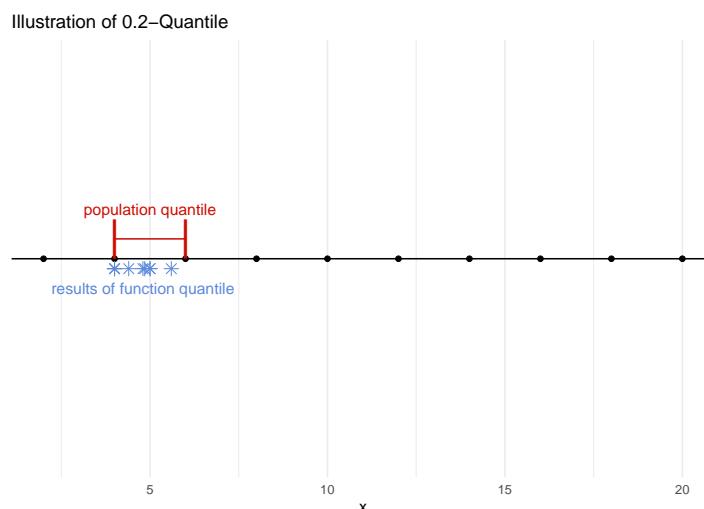
```
20%
4
```

```
1 ## Von SPSS und Minitab Software verwendet
2 quantile(x, type = 6, probs = 0.2)
```

```
20%
4.4
```

Wir stellen die Ergebnisse mittels `illustrate.quantile` aus dem Paket "MKdescr" (Kohl (2022a)) graphisch dar.

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



Anmerkung:

Wie das vorangegangene Beispiel 2.5 zeigt, müssen Sie damit rechnen, dass im Fall von Quantilen verschiedene Softwareprogramme unterschiedliche Ergebnisse liefern.

Wir kommen zurück zu unserem ITS-Datensatz. Der medizinische Score SAPS II ist ein typisches Beispiel für ein ordinales Merkmal. Wir berechnen zuerst den Median der Werte mit Hilfe der Funktion `median`.

```
1 median(ITSdaten$SAPS.II)
```

```
[1] 42
```

```
1 ## alternativ auch möglich
2 quantile(ITSdaten$SAPS.II, probs = 0.5)
```

```
50%
42
```

Entsprechend besitzen demnach 50% der Patienten einen SAPS II Score ≤ 42 und entsprechend 50% der Patienten einen Score ≥ 42 . Der Median ist ein sogenannter **Lageparameter**. Er beschreibt demnach, wo unsere Daten liegen. Er gibt uns aber keine Information darüber, wie stark die Werte um ihn

herum schwanken. Auch hierfür kann man Quantile verwenden. Ein sehr häufig verwendeter sogennanter **Streuungsparameter** ist der **Interquartilsabstand**, der Abstand zwischen dem dritten und ersten Quartil (d.h., $q_{0.75} - q_{0.25}$). In R können wir diesen mit Hilfe der Funktion `IQR` berechnen.

```
1 ## Intervalllänge
2 IQR(ITSDaten$SAPS.II)
```

```
[1] 26
```

```
1 ## Bestimmung des IQR-Intervalls
2 quantile(ITSDaten$SAPS.II, probs = c(0.25, 0.75))
```

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

Somit weisen also die mittleren 50% unserer Patienten eine Streubreite von 26 SAPS II Punkten auf. Eine weitere Möglichkeit die Streuung der Werte anzugeben, ist der **Median der absoluten Abweichungen vom Median** (kurz: **MAD**)

$$\text{MAD}(x_1, x_2, \dots, x_n) = \text{median}\{|x_1 - M|, |x_2 - M|, \dots, |x_n - M|\} \quad (2.2)$$

wobei $M = \text{median}\{x_1, x_2, \dots, x_n\}$. Wir erhalten mit Hilfe der Funktion `mad`

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

```
[1] 13
```

Wir müssen hier zusätzlich den Parameter `constant` setzen, da ansonsten eine standardisierte Version des MAD berechnet wird.

```
1 mad(ITSDaten$SAPS.II)
```

```
[1] 19.2738
```

Die standardisierte Version des MAD lautet:

$$\text{MAD}(x_1, x_2, \dots, x_n) = 1.4826 \cdot \text{median}\{|x_1 - M|, |x_2 - M|, \dots, |x_n - M|\} \quad (2.3)$$

Der Grund für diese Umstandardisierung ist, dass der MAD damit unter gewissen Voraussetzungen (normalverteilte Daten) vergleichbar zur Standardabweichung wird, welche wir in Abschnitt 2.6 kennenlernen werden. Auch im Fall des Interquartilsabstandes ist eine entsprechende Standardisierung möglich, die bei normalverteilten Daten zu einem Wert führt, der mit der Standardabweichung verglichen werden kann.

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

Berechnen kann man den standardisierten Interquartilsabstand etwa durch die Funktion `sIQR` aus dem Paket "MKdescr" (Kohl (2022a)).

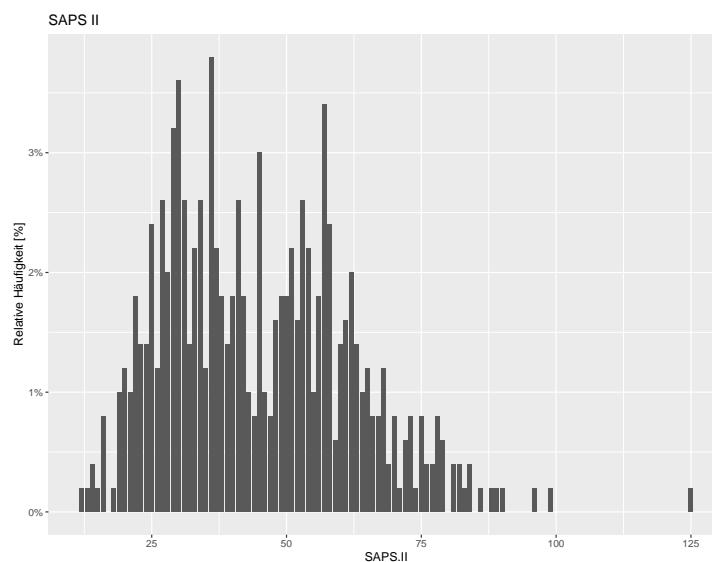
```
1 sIQR(ITSData$SAPS.II)
```

```
[1] 19.27383
```

Das Ergebnis ist identisch zum standardisierten MAD. Dies liegt daran, dass im vorliegenden Fall der unstandardisierte MAD gerade halb so groß ist wie der Interquartilsabstand, aber umgekehrt die Standardisierungskonstante des MAD gerade doppelt so groß ist wie die Standardisierungskonstante des Interquartilsabstandes.

Für die graphische Darstellung von ordinalen Daten können wir wieder Balkendiagramme verwenden.

```
1 ## Anlegen der Daten
2 ggplot(ITSData, aes(x=SAPS.II)) +
  ## Hinzufügen der Balken
4 geom_bar(aes(y = after_stat(count)/sum(after_stat(count)))) +
  ## Relative Häufigkeiten in Prozent
6 scale_y_continuous(labels = percent_format(accuracy = 1)) +
  ## Titel und Beschriftung der y-Achse
8 ggttitle("SAPS II") + ylab("Relative Häufigkeit [%]")
```

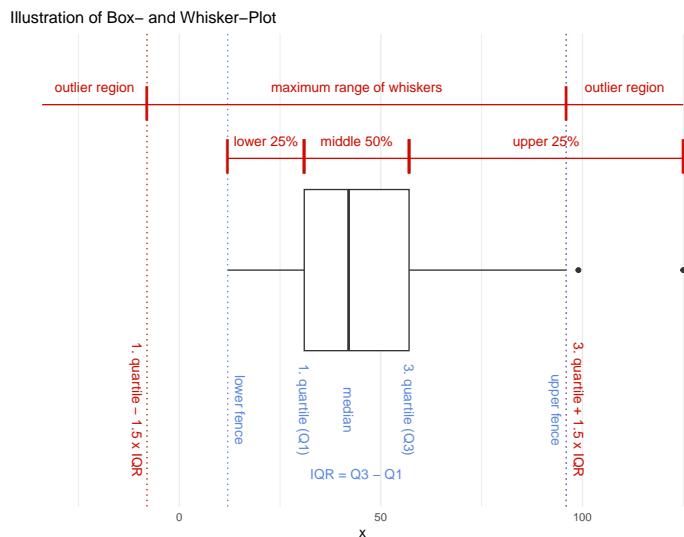


Aufgrund der großen Anzahl von möglichen Merkmalsausprägungen erinnert das Balkendiagramm bereits recht stark an ein Histogramm, welches wir in Abschnitt 2.6.1 kennenlernen werden.

Mit Hilfe der Quantile ist noch eine weitere in der Praxis sehr wichtige graphische Darstellung möglich, der sogenannte **Box-und-Whisker Plot** (kurz: Boxplot). Der Boxplot fasst auf sehr schöne Weise die Information von Median, IQR und dem Wertebereich für die Beobachtungen zusammen. Zusätzlich kann er dazu verwendet werden, auffällige Beobachtungen (Ausreißer) zu identifizieren. Eine detailliertere Beschreibung finden Sie in Kohl and Münch (2022a).

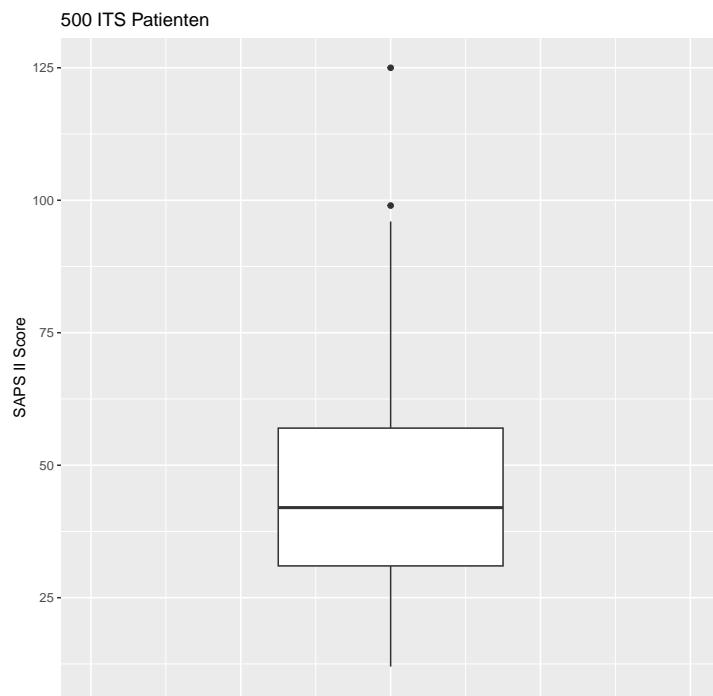
Wir erstellen mit Hilfe der Funktion `illustrate.boxplot` aus dem Paket "MKdescr" (Kohl (2022a)) einen Box-und-Whisker Plot für die SAPS II Werte, um die Definition des Boxplots anhand realer Daten zu illustrieren.

```
1 illustrate.boxplot(ITSDataen$SAPS.II)
```



Wie wir bereits wissen, liegt der Median bei 42. Die Box des Box-and-Whisker Plot beschreibt die mittleren 50% der Beobachtungen, die im Intervall [31, 57] liegen, dessen Länge dem bereits berechneten IQR von 26 Punkten entspricht. Es liegen 25% der Daten unterhalb von 31 und entsprechend 25% der Daten oberhalb von 57. Zwei der Patienten waren offenbar sehr krank mit Scorewerten von 99 und 125. Sie werden im Plot als Ausreißer angezeigt. Die Überlebenswahrscheinlichkeit dieser beiden Patienten war entsprechend klein und es überrascht nicht, dass beide Patienten verstorben sind. Von den zehn Patienten mit den höchsten SAPS II Scorewerten (≥ 83) sind neun verstorben. Wir wiederholen den Plot mit Hilfe der Funktion `geom_boxplot` aus dem Paket "ggplot2" (Wickham (2009)).

```
1 ggplot(ITSDataen, aes(x = 1, y = SAPS.II)) +
2   geom_boxplot() + xlim(0, 2) + ylab("SAPS II Score") +
3   ggtitle("500 ITS Patienten") +
4   ## keine Achsenbeschriftung für die x-Achse
5   xlab("") + theme(axis.ticks.x = element_blank(),
6                     axis.text.x = element_blank())
```

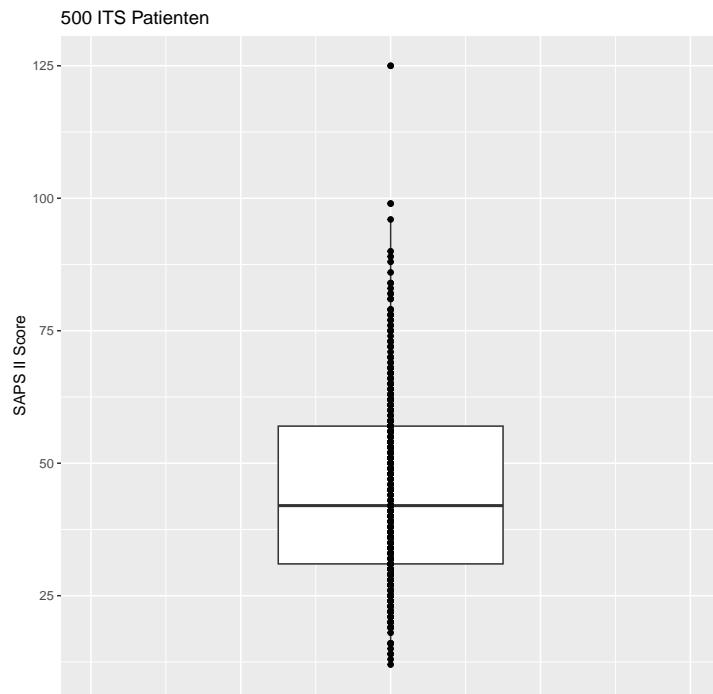


Damit die Box nicht zu breit erscheint, verwenden wir zusätzlich die Funktion `xlim`, mit der wir den Bereich der x-Achse vorgeben können. Die angegeben Zahlenwerte markieren Beginn und Ende der dargestellten x-Achse. Da die x-Achse in diesem Beispiel keine weitere Information transportiert und eventuelle automatisch erzeugte Angaben nur irreführend sein könnten, entfernen wir alle Einträge für diese Achse mit Hilfe der Funktionen `xlab` und `theme`. Gerade beim Vorliegen weniger Datenpunkte, ist es zudem zu empfehlen, auch die beobachteten Werte im Boxplot mit darzustellen. Hierzu kann man etwa zusätzlich die Funktion `geom_point` verwenden.

```

1 ggplot(ITSData, aes(x = 1, y = SAPS.II)) +
2   geom_boxplot() + xlim(0, 2) + ylab("SAPS II Score") +
3   geom_point() + ggtitle("500 ITS Patienten") +
4   ## keine Achsenbeschriftung für die x-Achse
5   xlab("") + theme(axis.ticks.x = element_blank(),
6                     axis.text.x = element_blank())

```

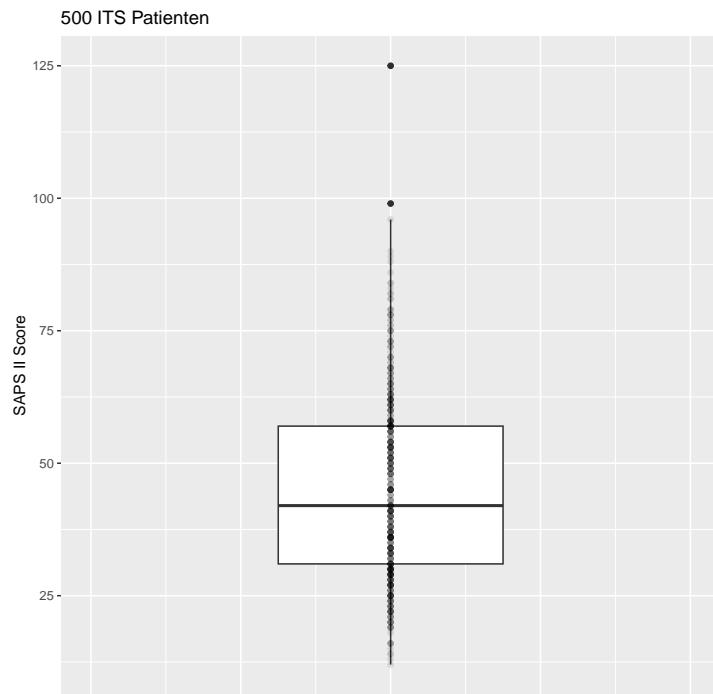


Der Nachteil ist, dass hierbei Werte übereinander zu liegen kommen. Ein möglicher Ausweg ist das sogenannte **Alpha Blending**. Hier werden die Punkte ebenfalls übereinander gezeichnet. Die Anzahl der Punkte wird durch die Farbintensität dargestellt; d.h., je dunkler ein Punkt in unserem Beispiel ist, desto mehr Punkte liegen dort übereinander.

```

1 ggplot(ITSDaten, aes(x = 1, y = SAPS.II)) +
2   geom_boxplot() + xlim(0, 2) + ylab("SAPS II Score") +
3   geom_point(alpha = 0.1) + ggtitle("500 ITS Patienten") +
4   ## keine Achsenbeschriftung für die x-Achse
5   xlab("") + theme(axis.ticks.x = element_blank(),
6                     axis.text.x = element_blank())

```

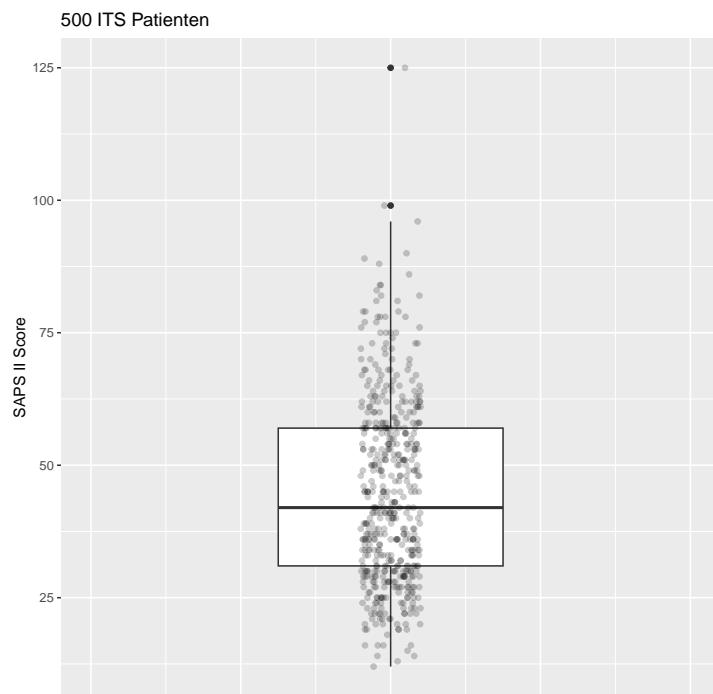


Eine andere Möglichkeit stellt die Verwendung von **Jittering** dar. In diesem Fall werden alle Punkte zufällig in x- und/oder y-Richtung verwackelt. In unserem Fall lassen wir nur ein Verwackeln in x-Richtung zu, da ein Verwackeln in y-Richtung die vorliegenden Werte verfälschen würde. Implementiert ist dies in der Funktion `geom_jitter`.

```

1 ggplot(ITSData, aes(x = 1, y = SAPS.II)) +
2   geom_boxplot() + xlim(0, 2) + ylab("SAPS II Score") +
3   geom_jitter(height = 0, width = 0.1, alpha = 0.2) +
4   ggtitle("500 ITS Patienten") +
5   ## keine Achsenbeschriftung für die x-Achse
6   xlab("") + theme(axis.ticks.x = element_blank(),
7                     axis.text.x = element_blank())

```



Zusammenfassend kann man sagen, dass der Box- und Whisker-Plot eine der wichtigsten Graphiken der deskriptiven Statistik darstellt und eigentlich jeder deskriptiven Analyse von ordinalen oder quantitativen Daten herangezogen werden sollte.

Eine weitere interessante Eigenschaft des α -Quantils ist, dass es auch dann noch verlässliche Informationen liefert, wenn die Daten einen gewissen Anteil an Ausreißern bzw. allgemeiner fehlerhaften Werten enthalten. Genauer dürfen die Daten bis zu $\alpha\%$ fehlerhafte Werte enthalten für $\alpha \in (0, 0.5]$ bzw. $1 - \alpha\%$ fehlerhafte Werte für $\alpha \in [0.5, 1)$. Dies macht insbesondere den Median sehr interessant, der in diesem Sinn die größtmögliche Robustheit besitzt.

Beispiel 2.6. Wir betrachten wieder die Zahlenfolge $2, 4, 6, \dots, 20$. Wir berechnen Median und drittes Quartil sowie 90% und 95% Quantil.

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

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

Nun vergrößern wir den größten Zahlenwert von 20 auf 200, was in diesem Fall 10% Ausreißern entspricht. Wir erhalten

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

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

Wir sehen, dass sowohl das 95%- als auch das 90%-Quantil davon beeinflusst werden und deutlich nach oben ausgelenkt werden. Bei Median und drittem Quartil hingegen führt dies zu keiner Veränderung.

Abschließend wollen wir uns noch mit einer weiteren Möglichkeit der graphischen Darstellung der Verteilung der beobachteten Werte befassen. Es ist dies die sogenannte empirische Verteilungsfunktion.

Definition 2.7 (Empirische Verteilungsfunktion). *Gegeben seien Beobachtungen $x_1, x_2, \dots, x_n \in \mathbb{R}$ ($n \in \mathbb{N}$) und wir bezeichnen mit $x_{(1)}, x_{(2)}, \dots, x_{(n)}$ die aufsteigend sortierten Beobachtungen. Weiter seien $h_{(1)}, h_{(2)}, \dots, h_{(n)}$ die zugehörigen relativen Häufigkeiten. Dann ist die **empirische Verteilungsfunktion** gegeben durch*

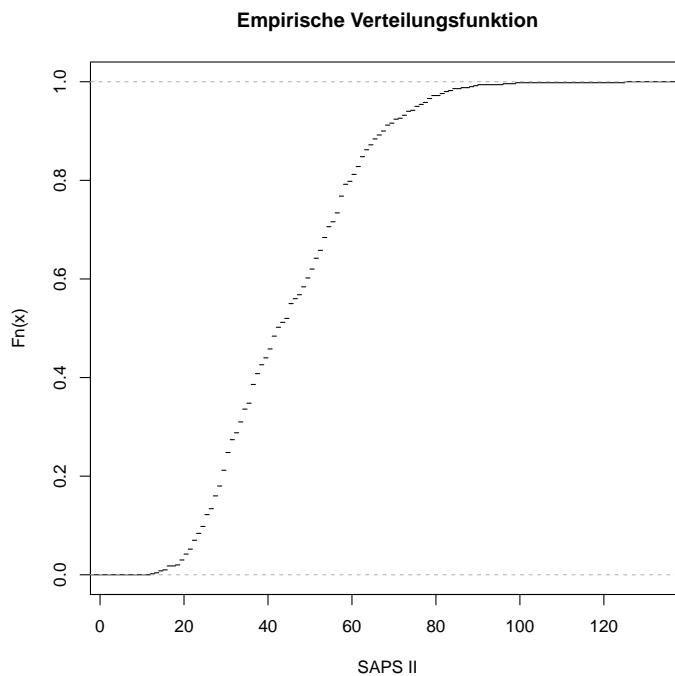
$$\hat{F}_n(x) = \begin{cases} 0 & \text{falls } x < x_{(1)} \\ \sum_{i=1}^k h_{(i)} & \text{falls } x_{(k)} \leq x < x_{(k+1)} \\ 1 & \text{falls } x > x_{(n)} \end{cases} \quad (2.5)$$

Wir geben einige zusätzliche Erläuterungen.

Bemerkung 2.8. *Die empirische Verteilungsfunktion ist demnach eine monoton wachsende und rechtsseitig stetige Treppenfunktion.*

Wir verwenden die Funktionen `ecdf` und `plot`, um die empirische Verteilungsfunktion der SAPS II Werte zu berechnen und graphisch darzustellen.

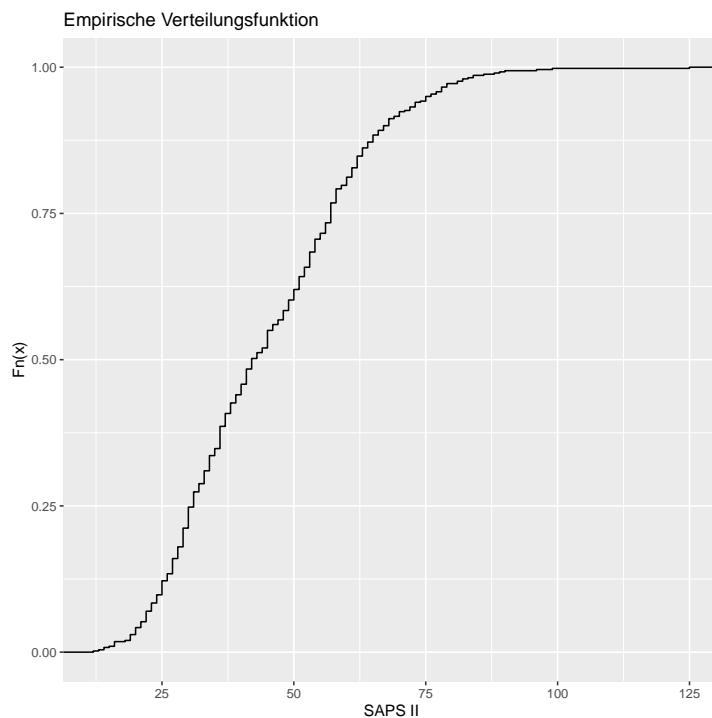
```
1 plot(ecdf(ITSData$SAPS.II), main = "Empirische Verteilungsfunktion",
2      xlab = "SAPS II", do.points = FALSE)
```



Aufgrund der feinen Unterteilung und der vielen kleinen Sprünge, verzichten wir auf das Einzeichnen von Punkten (`do.points = FALSE`). Die Punkte sollen verdeutlichen, dass es sich um eine rechtssei-

tige stetige Funktion handelt. Wir können eine entsprechende Abbildung auch mit Hilfe der Funktion `stat_ecdf` aus dem Paket "ggplot2" (Wickham (2009)) erzeugen.

```
1 ggplot(ITSDataen, aes(x = SAPS.II)) + stat_ecdf() + xlab("SAPS II") +
2   ylab("Fn(x)") + ggtitle("Empirische Verteilungsfunktion")
```



In der Praxis wird dieser Plot aber eher selten verwendet, da die Interpretation einige Erfahrung benötigt und für Anwender eher schwierig ist.

2.5.2 Bivariate Analyse

Bisher haben wir die Variablen nur einzeln behandelt. Jetzt wollen wir zwei Variablen in Beziehung zueinander setzen. Wir beginnen wieder mit nominalen Variablen. In diesem Fall besteht die Analyse in der Berechnung und Darstellung von absoluten oder relativen Häufigkeiten der möglichen Wertekombinationen. Dies führt uns auf eine **Kontingenztafel** oder **Kreuztabelle**. Wir setzen die Variablen Geschlecht und OP aus dem ITS-Datensatz zueinander in Beziehung. Die absoluten Häufigkeiten der Wertekombinationen erhalten wir mit Hilfe der Funktion `table`.

```
1 table(ITSDataen$Geschlecht, ITSDataen$OP)
```

	Andere	Herz-Thorax	Magen-Darm-Trakt	Neuro	Trauma
männlich	64	162	48	27	24
weiblich	57	61	31	19	7

Die absoluten Zahlen legen nahe, dass es deutlich mehr Herz-Thorax Operationen bei den Männern als bei den Frauen gab. Da der Datensatz aber auch deutlich mehr Männer als Frauen enthält, sollten wir

diesen Eindruck durch die Betrachtung von relativen Häufigkeiten absichern. Für die Berechnung der relativen Häufigkeiten können wir die Funktion `proportions` auf die Kreuztabelle anwenden. Durch Angabe des Arguments `margin` kann man steuern, ob die relativen Häufigkeiten zeilen- (`margin = 1`) oder spaltenweise (`margin = 2`) berechnet werden sollen. In unserem Beispiel soll dies zeilenweise geschehen.

```
1 proportions(table(ITSDaten$Geschlecht, ITSDaten$OP), margin = 1)
```

	Andere	Herz-Thorax	Magen-Darm-Trakt	Neuro	Trauma
männlich	0.19692308	0.49846154	0.14769231	0.08307692	0.07384615
weiblich	0.32571429	0.34857143	0.17714286	0.10857143	0.04000000

Um eine schönere und übersichtlichere Darstellung zu bekommen, können wir die obigen Ergebnisse noch in Prozent umrechnen und mit Hilfe der Funktion `round` auf eine Nachkommastelle runden.

```
1 round(100*proportions(table(ITSDaten$Geschlecht, ITSDaten$OP), margin = 1), 1)
```

	Andere	Herz-Thorax	Magen-Darm-Trakt	Neuro	Trauma
männlich	19.7	49.8	14.8	8.3	7.4
weiblich	32.6	34.9	17.7	10.9	4.0

Eine Darstellung von absoluten und relativen Häufigkeiten in einer Tabelle ist auch mit der Funktion `PercTable` aus dem Paket "DescTools" (Andri et mult. al. (2022)) möglich.

```
1 PercTable(table(ITSDaten$Geschlecht, ITSDaten$OP), rfrq = "010")
```

		Andere	Herz-Thorax	Magen-Darm-Trakt	Neuro	
männlich	freq	64	162	48	27	
	p.row	19.7%	49.8%	14.8%	8.3%	
weiblich	freq	57	61	31	19	
	p.row	32.6%	34.9%	17.7%	10.9%	
			Trauma			
männlich	freq	24				
	p.row	7.4%				
weiblich	freq	7				
	p.row	4.0%				

Der Parameter `rfrq` gibt dabei an, wie die relative Häufigkeit zu bilden ist. Der Eintrag "010" steht für zeilenweise relative Häufigkeiten; d.h., jede Zeile summiert sich zu 100%. Die Ergebnisse bestätigen

unseren ersten Eindruck, dass sich Männer deutlich häufiger einer Herz-Thorax Operation unterziehen mussten. Umgekehrt kommen bei Frauen "Andere" Operationen auffällig häufiger vor.

Die Stärke des Zusammenhangs zwischen zwei (oder mehr) nominalen (oder auch ordinalen) Variablen kann mit Hilfe sogenannter Kontingenzkoeffizienten bestimmen.

Definition 2.9 (Kontingenzkoeffizienten). *Gegeben seien $n \in \mathbb{N}$ Beobachtungen von zwei Variablen mit $l \in \mathbb{N}$ bzw. $m \in \mathbb{N}$ verschiedenen Merkmalsausprägungen. Die beobachteten Wertepaare können folglich in eine Matrix mit l Zeilen und m Spalten aufgeteilt werden, wobei $k = l \cdot m$ die Gesamtzahl der Zellen darstellt. Weiter seien n_i ($i = 1, \dots, k$) die Anzahl der Beobachtungen in Zelle i , p_i ($i = 1, \dots, k$) die theoretischen Wahrscheinlichkeiten für die Zellen und somit $e_i = N \cdot p_i$ ($i = 1, \dots, k$) die erwartete Anzahl von Beobachtungen in Zelle i . Dann ist die χ^2 -Statistik gegeben durch*

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

Auf Basis der χ^2 -Statistik erhalten wir die folgenden **Kontingenzkoeffizienten**

(i) **ϕ -Koeffizient**

$$\phi = \sqrt{\frac{\chi^2}{n}} \quad (2.7)$$

(ii) **Pearsons Kontingenzkoeffizient**

$$C = \sqrt{\frac{\chi^2}{n + \chi^2}} \quad (2.8)$$

(iii) **Cramérs V**

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

Wir geben einige zusätzliche Erklärungen zu Kontingenzkoeffizienten.

Bemerkung 2.10. (a) Bei der Verwendung von Kontingenzkoeffizienten, sollte man sich immer klar machen, welcher Maximalwert möglich ist. Zudem besitzen Kontingenzkoeffizienten den Nachteil, dass diese nur ein Maß für die Stärke, aber nicht die Richtung des Zusammenhangs liefern, was für ordinale Merkmale interessant ist.

(b) Der ϕ -Koeffizient liegt immer im Intervall $[0, 1]$, wobei 1 nur unter gewissen Gegebenheiten möglich ist. Der Wert 0 bedeutet, dass die beiden Merkmale unabhängig sind.

(c) Pearsons Kontingenzkoeffizient liegt im Intervall $[0, \sqrt{\frac{M}{M-1}}]$ ($M = \min\{l, m\}$), wobei 0 wieder die Unabhängigkeit der Merkmale anzeigen.

(d) Cramérs V nimmt Werte im Intervall $[0, 1]$ an, wobei erneut 0 für die Unabhängigkeit der Merkmale steht. Man spricht von schwacher Abhängigkeit für $V \leq 0.3$, von mittelgroßer Abhängigkeit für $0.3 < V \leq 0.7$ und von starker Abhängigkeit für $V > 0.7$.

Wir verwenden die Funktionen Phi (ϕ -Koeffizient), ContCoef (Pearsons Kontingenzkoeffizient) und CramerV (Cramérs V) aus dem Paket "DescTools" (Andri et mult. al. (2022)), um die Stärke des Zusammenhangs zwischen Geschlecht und OP zu ermitteln.

```
1 Phi(table(ITSDaten$Geschlecht, ITSDaten$OP))
```

```
[1] 0.1846974
```

```
1 ContCoef(table(ITSDaten$Geschlecht, ITSDaten$OP))
```

```
[1] 0.1816254
```

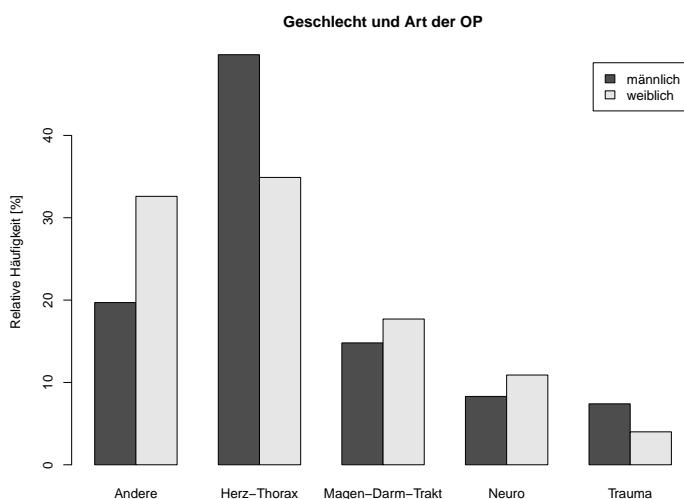
```
1 CramerV(table(ITSDaten$Geschlecht, ITSDaten$OP))
```

```
[1] 0.1846974
```

Wir erhalten lediglich einen schwachen Zusammenhang zwischen dem Geschlecht und der Art der Operation. Da $M = 2$ ist, fallen der ϕ -Koeffizient und Cramérs V zusammen.

Für die graphische Darstellung von Kontingenztafeln bieten sich Balkendiagramme an. Wir stellen die Variablen Geschlecht und OP zusammen dar, wobei wir die Funktion `barplot` in Kombination mit `table` und `proportions` verwenden.

```
1 barplot(round(100*proportions(table(ITSDaten$Geschlecht, ITSDaten$OP)),
2 margin = 1), 1),
3 beside = TRUE, legend.text = TRUE, ylab = "Relative Häufigkeit [%]",
4 main = "Geschlecht und Art der OP")
```



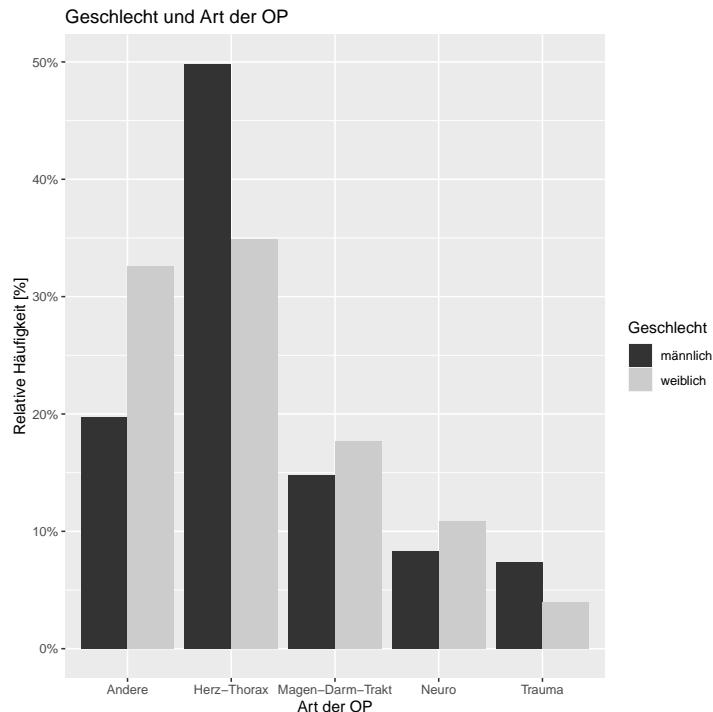
Mit dem Argument `beside = TRUE` sorgen wir dafür, dass die Balken für Männer und Frauen nebeneinander und nicht übereinander sind. Das Argument `legend.text = TRUE` sorgt für die Anzeige einer Legende, welche die Zuordnung der Farben zum Geschlecht erklärt. Wir wiederholen die Abbildung, wobei wir dieses Mal die Funktionen des Paketes "ggplot2" (Wickham (2009)) verwenden. Die relativen Häufigkeiten werden hierbei innerhalb der Funktion `geom_bar` mit Hilfe der Funktion `tapply` berechnet, was zugegebenermaßen durch einen etwas kryptischen Code passiert. Mit Angabe von `position =`

'dodge' als Position, werden die Balken nebeneinander (und nicht übereinander) dargestellt. Wir verwenden zusätzlich die Funktion `scale_fill_grey`, die für ein Füllen der Balken mit Grautönen sorgt.

```

1 ggplot(ITSData, aes(x = OP, fill = Geschlecht)) +
2   geom_bar(aes(y = after_stat(count) / tapply(after_stat(count),
3                           after_stat(fill),
4                           sum)[after_stat(fill)]),
5             position = "dodge") + scale_fill_grey() +
6   scale_y_continuous(labels = percent_format(accuracy = 1)) +
7   ylab("Relative Häufigkeit [%]") + xlab("Art der OP") +
8   gtitle("Geschlecht und Art der OP")

```



Mit Hilfe der Funktion `tapply` können wir die Werte einer Variable nach den Werten einer anderen Variable aufteilen und hierauf eine Funktion anwenden.

Im Fall ordinaler Variablen kann man anstelle von Kontingenzkoeffizienten auch Rangkorrelationen verwenden, welche nicht nur die Stärke, sondern auch die Richtung des Zusammenhangs anzeigen. Unter dem **Rang** einer Beobachtung versteht man deren Position innerhalb der Stichprobe, nachdem man die Beobachtungen absteigend sortiert hat; d.h., die größte Beobachtung hat Rang 1, die zweitgrößte Rang 2, etc.

Definition 2.11 (Spearmans ρ). *Gegeben seien Beobachtungspaare $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$ ($n \in \mathbb{N}$)*

mit Rängen $(rx_1, ry_1), (rx_2, ry_2), \dots, (rx_n, ry_n)$. Dann ist **Spearmans ρ** definiert durch

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

wobei mr_x und mr_y die jeweiligen mittleren Ränge sind; d.h.,

$$mr_x = \frac{1}{n} \sum_{i=1}^n rx_i \quad \text{und} \quad mr_y = \frac{1}{n} \sum_{i=1}^n ry_i \quad (2.11)$$

Spearmans ρ kann Werte im Intervall $[-1, 1]$ annehmen, wobei 1 für einen streng monoton wachsenden und -1 für einen streng monoton fallenden Zusammenhang steht.

Wir geben einige zusätzliche Erläuterungen zu Rangkorrelationen.

Bemerkung 2.12. (a) Im Fall, dass ein Wert mehrfach (mindestens zweimal) beobachtet wurde, spricht man von einer **Bindung**. Liegen keine Bindungen vor, so vereinfacht sich die Berechnung von Spearmans ρ und es gilt

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

(b) Neben Spearmans ρ wird häufig noch **Kendalls τ** als Rangkorrelationskoeffizient verwendet, bei dem man Anzahl der gleichsinnigen (konkordant) und gegensinnigen (diskonkordant) Wertepaare miteinander vergleicht. Auch hier ergeben sich Werte im Intervall $[-1, 1]$. Der Wert 1 bedeutet, dass beide Variablen die exakt gleiche Anordnung besitzen und -1, dass diese perfekt umgekehrt angeordnet sind. Kendalls τ ist vor allem bei kleinen Stichproben oder bei Scores mit einer ungleichen Einteilung Spearmans ρ vorzuziehen. Für weitere Einzelheiten verweisen wir auf Abschnitt 3.2.5 von Hedderich and Sachs (2018).

(c) Rangkorrelationen sind auch im Fall von metrischen Variablen hilfreich, um monotone Zusammenhänge zu identifizieren.

Wir berechnen die Korrelation zwischen SAPS II und der Aufenthaltsdauer (LOS), wobei wir hierfür die Funktion `cor` verwenden.

```
1 ## Spearmans rho
2 cor(ITSDaten$SAPS.II, ITSDaten$LOS, method = "spearman")
```

```
[1] 0.3379928
```

```
1 ## Kendalls tau
2 cor(ITSDaten$SAPS.II, ITSDaten$LOS, method = "kendall")
```

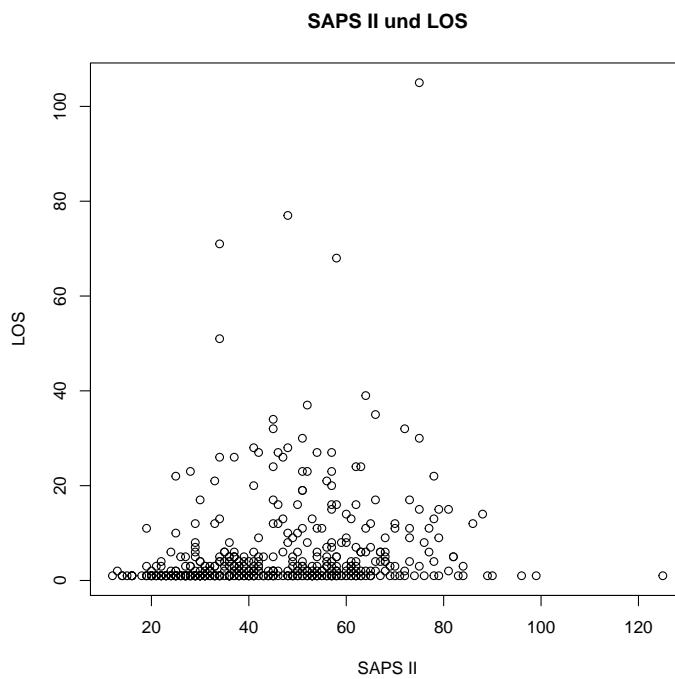
```
[1] 0.2518917
```

Anmerkung:

Die obigen Zahlen beschreiben nur die Stärke eines monotonen Zusammenhangs, falls der Zusammenhang auch wirklich monoton ist. Dies ist eine Voraussetzung, über die man sich Gedanken machen sollte, bevor man die Berechnung durchführt. Generell sollte man sich bewusst sein, dass die meisten statistischen Analysen auf gewissen Annahmen basieren. Sind diese Voraussetzungen verletzt, so ist auch die Bedeutung der Ergebnisse unklar. Liegt etwa kein einfacher monotoner Zusammenhang zwischen SAPS II und der Aufenthaltsdauer (LOS) vor, so können die obigen Ergebnisse von Spearmans ρ oder Kendalls τ zu Fehlinterpretationen führen. Die Überprüfung der Voraussetzungen erfordert in der Regel eine gute Fachkenntnis über die untersuchten Zusammenhänge.

Zunächst einmal könnte man erwarten, dass ein monotoner Zusammenhang zwischen SAPS II und der Aufenthaltsdauer (LOS) besteht. Patienten mit einem höheren SAPS II Wert, sind schwerer krank und sind daher länger auf der ITS. Dem entgegen wirkt jedoch die Tatsache, dass Patienten mit einem sehr hohen SAPS II Wert auch eine höhere Sterbewahrscheinlichkeit haben und somit unter Umständen kurz nach der Aufnahme auf der ITS versterben. Um zu überprüfen, ob dies tatsächlich zutrifft, stellen wir die beobachteten Werte in einem **Streudiagramm** (scatter plot) dar. Wir verwenden hierfür zunächst die Funktion `plot`.

```
1 plot(ITSDaten$SAPS.II, ITSDaten$LOS, xlab = "SAPS II", ylab = "LOS",
2      main = "SAPS II und LOS")
```



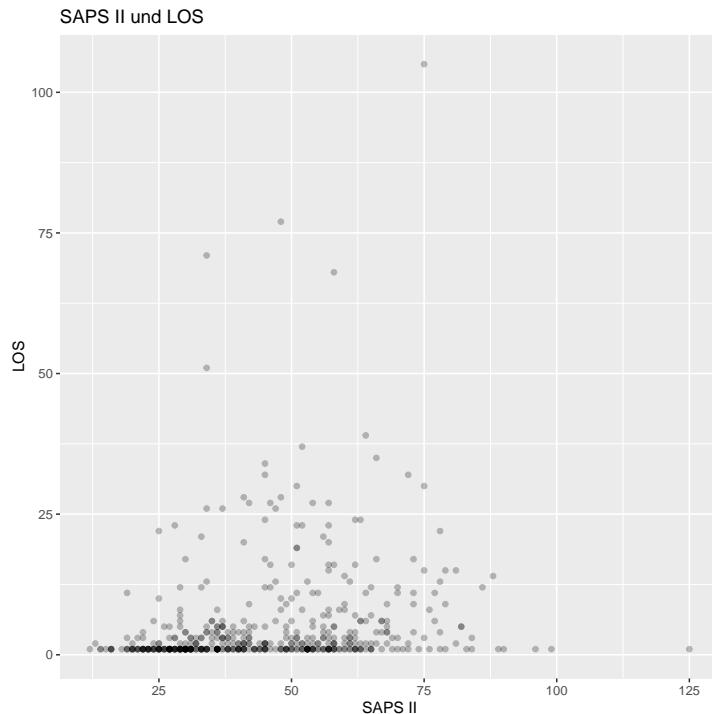
In der Tat waren die Patienten mit den höchsten SAPS II Werten nur sehr kurz auf der ITS und sind recht rasch verstorben. Durch die ordinale bzw. diskrete Struktur der betrachteten Variablen, kommt es zu einer Überlagerung der Beobachtungen. Um die Struktur der Punktewolke besser sichtbar zu machen, verwenden wir wieder das sogenannte **Alpha Blending**; d.h., die Endfarbe entsteht durch eine Überlagerung der Ausgangsfarben. Wir verwenden die Funktionen des Paketes "ggplot2" (Wickham (2009)).

Die möglichen Formen von Punkten, die in R standardmäßig zur Verfügung steht, ist in der Hilfeseite der Funktion `points` beschrieben.

```

1 ggplot(ITSDaten, aes(x=SAPS.II, y=LOS)) +
2   ## shape = 19: etwas größerer Punkt
3   ## alpha = 0.25: Stärke des Blendings
4   geom_point(shape=19, alpha=0.25) +
5   ## Beschriftung
6   ggttitle("SAPS II und LOS") + xlab("SAPS II") + ylab("LOS")

```



Je dunkler die Punkte sind, desto mehr Punkte überlappen dort. Für einen eingeschränkten Bereich des SAPS II Scores können wir somit vermutlich von einem monoton wachsenden Zusammenhang ausgehen, aber sicher nicht für den gesamten Wertebereich. Die Werte der Rangkorrelationskoeffizienten sollten daher im vorliegenden Fall mit großer Vorsicht interpretiert werden. Besser wäre es, den Zusammenhang auf andere Weise zu untersuchen, nämlich mit Hilfe einer zur Fragestellung passenden Regressionsanalyse. Dies geht jedoch über die Themen dieses Buches hinaus.

2.6 Metrische Variablen

2.6.1 Univariate Analyse

Im Fall metrischer Variablen kommen nun weitere Möglichkeiten der Auswertung hinzu, da Abstände und im Fall von ratio-skalierten Variablen auch Verhältnisse definiert sind. Falls nicht explizit darauf hingewiesen wird, sind die vorgestellten Analysen sowohl für intervall- also auch für ratioskalierte Variablen möglich. Die wohl am häufigsten verwendete Statistik, um Daten zu beschreiben, ist das arithmetische Mittel.

Definition 2.13 (Arithmetisches Mittel). *Gegeben seien Beobachtungen $x_1, x_2, \dots, x_n \in \mathbb{R}$ ($n \in \mathbb{N}$). Dann ist das **arithmetische Mittel** definiert als*

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

Wir verwenden auch in diesem Abschnitt wieder die ITS Daten (siehe Abschnitt 2.3). Wir berechnen mit Hilfe der Funktion `mean` das arithmetische Mittel der maximalen Körpertemperaturen während des Aufenthalts auf der ITS.

```
1 mean(ITSDataen$Temperatur)
```

```
[1] 37.6632
```

Der Wert liegt demnach nur ganz wenig über dem Bereich der Normaltemperatur, wobei das Ergebnis eine Genauigkeit suggeriert, die nicht wirklich vorliegt. Die Temperaturwerte sind nur mit einer Nachkommastelle angegeben. Entsprechend sollte auch das arithmetische Mittel besser auf eine Nachkommastelle gerundet werden. Wir können hierfür die Funktion `round` verwenden.

```
1 round(mean(ITSDataen$Temperatur), 1)
```

```
[1] 37.7
```

Es empfiehlt sich immer das arithmetische Mittel mit dem Median zu vergleichen, da der Median als 50% Quantil eine andere Beschreibung der Mitte liefert und zudem, wie wir im Beispiel 2.6 gesehen haben, sehr robust gegenüber Ausreißern ist.

```
1 median(ITSDataen$Temperatur)
```

```
[1] 37.7
```

Da Median und arithmetisches Mittel identisch sind, liegt es nahe, dass die Verteilung der Temperaturwerte recht symmetrisch um das arithmetische Mittel bzw. dem Median herum ist. Zudem kann man davon ausgehen, dass entweder keine Ausreißer vorkommen oder sich die positiven und negativen Ausreißer gerade aufheben. Wir wiederholen die Analyse mit der Variable LOS (Aufenthaltsdauer), die in Tagen angegeben ist.

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

```
[1] 5.3
```

```
1 median(ITSDataen$LOS)
```

```
[1] 1
```

In diesem Fall sehen wir einen deutlichen Unterschied zwischen arithmetischem Mittel und Median. Entweder ist die Verteilung der LOS-Werte schief (genauer rechtsschief, vgl. auch Bemerkung 2.23) oder aber es liegen einer oder mehrere Ausreißer vor, die das arithmetische Mittel nach rechts ziehen. Genauer werden wir dies im weiteren Verlauf dieses Abschnitts mit Hilfe von Diagrammen untersuchen.

Ein weiterer Lageparameter ist das geometrische Mittel, welches insbesondere bei relativen Änderungen verwendet wird. Dieses Lagemaß ist nur für positive Daten sinnvoll definiert.

Definition 2.14 (Geometrisches Mittel). *Gegeben seien Beobachtungen $x_1, x_2, \dots, x_n \in (0, \infty)$ ($n \in \mathbb{N}$). Dann ist das **geometrische Mittel** definiert als*

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

In der folgenden Bemerkung beschreiben wir einen wichtigen Zusammenhang zwischen dem geometrischen und dem arithmetischen Mittel.

Bemerkung 2.15. *Durch Anwendung der Rechenregeln für den Logarithmus erhalten wir.*

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

D.h., das arithmetische Mittel der logarithmierten Beobachtungen entspricht gerade dem Logarithmus des geometrischen Mittels der Beobachtungen. Die Basis des Logarithmus spielt dabei keine Rolle. Wählen wir den natürlichen Logarithmus (\ln), so können wir dies durch Anwendung der e -Funktion auch umschreiben in

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

Beobachtet man Vorgänge, die einem exponentiellem Wachstum bzw. Zerfall unterliegen, so ist es oftmals einfacher, die zugehörigen Daten zu logarithmieren und die logarithmierten Werte zu analysieren. Dies gilt zum Beispiel auch für die Bilirubinwerte, welche in unserem ITS-Datensatz enthalten sind. Die Standardpakete von R enthalten leider keine Implementation des geometrischen Mittel. Wir verwenden daher die Funktion `Gmean` aus dem Paket "DescTools" (Andri et mult. al. (2022)), um das geometrische Mittel zu berechnen und nehmen hiervon den natürlichen Logarithmus (Funktion `log`).

```
1 log(Gmean(ITSData$Bilirubin))
```

```
[1] 2.847326
```

Wie unsere Rechnung in Bemerkung 2.15 zeigt, muss der folgende Code das gleiche Ergebnis liefern, was auch tatsächlich der Fall ist.

```
1 mean(log(ITSData$Bilirubin))
```

```
[1] 2.847326
```

Entsprechend erhalten wir das geometrische Mittel nicht nur mit Hilfe von Funktion `Gmean`, sondern auch durch

```
1 exp(mean(log(ITSDaten$Bilirubin)))
```

```
[1] 17.24162
```

wobei `exp` die e-Funktion berechnet. Diese Form der Berechnung hat sogar numerische Vorteile, da die Summation numerisch stabiler ist als die Berechnung von Produkten. Entsprechend wird das geometrische Mittel üblicherweise auch auf diese Weise implementiert.

In der Praxis ist aber nicht nur die Lage der Werte von Interesse, sondern auch deren Streuung. Das wohl am häufigsten verwendete Streuungsmaß ist die Standardabweichung, die sich also Quadratwurzel der Varianz ergibt.

Definition 2.16 (Varianz, Standardabweichung). *Gegeben seien Beobachtungen $x_1, x_2, \dots, x_n \in \mathbb{R}$ ($n \in \mathbb{N}$). Dann ist die Stichproben-Varianz definiert als*

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

und die Stichproben-Standardabweichung lautet

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

Die folgende Bemerkung enthält einige zusätzliche Erklärungen.

Bemerkung 2.17. (a) Anstelle von $\frac{1}{n}$ wird bei Varianz und Standardabweichung häufig $\frac{1}{n-1}$ verwendet. Man kann den Unterschied zwischen deskriptiver und induktiver Statistik an diesem kleinen Unterschied festmachen. Mit der Standardisierung $\frac{1}{n}$ beschreiben wir die Stichprobe, hingegen liefert die Standardisierung $\frac{1}{n-1}$ eine unverzerrte Parameterschätzung für die zugrunde liegende Population. Genauer werden wir dies in Beispiel 5.3 behandeln. Bei nicht zu kleinem n ist der Unterschied in der Praxis vernachlässigbar.

(b) Angenommen die vorliegenden Beobachtungen wurden in der Einheit E gemessen, so besitzt die Varianz die Einheit E^2 und folglich die Standardabweichung die Einheit E . Dies dürfte ein wichtiger Grund dafür, warum in der Praxis die Standardabweichung deutlich häufiger verwendet wird als die Varianz, um die Streuung von Werten zu beschreiben.

Wir berechnen Varianz und Standardabweichung für die maximale Körpertemperatur. Die entsprechenden Funktionen in R heißen `var` und `sd`.

```
1 var(ITSDaten$Temperatur)
```

```
[1] 3.011869
```

```
1 sd(ITSDaten$Temperatur)
```

```
[1] 1.735474
```

Die beiden Funktionen verwenden die Standardisierung $\frac{1}{n-1}$. Indem wir das Ergebnis mit $\frac{n-1}{n}$ multiplizieren, erhalten wir die “echten” Stichprobenwerte.

```
1 n ← nrow(ITSDaten)
2 (n-1) / n * var(ITSDaten$Temperatur)
```

```
[1] 3.005846
```

```
1 sqrt((n-1)/n)*sd(ITSDaten$Temperatur)
```

```
[1] 1.733738
```

Würden wir die Ergebnisse noch zusätzlich auf eine Nachkommastelle runden, was aufgrund der vorgegebenen Genauigkeit der Temperaturwerte zu empfehlen ist, so erhielten wir identische Ergebnisse. In ähnlicher Weise wie wir arithmetisches Mittel und Median verglichen haben, vergleichen wir jetzt die Standardabweichung mit dem standardisierten MAD und dem standardisierten Interquartilsabstand. (vgl. Gleichungen (2.3) und (2.4)).

```
1 sd(ITSDaten$Temperatur)
```

```
[1] 1.735474
```

```
1 mad(ITSDaten$Temperatur)
```

```
[1] 1.18608
```

```
1 sIQR(ITSDaten$Temperatur)
```

```
[1] 1.111952
```

Wir erhalten erkennbare Unterschiede zwischen den drei Statistiken. Entweder kann die Temperaturverteilung in unserer Patientenpopulation eher nicht durch eine Verteilung, die symmetrisch um das arithmetische Mittel ist, beschrieben werden oder das Ergebnis für die Standardabweichung wurde durch Ausreißer ausgelenkt. Wir werden die Ursache in Kürze noch näher untersuchen.

In der Praxis verwendet man bei positiven Messwerten auch häufig das folgende standardisierte Streuungsmaß.

Definition 2.18 (Variationskoeffizient). *Gegeben seien Beobachtungen $x_1, x_2, \dots, x_n \in [0, \infty)$ ($n \in \mathbb{N}$). Dann ist der Variationskoeffizient definiert als*

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

Wir geben wieder einige zusätzliche Erklärungen hierzu.

Bemerkung 2.19. (a) Der Variationskoeffizient ist eine dimensionslose Größe, die häufig in Prozent angegeben wird; d.h., als prozentuale Streuung bezogen auf das arithmetische Mittel. Folglich sollte diese Statistik nur für ratio-skalierte Variablen verwendet werden. Im Englischen verwendet man üblicherweise die Abkürzung **CV** (coefficient of variation), die auch im Deutschen gebräuchlich ist.

(b) Mögliche Alternativen des Variationskoeffizienten basieren auf Quantilen. Eine Möglichkeit besteht in der Verwendung von Median und MAD

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

Alternativ kann man auch die Quartile heranziehen. Man nennt dies dann auch den **Quartilsdispersionskoeffizienten**

$$iqrCV(x_1, \dots, x_n) = \frac{IQR(x_1, \dots, x_n)}{\text{median}(x_1, \dots, x_n)} \quad (2.21)$$

Indem wir sowohl im Fall des MAD als auch des Interquartilsabstandes die standardisierte Version verwenden, sollten die drei Varianten bei (näherungsweise) normalverteilten Daten recht ähnliche Ergebnisse liefern.

Wir wenden die standardisierten Streuungsmaße auf die maximale Körpertemperatur an und verwenden hierfür die Funktionen `CV`, `medCV` und `iqrCV` aus dem Paket "MKdescr" (Kohl (2022a)).

```
1 CV(ITSDaten$Temperatur)
```

```
[1] 0.04607877
```

```
1 medCV(ITSDaten$Temperatur)
```

```
[1] 0.03146106
```

```
1 iqrCV(ITSDaten$Temperatur)
```

```
[1] 0.02949474
```

Wir beobachten demnach eher kleine Schwankungen um das arithmetische Mittel bzw. den Median herum, die im Bereich von ca. 3-5% liegen, wobei die Werte von `medCV` und `iqrCV` recht ähnlich sind.

Im Folgenden definieren wir die Standardabweichung zum geometrischen Mittel.

Definition 2.20 (Geometrische Standardabweichung). *Gegeben seien Beobachtungen $x_1, x_2, \dots, x_n \in (0, \infty)$ ($n \in \mathbb{N}$). Dann ist die **geometrische Standardabweichung** definiert als*

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

Wir geben eine kurze Motivation für diese Definition.

Bemerkung 2.21. *Es gilt*

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

Indem wir den Zusammenhang (2.15) nutzen und in Analogie die geometrische Standardabweichung einführen, erhalten wir

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

Durch Anwendung der e -Funktion folgt hieraus die Definition 2.20. Den Ausdruck unter dem Summenzeichen könnte man noch umschreiben

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

Wir überprüfen den Zusammenhang (2.24) anhand der Bilirubinwerte im ITS-Datensatz, wobei wir die Funktion `Gsd` aus dem Paket "DescTools" (Andri et al. (2022)) für die Berechnung der geometrischen Standardabweichung verwenden.

```
1 log(Gsd(ITSDaten$Bilirubin))
```

```
[1] 0.7238379
```

```
1 sd(log(ITSDaten$Bilirubin))
```

```
[1] 0.7238379
```

Neben den Lage- und Streuungsmaßen verwendet man bei metrischen Variablen auch sogenannte **Formmaße**. Ein Formmaß für die Symmetrie ist die Schiefe.

Definition 2.22 (Schiefe). *Gegeben seien Beobachtungen $x_1, x_2, \dots, x_n \in \mathbb{R}$ ($n \in \mathbb{N}$). Dann ist die **Schiefe** definiert als*

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

*Für $Schiefe(x_1, \dots, x_n) < 0$ spricht man von **linksschief** oder **negativer Schiefe**, für $Schiefe(x_1, \dots, x_n) > 0$ von **rechtsschief** oder **positiver Schiefe**.*

Wir geben einige zusätzliche Erläuterungen.

Bemerkung 2.23. (a) Die in der Definition der Schiefe vorgenommene Zentrierung am arithmetischen Mittel und Standardisierung mittels der Standardabweichung macht die Schiefe unabhängig von der Maßeinheit der Variable. Man nennt dies auch **z-Transformation** und spricht daher auch von **z-Score**.

(b) Die Schiefe einer Verteilung kann auch anhand von arithmetischem Mittel und Median erkannt werden. Beobachtet man, dass $AM(x_1, \dots, x_n) < median(x_1, \dots, x_n)$, so spricht dies für eine linksschiefe Verteilung. Im Fall $AM(x_1, \dots, x_n) > median(x_1, \dots, x_n)$ kann man hingegen von einer rechtsschiefen Verteilung ausgehen; siehe auch Abbildung 2.8.

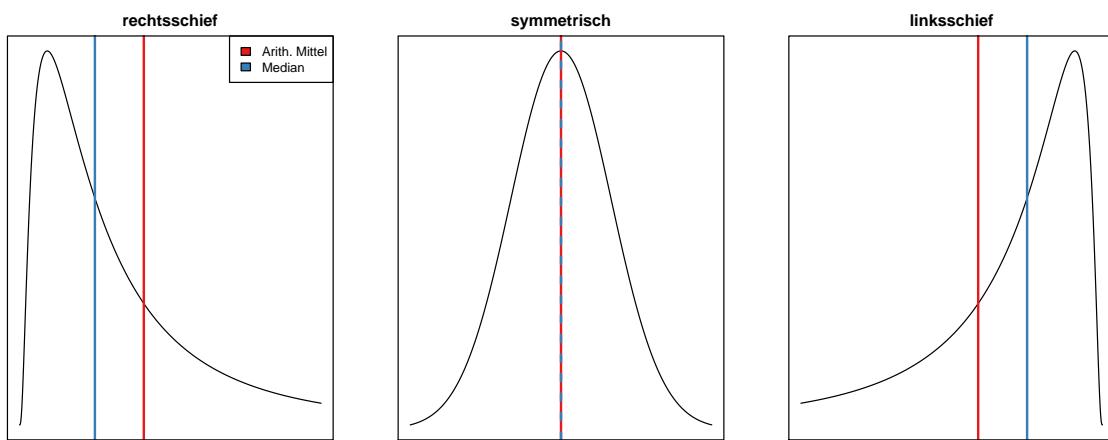


Abbildung 2.8: Beispiele für Schiefe.

Wir berechnen die Schiefe für die maximale Körpertemperatur und verwenden hierfür die Funktion Skew aus dem Paket "DescTools" (Andri et al. (2022)).

```
1 Skew(ITSDaten$Temperatur)
```

```
[1] -8.77457
```

Das Ergebnis, welches auf eine stark linksschiefe Verteilung der Temperaturmessungen hindeutet, widerspricht unserer Beobachtung von oben. Dort haben wir gesehen, dass Median und Mittelwert nahezu identisch sind und daher geschlossen, dass die Verteilung recht symmetrisch sein muss. Eine genauere Betrachtung der gemessenen Temperatur zeigt, dass Patient 398 eine ungewöhnlich niedrige maximale (!) Körpertemperatur von nur 9.1°C aufweist (Mess- bzw. Übertragungsfehler?). Wir wiederholen daher die Berechnung, wobei wir den Wert des Patienten 398 vorher entfernen. Auf einzelne Beobachtungen innerhalb der Daten kann man mit Hilfe eckiger Klammern [und der Angabe des Index zugreifen.

```
1 ## Patient 398
2 ITSDaten$Temperatur[398]
```

```
[1] 9.1
```

Ein negativer Index bedeutet, dass dieser Index ausgeschlossen werden soll. Wir erhalten damit

```
1 Skew( ITSDaten$Temperatur [-398] )
```

```
[1] 0.3142909
```

Die Schiefe ist jetzt nahe 0 und bestätigt damit unseren ersten Eindruck. Die Verteilung der Werte muss, Patient 398 ausgenommen, doch recht symmetrisch um das arithmetische Mittel herum sein. Durch Weglassen von Patient 398 reduziert sich auch die Standardabweichung deutlich

```
1 sd( ITSDaten$Temperatur [-398] )
```

```
[1] 1.173187
```

und liegt nun sehr nahe beim standardisierten MAD und standardisierten IQR.

Anmerkung:

Einzelne Ausreißer können einen starken Einfluss auf gewisse Statistiken haben und die Ergebnisse deutlich verfälschen. Beispiele hierfür sind das arithmetische Mittel, die Varianz/Standardabweichung und die Schiefe. Es ist daher wichtig, die vorliegenden Daten immer hinsichtlich auffälliger Werte zu untersuchen. Im Zweifelsfall sollte man sich besser auf die Ergebnisse robuster Verfahren wie Median, MAD oder IQR verlassen.

Wir berechnen die Schiefe für die Aufenthaltsdauer (LOS). Anhand von arithmetischem Mittel und Median haben wir oben auf eine rechtsschiefe Verteilung geschlossen und würden daher eine positiven Wert für die Schiefe erwarten.

```
1 Skew( ITSDaten$LOS )
```

```
[1] 4.880826
```

Das Ergebnis bestätigt folglich unsere erste Analyse, denn wir erhalten eine recht große Schiefe. Die Verteilung der LOS-Werte ist demnach deutlich rechtsschief.

Ein weiteres Formmaß ist die Wölbung.

Definition 2.24 (Wölbung). *Gegeben seien Beobachtungen $x_1, x_2, \dots, x_n \in \mathbb{R}$ ($n \in \mathbb{N}$). Dann ist die Wölbung oder Kurtosis definiert als*

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

Für $Kurt(x_1, \dots, x_n) < 0$ spricht man von **flachgipflig** oder **platykurtisch**, für $Kurt(x_1, \dots, x_n) > 0$ von **steilgipflig** oder **leptokurtisch**.

Wir geben einige zusätzliche Erklärungen.

Bemerkung 2.25. Der Bezugspunkt für die Definition ist die Wölbung der Normalverteilung (vgl. Abschnitt 4.2). Durch die Subtraktion von 3 in der obigen Definition besitzt die Normalverteilung gerade eine Kurtosis von 0. Man spricht zur Verdeutlichung in diesem Fall auch von **Exzess** oder **Exzess-Kurtosis**. Beobachtet man eine negative (Exzess-)Kurtosis, so verläuft die Verteilung flacher und weniger stark gewölbt als die Normalverteilung. Ist die (Exzess-)Kurtosis positiv, ist die Verteilung steiler und stärker gewölbt als die Normalverteilung; siehe auch Abbildung 2.9.

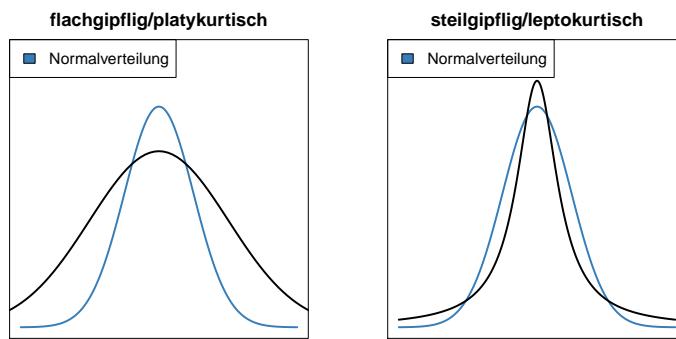


Abbildung 2.9: Beispiele für Kurtosis.

Wir berechnen die Kurtosis für die maximale Körpertemperatur der ITS-Patienten mit der Funktion `Kurt` aus dem Paket "DescTools" (Andri et mult. al. (2022)). Aufgrund des großen Einflusses von Patient 398 auf die Schiefe vergleichen wir die Kurtosiswerte mit und ohne diesen Patienten.

```
1 Kurt (ITSDaten$Temperatur)
```

```
[1] 144.4649
```

```
1 Kurt (ITSDaten$Temperatur [-398])
```

```
[1] 0.3431707
```

Erneut sehen wir, welchen großen Einfluss einzelne Beobachtungen haben können. Wir können davon ausgehen, dass die Verteilung der Werte nicht extrem steilgipflig ist, sondern mit Ausnahme eines Punktes recht gut durch eine Normalverteilung beschrieben werden kann. Wir berechnen die Kurtosis für die Aufenthaltsdauer (LOS).

```
1 Kurt (ITSDaten$LOS)
```

```
[1] 33.59482
```

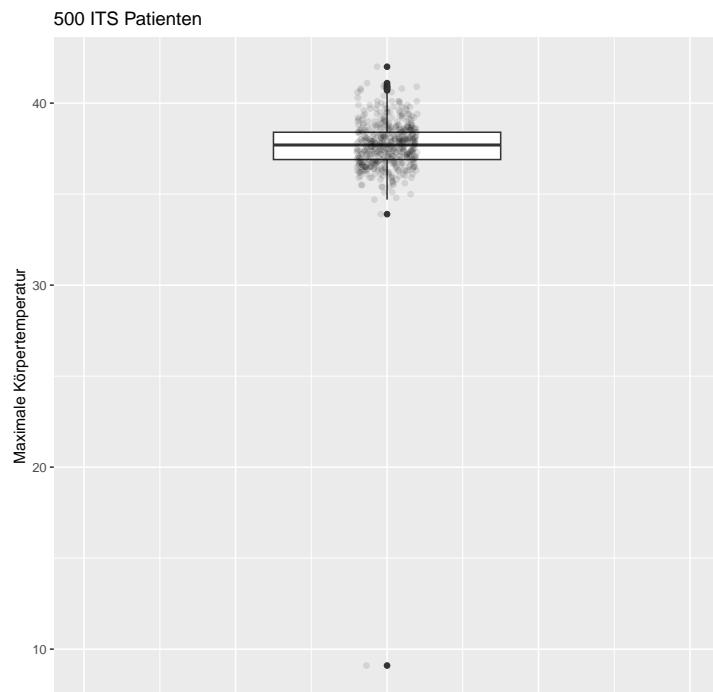
Die Werteverteilung für LOS ist demnach deutlich steilgipfliger als eine Normalverteilung.

Wir fahren fort mit verschiedenen graphischen Darstellungsmöglichkeiten für metrische Daten. Wir beginnen mit einem Box-und-Whisker Plot der maximalen Körpertemperatur. Wir verwenden die Funktionen aus dem Paket "ggplot2" (Wickham (2009)).

```

1 ## Box- und Whisker Plot an der Stelle x = 1
2 ggplot(ITSData, aes(x = 1, y = Temperatur)) +
3   geom_boxplot() + xlim(0, 2) + ylab("Maximale Körpertemperatur") +
4   geom_jitter(height = 0, width = 0.1, alpha = 0.1) +
5   ggttitle("500 ITS Patienten") +
6   ## keine Achsenbeschriftung für die x-Achse
7   xlab("") + theme(axis.ticks.x = element_blank(),
8                     axis.text.x = element_blank())

```



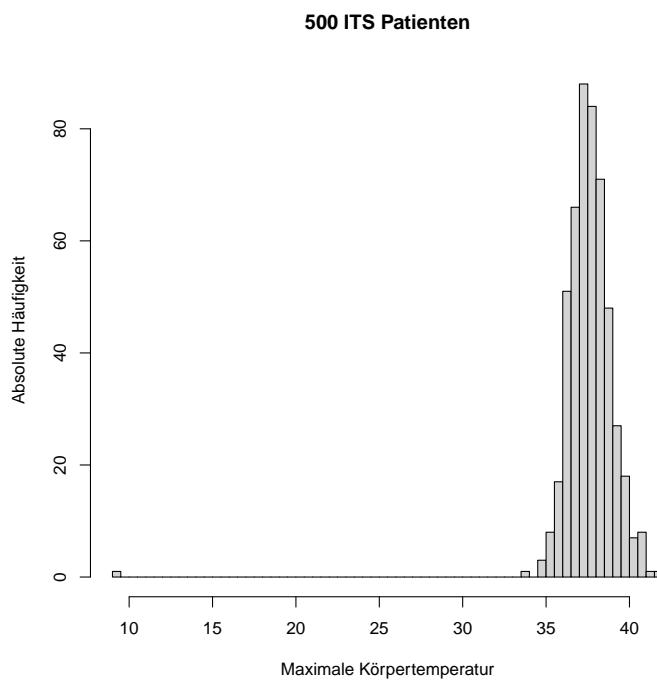
Wir sehen einige leichte Ausreißer nach oben und unten und den Wert von Patient 398, der sich extrem von den anderen unterscheidet.

Wir analysieren die Verteilung der Werte genauer unter Verwendung eines Histogramms. Ein **Histogramm** ist ein spezielles Balkendiagramm, welches man erhält, indem man den Wertebereich einer metrischen Variable in aufeinanderfolgende Intervalle unterteilt. Für jedes der Intervalle stellt man dann die absolute oder relative Häufigkeit mittels eines Balken dar. Für die Wahl der Intervalle gibt es Faustformeln, mit denen die Software automatisch eine Anzahl gleich großer Intervalle wählen kann. Meistens ist es jedoch besser, die Intervalle selbst zu wählen und hier eine zum Kontext passende Einteilung vorzunehmen. Wir erzeugen ein Histogramm der gemessenen maximalen Körpertemperaturen mit Hilfe der Funktion `hist`, wobei wir Intervalle mit einer Länge von 0.5°C verwenden, die wir über das Argument `breaks` der Funktion vorgeben.

```

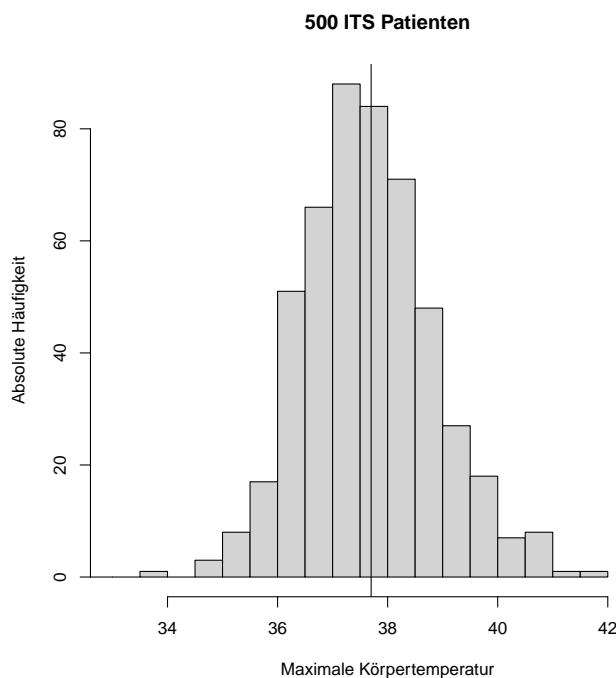
1 hist(ITSData$Temperatur, breaks = seq(from = 9.0, to = 42, by = 0.5),
2       main = "500 ITS Patienten", xlab = "Maximale Körpertemperatur",
3       ylab = "Absolute Häufigkeit")

```



Wir sehen erneut deutlich den Ausreißer. Um ein besseres Bild von der Verteilung zu bekommen, können wir entweder den Wert des Patienten 398 entfernen oder aber die x-Achse mit Hilfe des Arguments `xlim` entsprechend einschränken. Wir wählen dieses Mal die zweite Möglichkeit. Wir ergänzen außerdem mit Hilfe der Funktion `abline` eine vertikale Linie für den Median.

```
1 hist(ITSDaten$Temperatur, breaks = seq(from = 9.0, to = 42, by = 0.5),
2       main = "500 ITS Patienten", xlab = "Maximale Körpertemperatur",
3       ylab = "Absolute Häufigkeit", xlim = c(33,43))
4 abline(v = median(ITSDaten$Temperatur))
```



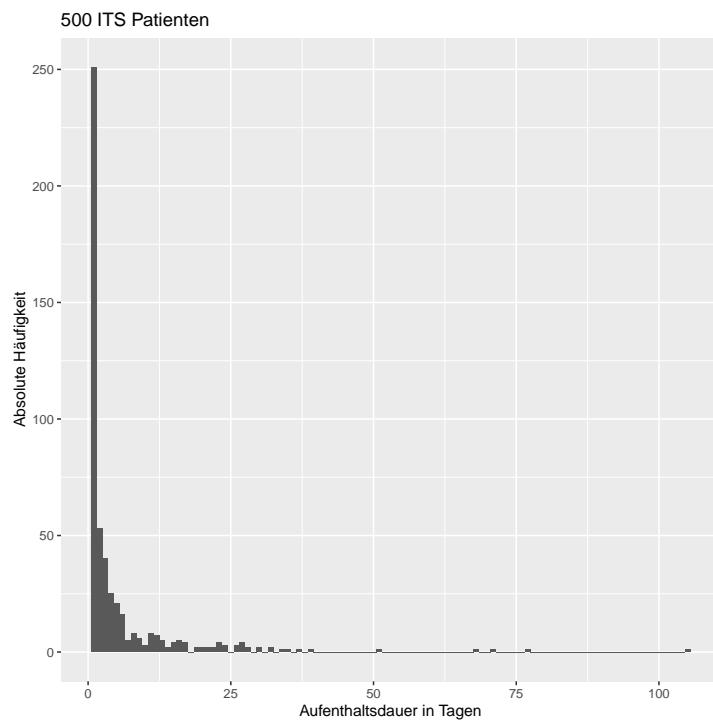
Das Bild bestätigt die vorangegangenen Berechnungen; d.h., die Verteilung ist relativ symmetrisch um das arithmetische Mittel bzw. dem Median herum und die Verteilung der maximalen Körpertemperatur für die ITS Population kann (von Ausreißern abgesehen) vermutlich recht gut durch eine Normalverteilung beschrieben werden.

In einer zweiten Analyse werfen wir einen Blick auf die Aufenthaltsdauer, wobei wir dieses Mal die Funktion `geom_histogram` aus dem Paket "ggplot2" (Wickham (2009)) verwenden. Als Intervalllänge für die Kategorien verwenden wir einen Tag. Dies können wir mit dem Parameter `binwidth` vorgeben.

```

1 ggplot(ITSData, aes(x = LOS)) + geom_histogram(binwidth = 1) +
2   xlab("Aufenthaltsdauer in Tagen") + ylab("Absolute Häufigkeit") +
3   ggttitle("500 ITS Patienten")

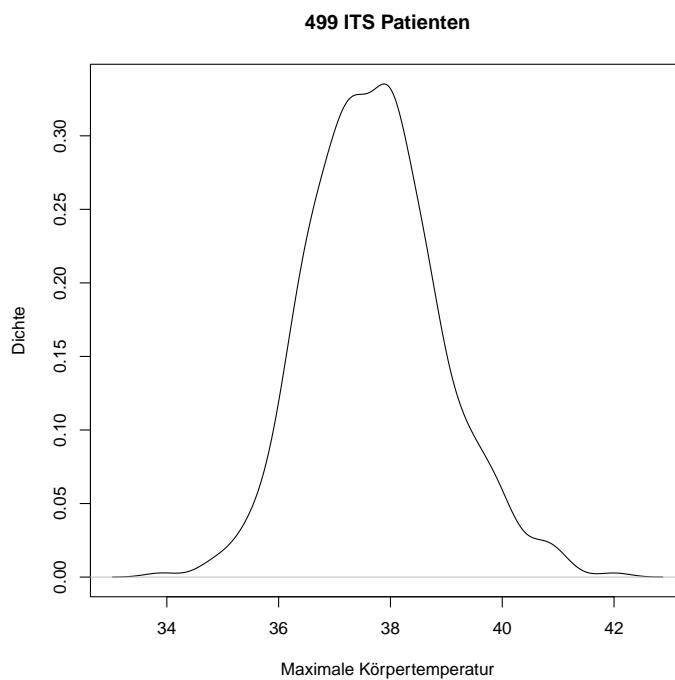
```



Das Bild bestätigt unsere Berechnungen. Wir erhalten eine eindeutig rechtsschiefe und recht spitze Verteilung. Die Mehrzahl der Patienten hatte eine Aufenthaltsdauer von wenigen Tagen. Die maximale Aufenthaltsdauer betrug 105 Tage.

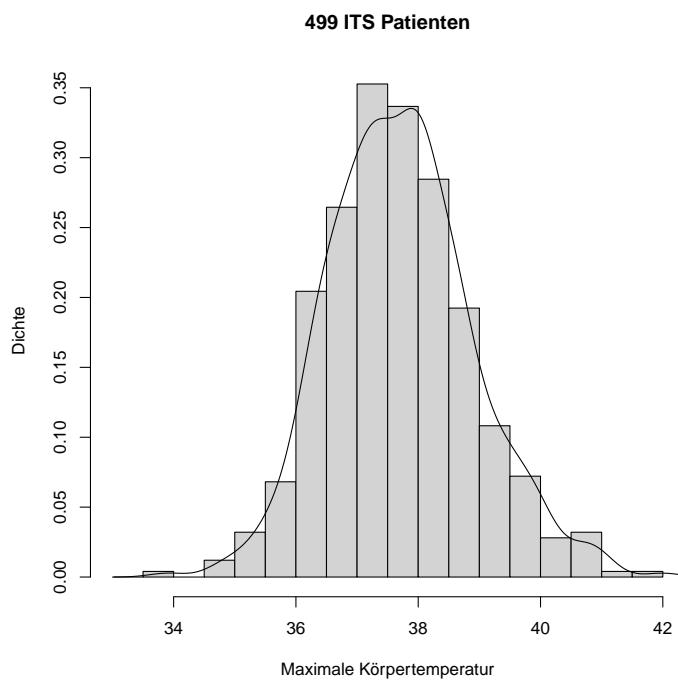
Alternativ kann man die Verteilung der Werte auch mittels ihrer geschätzten Dichte darstellen. Unter der (empirischen) **Dichte** kann man sich eine geglättete Version des Histogramms vorstellen. In R können wir zur Berechnung der Dichte (genauer: der Kerndichteschätzung) die Funktion density verwenden. Das Ergebnis dieser Berechnung kann mit Hilfe von plot graphisch dargestellt werden. Wir betrachten die maximale Körpertemperatur und lassen den Wert von Patient 398 unberücksichtigt.

```
1 plot(density(ITSDaten$Temperatur[-398]), xlab = "Maximale Körpertemperatur",
2      ylab = "Dichte", main = "499 ITS Patienten")
```



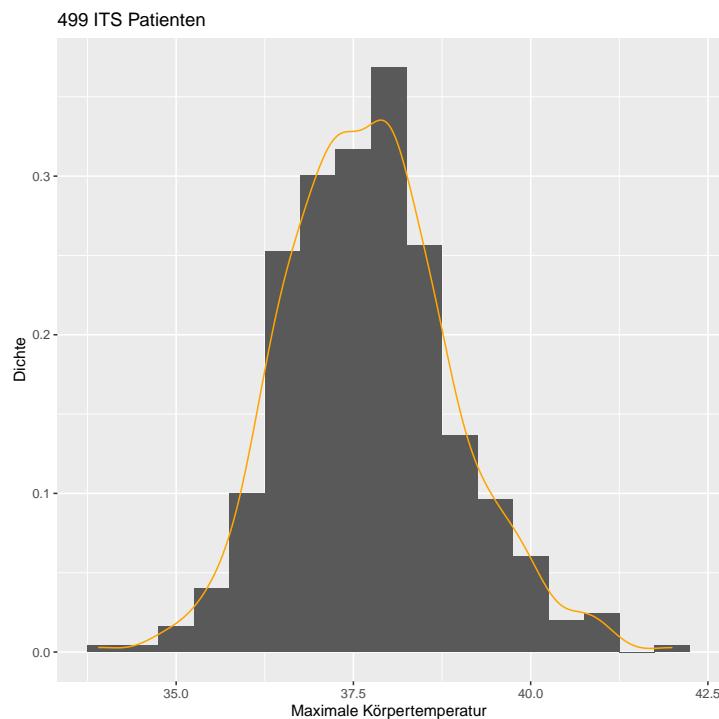
Wir erhalten eine recht symmetrische Dichte um das arithmetische Mittel herum. Wollen wir Histogramm und Dichte zusammen darstellen, müssen wir bei `hist` zusätzlich den Parameter `freq = FALSE` verwenden. Damit erreichen wir, dass auch das Histogramm auf der Dichteskala dargestellt wird. Für das Hinzufügen der Dichte kommt die Funktion `lines` zum Einsatz, mit der zu bereits bestehenden Plots nachträglich noch Linien hinzugefügt werden können.

```
1 hist(ITSDaten$Temperatur[-398], breaks = seq(from = 33, to = 42, by = 0.5),
2       xlab = "Maximale Körpertemperatur", ylab = "Dichte", freq = FALSE,
3       main = "499 ITS Patienten")
4 lines(density(ITSDaten$Temperatur[-398]))
```



Wir sehen, dass sich die Dichtekurve gut an das Histogramm anfügt. Für die Erzeugung eines ähnlichen Plots mit Hilfe des Pakets "ggplot2" (Wickham (2009)) können wir die Funktionen `geom_histogram` und `geom_density` verwenden.

```
1 ggplot(ITSDaten[-398,], aes(x=Temperatur)) +  
2   geom_histogram(aes(y=after_stat(density)), binwidth = 0.5) +  
3   geom_density(color = "orange") + ylab("Dichte") +  
4   xlab("Maximale Körpertemperatur") +  
5   ggtitle("499 ITS Patienten")
```

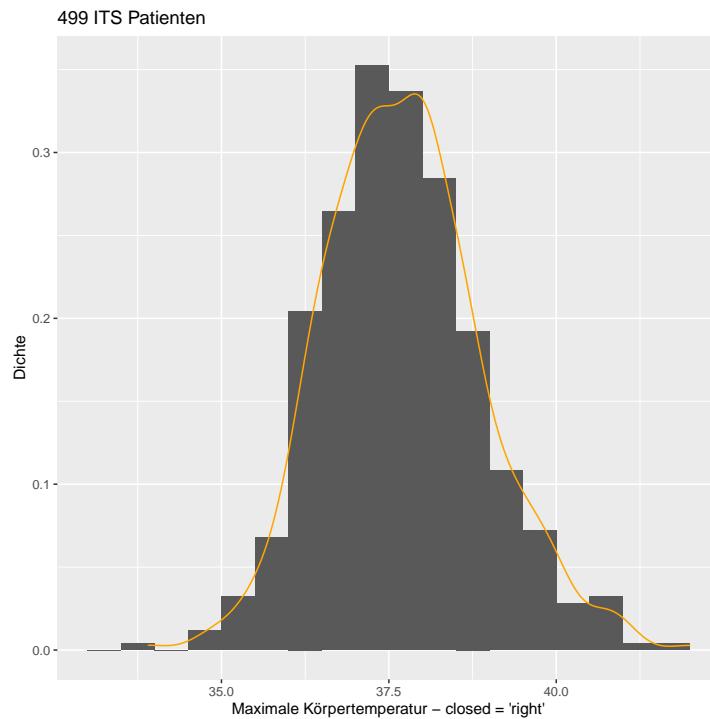


Die geschätzten Dichten sind in beiden Grafiken identisch, die Histogramme hingegen unterscheiden sich etwas. Dies liegt daran, dass im Fall von `geom_histogram` nach rechts offene und links geschlossene Intervalle betrachtet werden. Im Fall von `hist` ist es umgekehrt; d.h., links offen und rechts geschlossen. Außerdem haben wir durch die Verwendung von `binwidth` keine genaue Kontrolle über die Intervalle. Durch zusätzliches Setzen von `closed = "right"` und der Verwendung des Argument `breaks` in `geom_histogram` können wir ein identisches Histogramm erzeugen.

```

1 ggplot(ITSData[ -398 , ], aes(x=Temperatur)) +
2   geom_histogram(aes(y=after_stat(density)),
3                 breaks = seq(from = 33, to = 42, by = 0.5),
4                 closed = "right") +
5   geom_density(color = "orange") + ylab("Dichte") +
6   xlab("Maximale Körpertemperatur - closed = 'right'") +
7   ggtitle("499 ITS Patienten")

```



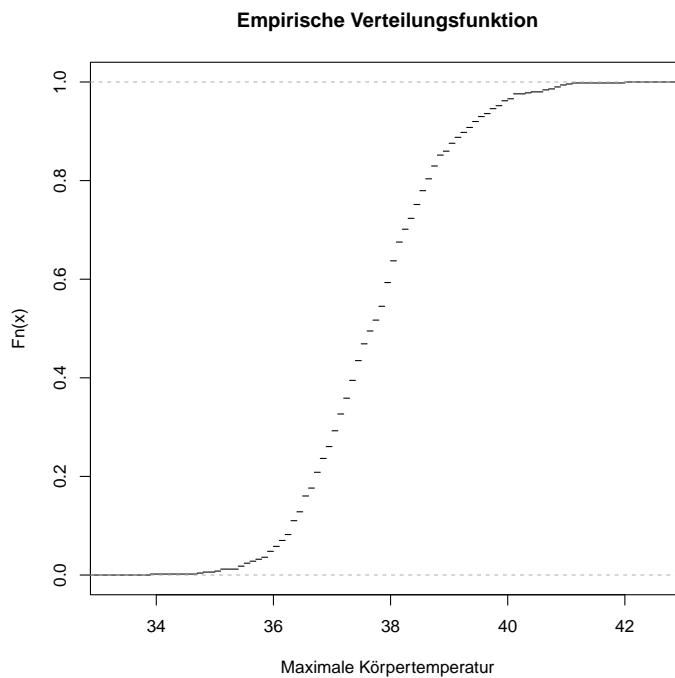
Damit sind die beiden Histogramme identisch.

Anmerkung:

Im Fall von ggplot schließen wir nicht nur den Temperaturwert von Patient 398 aus, sondern entfernen mit [-398,] den Patienten komplett, genauer die Zeile 398, aus dem Datensatz.

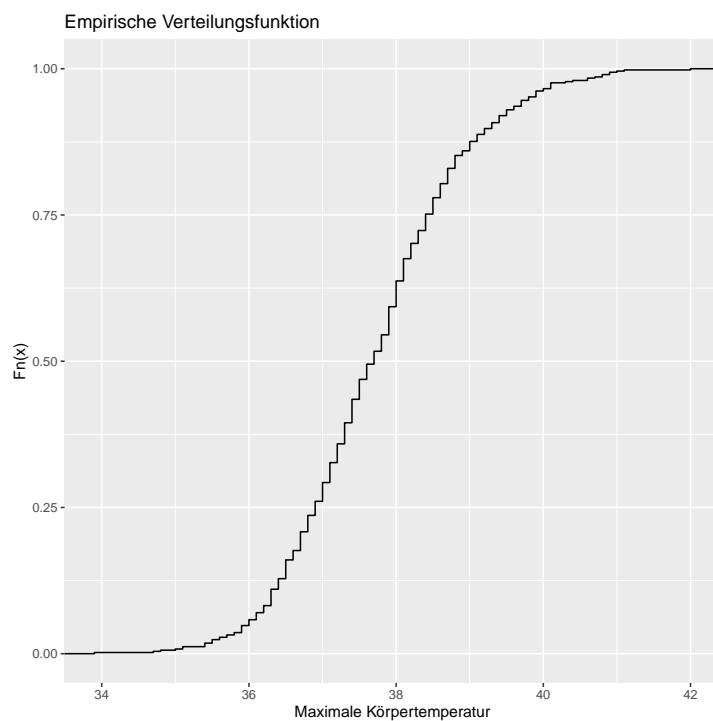
Wir können die Verteilung der maximalen Temperaturwerte auch mittels der empirischen Verteilungsfunktion (vgl. Definition 2.7) graphisch darstellen. Wir verwenden zunächst die Funktionen ecdf und plot, wobei wir wieder Patient 398 vorher entfernen.

```
1 plot(ecdf(ITSData$Temperatur[-398]), main = "Empirische Verteilungsfunktion",
2      xlab = "Maximale Körpertemperatur", do.points = FALSE)
```



Aufgrund der feinen Unterteilung und der vielen kleinen Sprünge, verzichten wir auf das Einzeichen von Punkten (`do.points = FALSE`). Wir generieren die entsprechende Abbildung mit Hilfe der Funktion `stat_ecdf` aus dem Paket "ggplot2" (Wickham (2009)).

```
1 ggplot(ITSData[ -398 ,], aes(x = Temperatur)) + stat_ecdf() +
2   xlab("Maximale Körpertemperatur") + ylab("Fn(x)") +
3   ggtitle ("Empirische Verteilungsfunktion")
```



Eine weitere wichtige Anwendung dieser Möglichkeiten, die empirische Verteilung der Daten zu visualisieren, ist, die Verteilung des angenommenen Wahrscheinlichkeitsmodells in die Darstellung mit aufzunehmen. Damit ist eine graphische Überprüfung des angenommen Modells möglich. Wir werden dies in Kapitel 5 genauer behandeln.

2.6.2 Bivariate Analyse

Die Stärke und Richtung des Zusammenhangs zwischen metrischen Variablen können wir ähnlich wie im Fall ordinaler Daten mit Hilfe der Korrelation beschreiben (vgl. Abschnitt 2.5.2). Neben den Rangkorrelationen kann hierfür die Pearson-Korrelation herangezogen werden.

Definition 2.26. Gegeben seien Beobachtungspaare $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n) \in \mathbb{R}^2$. Dann ist die **Pearson-Korrelation** definiert durch

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

Die Pearson-Korrelation kann Werte im Intervall $[-1, 1]$ annehmen, wobei 1 für einen perfekten positiven linearen Zusammenhang und -1 für einen perfekten negativen linearen Zusammenhang steht.

Wir geben einige zusätzliche Erläuterungen.

Bemerkung 2.27. (a) Die Annahme, dass ein linearer Zusammenhang vorliegt und somit die Pearson-Korrelation geeignet ist, um die Stärke dieses Zusammenhangs zu beschreiben, stellt eine recht starke Voraussetzung dar. Die Rangkorrelationen besitzen hier mit der Möglichkeit, monotone Zusammenhänge zu beschreiben, deutlich breitere Anwendungsmöglichkeiten.

(b) Eine genauere Betrachtung der Formeln zeigt, dass Spearmans ρ (vgl. Definition 2.11) gerade die Pearson-Korrelation der Ränge ist.

(c) Ergänzt man im Zähler der Pearson-Korrelation noch den Vorfaktor $\frac{1}{n}$ entspricht dieser gerade der **Stichproben-Kovarianz** der beiden betrachteten Variablen. Erweitert man den Nenner entsprechend, so erhält man das Produkt der beiden Standardabweichungen. Somit kann man die Pearson-Korrelation auch als eine standardisierte Kovarianz auffassen.

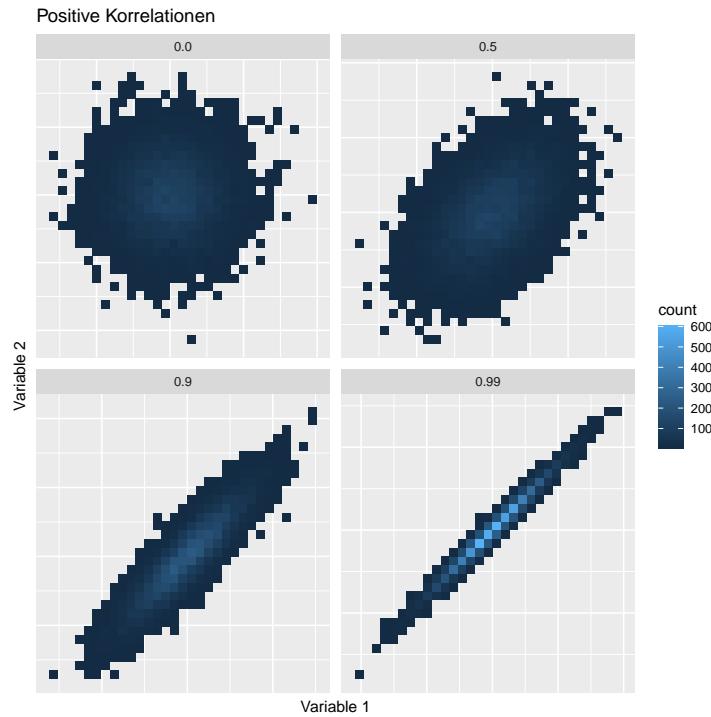
Wir untersuchen den Zusammenhang zwischen der maximalen Körpertemperatur und der maximalen Herzfrequenz.

```
1 cor(ITS Daten$Temperatur, ITS Daten$Herzfrequenz)
```

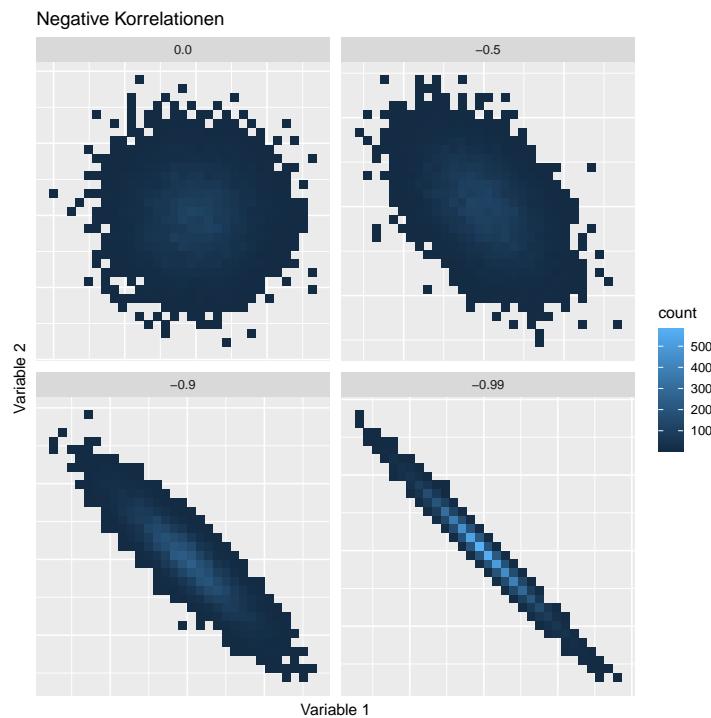
```
[1] 0.1763067
```

Wir erhalten demnach einen leicht positiven Zusammenhang; d.h., mit steigender maximaler Körpertemperatur nimmt tendenziell auch die maximale Herzfrequenz linear zu. Die Aussagekraft dieses Ergebnisses ist wieder schwer zu beurteilen, da unklar ist, ob wirklich ein linearer Zusammenhang zwischen

diesen beiden Variablen vorliegt. Bevor wir die Daten darstellen, wollen wir uns zunächst einmal ansehen, wie Punktewolken von Daten, die von zwei linear zusammenhängenden Variablen erzeugt werden, überhaupt aussehen. Wir verwenden hierfür die Funktion `simCorVars` aus dem Paket "MKdescr" (Kohl (2022a)).



Wir erhalten demnach ellipsenförmige Punktewolken und im Fall einer Korrelation von 0 einen Kreis. Im Fall negativer Korrelationen erhalten wir entsprechend gespiegelte Punktewolken.

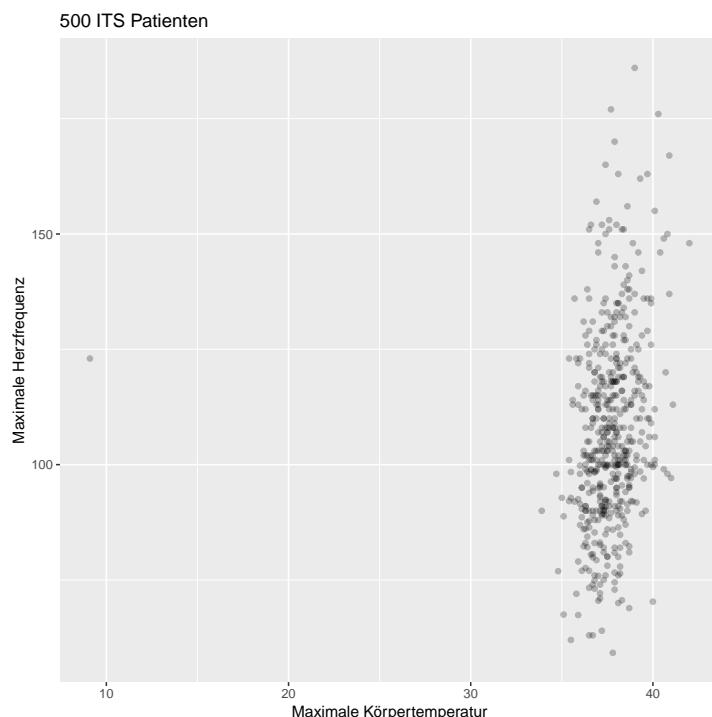


Wir stellen die vorliegenden Daten mit einem Streudiagramm dar.

```

1 ggplot(ITSDataen, aes(x=Temperatur, y=Herzfrequenz)) +
2   ## shape = 19: etwas größerer Punkt
3   ## alpha = 0.25: Stärke des Blendings
4   geom_point(shape=19, alpha=0.25) +
5   ## Beschriftung
6   ggttitle("500 ITS Patienten") + xlab("Maximale Körpertemperatur") +
7   ylab("Maximale Herzfrequenz")

```



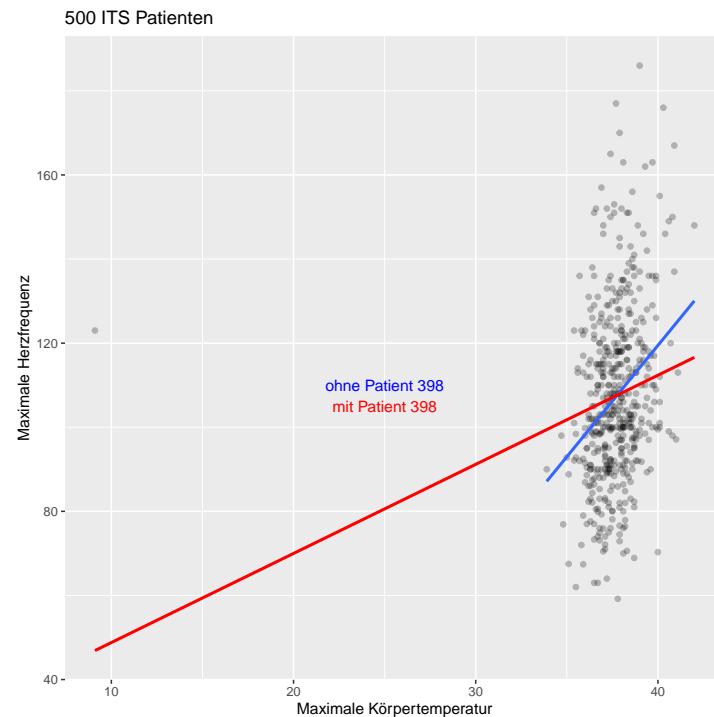
Wir sehen wieder den auffälligen Wert von Patient 398. Außerdem sehen wir eine recht enge ellipsenförmige Punktwolke, welche auf eine hohe positive Korrelation hindeutet und der Berechnung zu widersprechen scheint. Hier ist zunächst zu beachten, dass der Wert von Patient 398 einen sogenannten **Heelpunkt** darstellt; d.h., er hebelt unsere Analyse aus. Dies ist sofort zu sehen, wenn wir mit Hilfe der Funktion `geom_smooth` die entsprechenden beiden linearen Regressionsgeraden (mit und ohne Patient 398) zum Plot hinzufügen. Wir beschriften den Plot außerdem durch Anwendung der Funktion `annotate`.

```

1 ggplot(ITSDataen, aes(x=Temperatur, y=Herzfrequenz)) +
2   ## shape = 19: etwas größerer Punkt
3   ## alpha = 0.25: Stärke des Blendings
4   geom_point(shape=19, alpha=0.25) +
5   ## Lineare Regressionsgeraden
6   geom_smooth(data = ITSDataen[-398,], method = "lm", se = FALSE) +
7   geom_smooth(method = "lm", se = FALSE, color = "red") +
8   annotate("text", x = c(25, 25), y = c(110, 105),
9           label = c("ohne Patient 398", "mit Patient 398"),
10          color = c("blue", "red"))

```

```
11  ## Beschriftung
12  ggtile("500 ITS Patienten") + xlab("Maximale Körpertemperatur") +
13  ylab("Maximale Herzfrequenz")
```



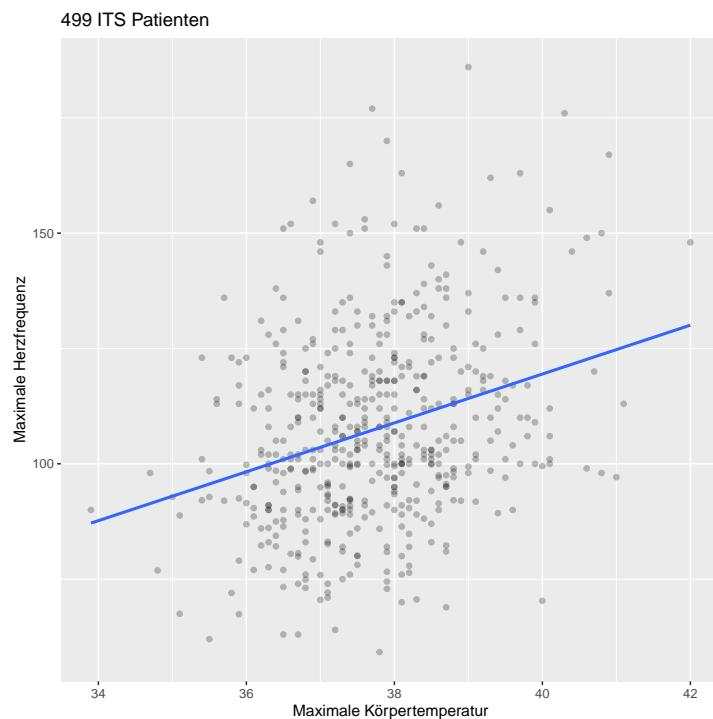
Die blaue Regressionsgerade, die wir ohne den Wert von Patient 398 erhalten, kippt durch Hinzunahme von Patient 398 sichtbar weg und wir erhalten ein deutlich anderes Ergebnis (rote Regressionsgerade). Wir wiederholen die Analyse daher ohne diesen Patienten.

```
1 cor(ITSData$Temperatur[-398], ITSData$Herzfrequenz[-398])
```

```
[1] 0.2978033
```

Die Pearson-Korrelation hat sich nahezu verdoppelt, entspricht aber immer noch nicht dem visuellen Eindruck. Die zu sehende sehr enge ellipsenförmige Punktwolke spricht für eine noch deutlich höhere positive Korrelation. Der Grund hierfür ist, dass der Ausreißer nicht nur die Berechnung aushebelt, sondern auch die Achsenkalierung deutlich verändert. Wir wiederholen die Darstellung ohne den Patienten 398.

```
1 ggplot(ITSData[-398,], aes(x=Temperatur, y=Herzfrequenz)) +
2   ## shape = 19: etwas größerer Punkt
3   ## alpha = 0.25: Stärke des Blendings
4   geom_point(shape=19, alpha=0.25) +
5   ## Lineare Regressionsgerade
6   geom_smooth(method = "lm", se = FALSE) +
7   ## Beschriftung
8   ggtile("499 ITS Patienten") + xlab("Maximale Körpertemperatur") +
9   ylab("Maximale Herzfrequenz")
```



Jetzt decken sich die graphische Darstellung und der berechnete Wert recht gut, und wir können eine schwach ellipsenförmige Punktwolke erahnen.

Anmerkung:

Einzelne Ausreißer können einen sehr starken Einfluss auf viele statistische Verfahren haben. Dazu zählen zum Beispiel auch die Pearson-Korrelation und die lineare Regressionsanalyse. Die Ergebnisse können davon, anschaulich gesprochen, förmlich ausgehebelt werden. Vor diesem Effekt kann auch das Vorliegen von sehr vielen Daten (“big data”) nicht zuverlässig schützen. Im Extremfall kann durch eine einzelne Beobachtung ein völlig anderes statistisches Ergebnis verursacht werden. Man nennt dies in der Statistik auch einen Zusammenbruch (“breakdown”) des verwendeten statistischen Verfahrens.

Wir untersuchen die Auswirkung des Ausreißers auf Spearmans ρ und Kendalls τ .

```
1 ## Spearmans rho
2 cor(ITSData$Temperatur, ITSData$Herzfrequenz, method = "spearman")
```

```
[1] 0.2659957
```

```
1 cor(ITSData$Temperatur[-398], ITSData$Herzfrequenz[-398], method = "spearman")
```

```
[1] 0.2707241
```

```
1 ## Kendalls tau
2 cor(ITSData$Temperatur, ITSData$Herzfrequenz, method = "kendall")
```

```
[1] 0.1826903
```

```
1 cor(ITSData$Temperatur[-398], ITSData$Herzfrequenz[-398], method = "kendall")
```

```
[1] 0.1858804
```

Die Werte der beiden Rangkorrelationskoeffizienten ändern sich nur wenig. Durch den Übergang zu Rängen erreicht man nicht nur eine allgemeinere Gültigkeit (monotoner Zusammenhang anstelle linearer Zusammenhang), sondern ähnlich wie bei den Quantilen eine gewisse Robustheit gegenüber Ausreißern (vgl. Beispiel 2.6). Wir berechnen die Pearson-Korrelation unter zusätzlicher Verwendung der Funktion `rank`.

```
1 cor(rank(ITSData$Temperatur), rank(ITSData$Herzfrequenz))
```

```
[1] 0.2659957
```

Es zeigt sich in der Tat, dass die Spearman-Korrelation gerade die Pearson-Korrelation der Ränge ist; vergleiche Bemerkung 2.27 (b).

Anmerkung:

Auch in der Statistik gilt das Sprichwort: “Ein Bild sagt mehr als tausend Worte”. Versuchen Sie daher immer, einen Blick auf Ihre Daten zu werfen. Dies dient sowohl zur Kontrolle der Daten, wie etwa für die Identifikation von falschen oder fehlerhaften Beobachtungen oder Ausreißern, als auch zur Validierung berechneter Statistiken.

2.7 Übungsaufgaben

Verwenden Sie den ITS-Datensatz und beschreiben Sie jeweils detailliert die Ergebnisse.

1. Berechnen Sie die absoluten und relativen Häufigkeiten für die Variable `Ergebnis`.
2. Verwenden Sie ein Balkendiagramm, um die relativen Häufigkeiten für die Variable `Ergebnis` darzustellen. Verwenden Sie hierfür sowohl die Standardfunktion `barplot` also auch das Paket "ggplot2" (Wickham (2009)).
3. Berechnen Sie das 95% Quantil, Median, Interquartilsabstand, MAD, arithmetisches Mittel, Standardabweichung, Variationskoeffizient, Schiefe und Kurtosis für die Variable `Herzfrequenz`. Vergleichen und interpretieren Sie die Werte. Was bedeuten die Ergebnisse für die Verteilung der Werte?
4. Berechnen Sie das 95% Quantil, Median, Interquartilsabstand, MAD, arithmetisches Mittel, Standardabweichung, Variationskoeffizient, Schiefe und Kurtosis für die Variable `Alter`. Vergleichen und interpretieren Sie die Werte. Was bedeuten die Ergebnisse für die Verteilung der Werte?
5. Berechnen Sie das 95% Quantil, Median, Interquartilsabstand, MAD, arithmetisches Mittel, Standardabweichung, Variationskoeffizient, Schiefe und Kurtosis für die Variable `L0S`. Vergleichen und interpretieren Sie die Werte. Was bedeuten die Ergebnisse für die Verteilung der Werte?

6. Zeichnen Sie einen Box-und-Whisker Plot sowie ein Histogramm zusammen mit einem Dichteplot für die Variable Herzfrequenz. Verwenden Sie hierfür sowohl die Standardfunktionen als auch die Funktionen aus dem Paket "ggplot2" (Wickham (2009)). Beschreiben und interpretieren Sie die Plots.
7. Zeichnen Sie einen Box-und-Whisker Plot sowie ein Histogramm zusammen mit einem Dichteplot für die Variable Alter. Verwenden Sie hierfür sowohl die Standardfunktionen als auch die Funktionen aus dem Paket "ggplot2" (Wickham (2009)). Beschreiben und interpretieren Sie die Plots.
8. Zeichnen Sie einen Box-und-Whisker Plot sowie ein Histogramm zusammen mit einem Dichteplot für die Variable LOS. Verwenden Sie hierfür sowohl die Standardfunktionen als auch die Funktionen aus dem Paket "ggplot2" (Wickham (2009)). Beschreiben und interpretieren Sie die Plots.
9. Untersuchen Sie auf geeignete Weise (!), ob es einen Zusammenhang zwischen den Variablen Leberversagen und Ergebnis gibt. Interpretieren Sie die Ergebnisse und stellen Sie die Daten geeignet graphisch dar. Ist die graphische Darstellung in Übereinstimmung mit Ihren Berechnungen?
10. Untersuchen Sie auf geeignete Weise (!), ob es einen Zusammenhang zwischen den Variablen Geschlecht und Ergebnis gibt. Interpretieren Sie die Ergebnisse und stellen Sie die Daten geeignet graphisch dar. Ist die graphische Darstellung in Übereinstimmung mit Ihren Berechnungen?
11. Überprüfen Sie auf geeignete Weise (!), ob es einen Zusammenhang zwischen den Variablen Alter und SAPS.II gibt. Interpretieren Sie die Ergebnisse und stellen Sie die Daten geeignet graphisch dar. Ist die graphische Darstellung in Übereinstimmung mit Ihren Berechnungen?
12. Überprüfen Sie auf geeignete Weise (!), ob es einen Zusammenhang zwischen den Variablen Alter und LOS gibt. Interpretieren Sie die Ergebnisse und stellen Sie die Daten geeignet graphisch dar. Ist die graphische Darstellung in Übereinstimmung mit Ihren Berechnungen?
13. Überprüfen Sie auf geeignete Weise (!), ob es einen Zusammenhang zwischen den Variablen Alter und Ergebnis gibt. Interpretieren Sie die Ergebnisse und stellen Sie die Daten geeignet graphisch dar. Ist die graphische Darstellung in Übereinstimmung mit Ihren Berechnungen?
14. Überprüfen Sie auf geeignete Weise (!), ob es einen Zusammenhang zwischen den Variablen Geschlecht und LOS gibt. Interpretieren Sie die Ergebnisse und stellen Sie die Daten geeignet graphisch dar. Ist die graphische Darstellung in Übereinstimmung mit Ihren Berechnungen?

3 Farben und Diagramme

Dieses relativ kurze Kapitel behandelt den richtigen Einsatz von Farben und die Erstellung von Diagrammen, welche die vorliegenden Daten möglichst gut präsentieren. Im einzelnen werden folgende Themen behandelt:

- Empfehlungen für den Umgang mit Farben
- Verwendung vordefinierter Farbpaletten
- Export von Diagrammen
- Empfehlungen für die Erstellung von Diagrammen nach E. Tufte

Der R Code für dieses Kapitel ist in der Datei `Farben.Rmd` enthalten, die Sie von meinem GitHub-Account herunterladen können (Link: <https://github.com/stamats/ESDR/blob/master/Farben.Rmd>). Klicken Sie mit der rechten Maustaste auf Raw. Dann können Sie das Ziel speichern unter.... Am wenigsten Schwierigkeiten ergeben sich, wenn Sie meine R Markdown Dateien im selben Ordner wie die Daten, welche in den jeweiligen Kapiteln verwendet werden, speichern.

Wir installieren zunächst die in diesem Kapitel benötigten Pakete.

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

Achten Sie darauf, dass Sie die Pakete von Kapitel 2 bereits installiert haben (vgl. Abschnitt 2.4). Wir laden alle Pakete, die wir in diesem Kapitel verwenden werden.

```
1 library(ggplot2)
2 library(RColorBrewer)
3 library(ggsci)
```

Wie bereits in Abschnitt 2.4 erklärt, ist ein wiederholtes Ausführen von `library` unproblematisch.

3.1 Farben

Wie wir bereits im letzten Kapitel gesehen haben, spielen Diagramme eine entscheidende Rolle für das Verständnis der Daten. Durch den richtigen Einsatz von Farben kann die Aussagekraft und die Ästhetik einer Grafik deutlich erhöht werden. Die Abbildung 3.1 zeigt ein Negativbeispiel. Es sind weder der Diagrammtyp noch die Farben gut gewählt. Aus dem gewählten Diagrammtyp sind die genauen Anteile (in absoluten oder relativen Zahlen) nicht ersichtlich. Die Farben rot und blau sind sehr intensiv und nicht an die Antworten angepasst. Die Grafik enthält noch eine weitere Schwäche: Es ist nicht klar, ob es

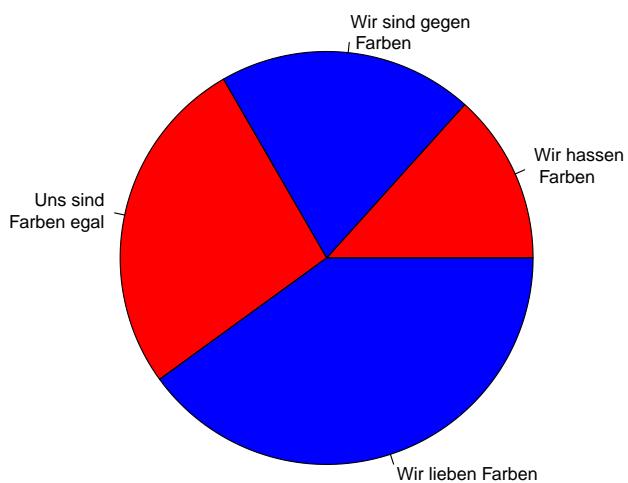


Abbildung 3.1: Ein Negativbeispiel für den Einsatz von Farben und Diagrammen.

zwischen “Uns sind Farben egal” und “Wir lieben Farben” entsprechend zur negativen Skala noch eine Zwischenkategorie gibt. Es könnte entweder sein, dass es diese Kategorie gibt und niemand eine entsprechende Antwort gab oder aber, dass diese Kategorie gar nicht vorgesehen war. In Kuchendiagrammen sind “leere” Kategorien nicht direkt ersichtlich.

Auf der DSC Konferenz 2003 führte Ross Ihaka folgende Möglichkeiten für den generellen Umgang mit Farben an (Ihaka (2003)):

1. Vermeide Farben.
2. Wähle die Farben durch Experimentieren aus.
3. Wähle die Farben nach “gutem Geschmack” oder Erfahrung.
4. Verwende vorgegebene Farbpaletten von Experten.
5. Suche nach Grundprinzipien für die Farbauswahl.

Wir kommentieren kurz die genannten Möglichkeiten:

Zu 1: Natürlich kann man versuchen, Farben zu vermeiden, Farben können jedoch sehr hilfreich sein und die Aussagekraft einer Grafik deutlich erhöhen.

Zu 2: Das Auswählen von Farben durch Experimentieren ist meist sehr zeitaufwendig.

Zu 3: Dies erfordert entweder ein spezielles Talent für Farben oder eine entsprechende Erfahrung im Umgang mit Farben.

Zu 4: Eine gute Idee!

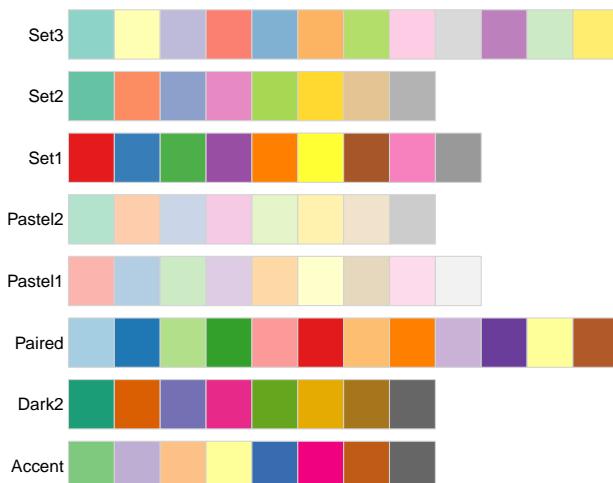
Zu 5: Wo findet man entsprechende Grundprinzipien?

Die folgenden grundlegenden Prinzipien im Umgang mit Farben sind in Zeileis et al. (2009) angegeben:

- Die Farben sollten attraktiv sein.
- Die Farben in einer statistischen Grafik sollten mit einander kooperieren.
- Die Farben sollten überall funktionieren.

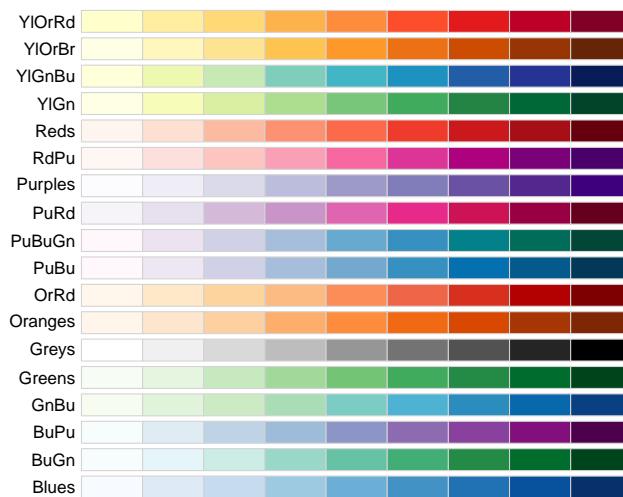
Ein Projekt, das diese Prinzipien umsetzt, ist ColorBrewer (Harrower and Brewer (2003)), welches Farbpaletten für verschiedene Zwecke auf der Internetseite www.colorbrewer2.org zur Verfügung stellt. Die dort angegeben Farbpaletten können in R über das Paket "RColorBrewer" (Neuwirth (2014)) genutzt werden. Wir laden das Paket und werfen einen Blick auf die verschiedenen Farbpaletten. Hierfür können wir die Funktion `display.brewer.all` verwenden. Wir betrachten zunächst die qualitativen Farbpaletten, die wir für die Darstellung von kategorialen Variablen verwenden können.

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



In diesem Fall ist es wichtig, dass keine Farbe die anderen Farben dominiert, sondern alle Farben gleich “wichtig” erscheinen. Die zweite Gruppe von Farbpaletten stellt Farben für Merkmale zur Verfügung, deren Merkmalsausprägungen von unwichtig bzw. uninteressant bis wichtig bzw. interessant reichen.

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



Schließlich gibt es noch eine dritte Gruppe von Farbpaletten für Variablen mit einem Wertebereich von negativ über neutral bis hin zu positiv.

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



Die Internetseite www.colorbrewer2.org ermöglicht es zusätzlich, die Farben nach den folgenden Kriterien auszuwählen:

- Für rot-grün blinde Personen
- Für Farbausdrucke
- Für schwarz-weiß Kopien
- Für Laptop Displays

Die Abbildung 3.2 vergleicht das einleitende Negativbeispiel mit einem entsprechenden Kuchendiagramm, bei dem die Farben an die Kategorien angepasst wurden. Beim neuen Diagramm wurde die ColorBrewer Palette RdYlGn verwendet. Eine weitere Verbesserung des Diagramms könnte entweder

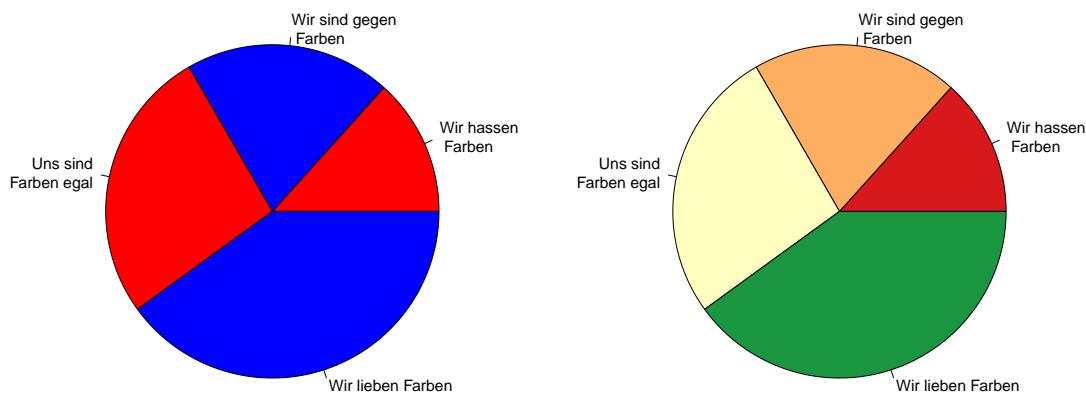


Abbildung 3.2: Ein Negativbeispiel mit verbesserter Farbauswahl.

durch eine Beschriftung der Kuchenstücke mit den zugehörigen (relativen oder absoluten) Häufigkeiten oder durch einen Übergang zu einem Balkendiagramm erreicht werden. Insbesondere könnte im Fall des Balkendiagramms durch einen Balken der Höhe 0 klargestellt werden, dass es auch eine Kategorie "Wir sind für Farben" gibt; siehe Abbildung 3.3.

Wir stellen die absolute Häufigkeit der Operationsarten wie in Abschnitt 2.5.1 graphisch dar, wobei wir zusätzlich die Farbpalette Set1 von ColorBrewer verwenden. Wir erzeugen hierfür zunächst einen entsprechenden Farbvektor mit Hilfe der Funktion `brewer.pal` aus dem Paket "RColorBrewer" (Neuwirth (2014)).

```
1 ## n = 5 Farben der Palette mit Namen Set1
2 farben <- brewer.pal(n = 5, name = "Set1")
3 farben
```

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

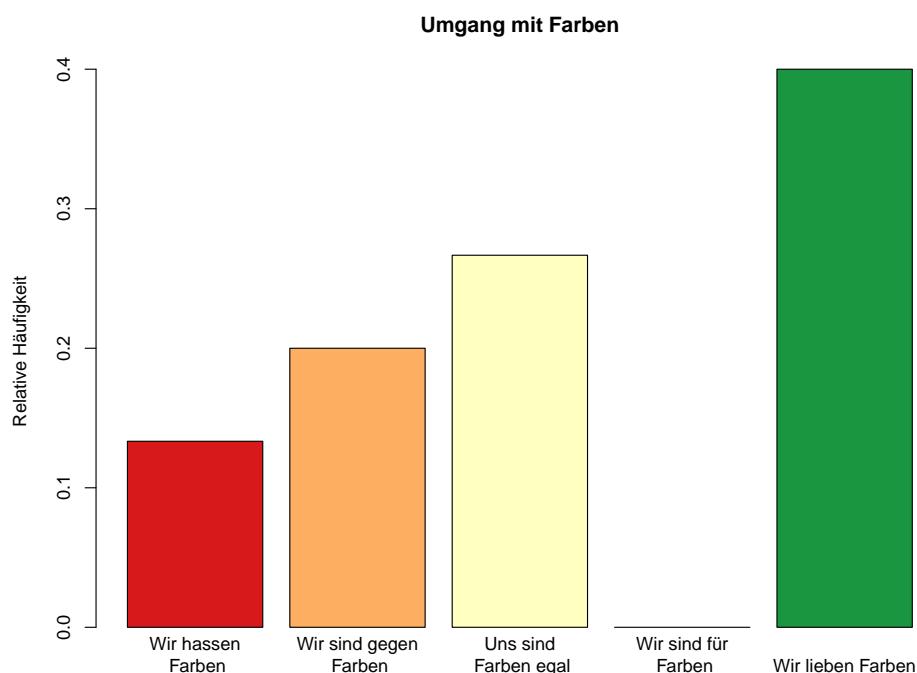


Abbildung 3.3: Von einem Negativ- zu einem Positivbeispiel.

Die Farben sind demnach in ihrem Hexadezimalcode angegeben. R stellt eine Reihe von Funktionen für verschiedene Farbräume zur Verfügung, mit denen man den Hexadezimalcode von Farben bestimmen kann. Ist zum Beispiel die rot-grün-blau (RGB) Kodierung bekannt, so kann man diese mit Hilfe der Funktion `rgb` umrechnen.

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

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

Man kann auch eine Vielzahl von Farben einfach durch ihren Namen angeben. Genauer sind in R 657 Farben über ihren Namen gespeichert. Man kann diese mit der Funktion `colors` anzeigen lassen und mit `col2rgb` die rot-grün-blau Kodierung dazu ermitteln; zum Beispiel

```
1 str(colors())
```

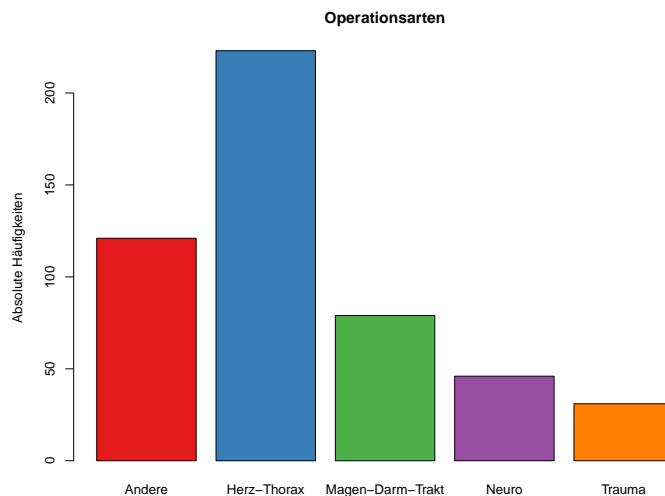
```
chr [1:657] "white" "aliceblue" "antiquewhite" "antiquewhite1" ...
```

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

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

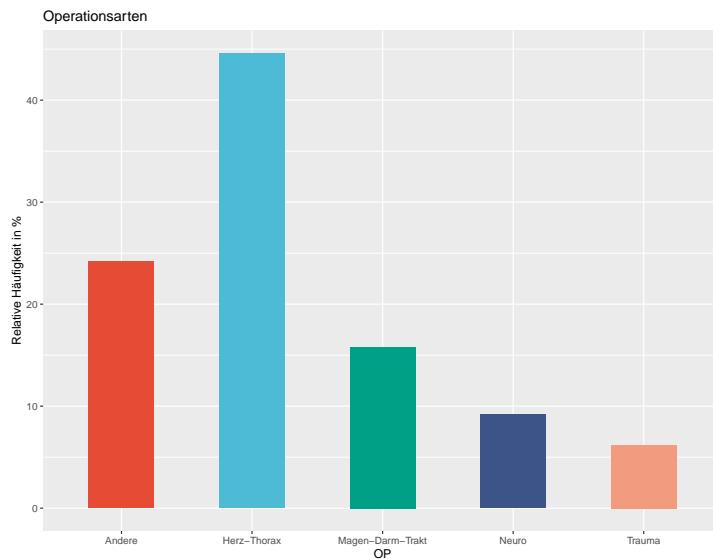
Die Standardfunktionen für das Plotten von Daten haben alle ein Argument `col`, mit dem man die ausgewählten Farben an die Funktion übergeben kann. Wir erzeugen jetzt das Balkendiagramm, wobei wir die Farbpalette `Set1` zum Füllen der Balken verwenden.

```
1 ITSDaten <- read.csv2(file = "ITSDaten.csv", fileEncoding = "utf8")
2 barplot(table(ITSDaten$OP), main = "Operationsarten",
3         ylab = "Absolute Häufigkeiten", col = farben)
```



Das Paket "ggplot2" (Wickham (2009)) stellt ebenfalls vielfältige Möglichkeiten zur Verfügung Farben einzusetzen. Wir wiederholen das Balkendiagramm der relativen Häufigkeiten aus Abschnitt 2.5.1, wobei wir dieses Mal die Balken mit der Palette `NPG` der Paketes "ggsci" (Xiao (2018)) einfärben.

```
1 ## Anlegen der Daten
2 ggplot(ITSDaten, aes(x=OP)) +
3   ## Hinzufügen der Balken für die relativen Häufigkeiten
4   geom_bar(aes(y = 100*after_stat(count)/sum(after_stat(count))), 
5             width = 0.5,
6   ## Füllen der Balken
7   fill = pal_npg()(5)) +
8   ## Titel und Beschriftung der y-Achse
9   ggtitle("Operationsarten") + ylab("Relative Häufigkeit in %")
```



Die verwendete Funktion `pal_npg()` gibt nicht direkt die Farbpalette zurück, sondern erzeugt eine Funktion, mit der eine vorgegebene Anzahl von Farben aus der NPG Farbpalette ausgewählt werden kann.

```
1 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: 0x5607f2ecfc38>

<environment: 0x5607fa5c2b10>

Deshalb sind im Code zwei aufeinanderfolgende Paare von runden Klammern zu sehen. Es wird zuerst mit dem Funktionsaufruf die genannte Funktion erzeugt und anschließend sofort mit dem Wert 5 ausgewertet.

```
1 pal_npg()(5)

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

Neben den vorgestellten Paketen "RColorBrewer" (Neuwirth (2014)) und "ggsci" (Xiao (2018)) gibt es noch eine Vielzahl weiterer Pakete, die unterschiedlichste Farbpaletten zur Verfügung stellen.

Zum Abschluss dieses Abschnitts möchten wir noch auf drei Situationen eingehen, die immer wieder im Zusammenhang mit Farbpaletten auftreten. Die erste Situation bezieht sich speziell auf das Paket "RColorBrewer".

```
1 farben2 ← 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
```

```
1 farben2

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

Die Warnmeldung, die vom obigen Code ausgelöst wird, zeigt, dass es in Fall von diesem Paket nicht möglich ist, weniger als drei Farben aus einer Farbpalette auszuwählen. Das Paket hält sich hierbei strikt an die Vorgabe der Internetseite www.colorbrewer2.org, auf der es ebenfalls nicht möglich ist, weniger als drei Farben auszuwählen. Benötigen wir also nur eine oder zwei Farben aus einer ColorBrewer 2.0 Farbpalette, so müssen wir hierfür zusätzlich mit Hilfe der eckigen Klammern auswählen. Wir selektieren im folgenden die erste und zweite Farbe.

```
1 farben2 ← brewer.pal(n = 3, name = "Set1")[1:2]
2 farben2

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

Damit haben wir nun einen Farbvektor erzeugt, der nur die ersten beiden Farben der Farbpalette enthält. Die zweite Situation, mit der wir uns befassen wollen, ist die gezielte Auswahl bestimmter Farben aus einer Farbpalette. Angenommen wir wollen aus der NPG Farbpalette die Farben orange, rot und grün und zwar genau in dieser Reihenfolge verwenden. Hierbei helfen uns erneut die eckigen Klammern in Kombination mit der Funktion `c`, mit der wir einen beliebigen Indexvektor für die Auswahl erzeugen können.

```
1 farben3 ← pal_npg()(5)[c(5,1,3)]
2 farben3

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

Wir wählen demnach zuerst die Farbe 5 (orange), dann die Farbe 1 (rot) und schließlich die Farbe 3 (grün). Abschließend wollen wir uns mit der Situation befassen, dass wir mehr Farben benötigen als in der ausgewählten Farbpalette enthalten sind. Mit der Funktion `colorRampPalette` können wir Farben interpolieren. Dies macht natürlich nicht bei jeder Farbpalette Sinn. Geeignet sind hierfür insbesondere sequentielle oder divergierende Farbpaletten. Für divergierende Farbenpaletten stellt ColorBrewer 2.0 maximal elf Farben zur Verfügung wie wir hier am Beispiel der Farbpalette RdYlBu (rot - gelb - blau) sehen.

```
1 farben11 <- brewer.pal(n = 11, name = "RdYlBu")
2 farben11
```

```
[1] "#A50026" "#D73027" "#F46D43" "#FDAE61" "#FEE090" "#FFFFBF" "#EOF3F8"
[8] "#ABD9E9" "#74ADD1" "#4575B4" "#313695"
```

Diese elf Farben werden wir nun zu 32 Farben erweitern.

```
1 farben32 <- colorRampPalette(farben11)(32)
2 farben32
```

```
[1] "#A50026" "#B50F26" "#C51E26" "#D52E26" "#DF412F" "#E85538" "#F26941"
[8] "#F67D4A" "#F99254" "#FCA75E" "#FDB96B" "#FDC97A" "#FDD989" "#FEE599"
[15] "#FEFOA8" "#FEFAB7" "#FAFDC8" "#F0F9DA" "#E6F5EC" "#D9EFF6" "#C8E7F1"
[22] "#B6DEEC" "#A5D4E6" "#93C6DE" "#82B8D7" "#70A9CF" "#6197C5" "#5285BC"
[29] "#4472B3" "#3D5EA9" "#374A9F" "#313695"
```

Wir erhalten die benötigte Anzahl an Farben. Die Funktion `colorRampPalette` erzeugt ähnlich wie die Funktion `pal_npg` eine Funktion, mit der dann die gewünschte Anzahl von Farben gewonnen werden kann. Dies erklärt, warum auch hier zwei Paare von runden Klammern nötig sind.

3.2 Exkurs: Export von Diagrammen

Beim Arbeiten mit RStudio werden die Diagramme im Fenster *Plots* angezeigt und können dort mit Hilfe des Menüpunktes *Export* in verschiedene Grafikformate oder pdf exportiert werden; siehe Abbildung 3.4. Anklicken von *Save as Image...* öffnet ein neues Fenster, in dem die Größe des Bildes, der Dateiname, das

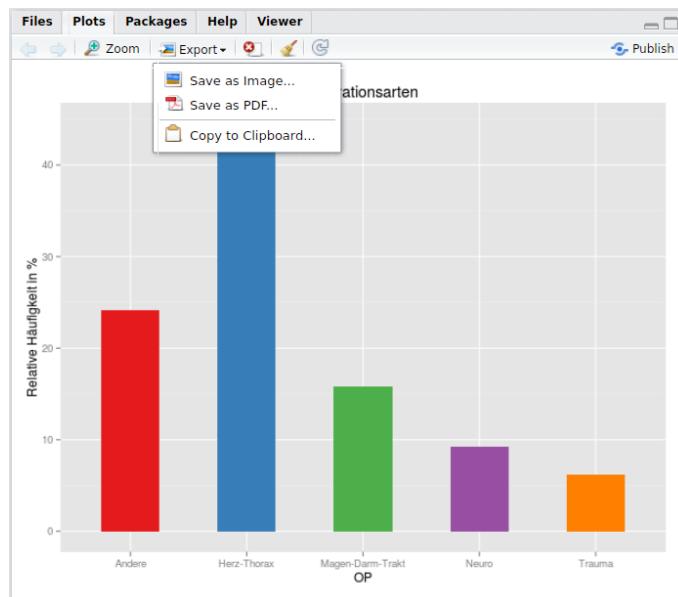


Abbildung 3.4: RStudio Fenster *Plots* mit einem Beispiel.

Verzeichnis und das Grafikformat ausgewählt werden können (vgl. Abbildung 3.5). Die zur Verfügung

stehenden Grafikformate hängen vom Betriebssystem sowie von evtl. zusätzlich installierter Software für Grafik ab. Wählt man stattdessen *Save as PDF...* öffnet sich das in Abbildung 3.6 dargestellte Fenster.

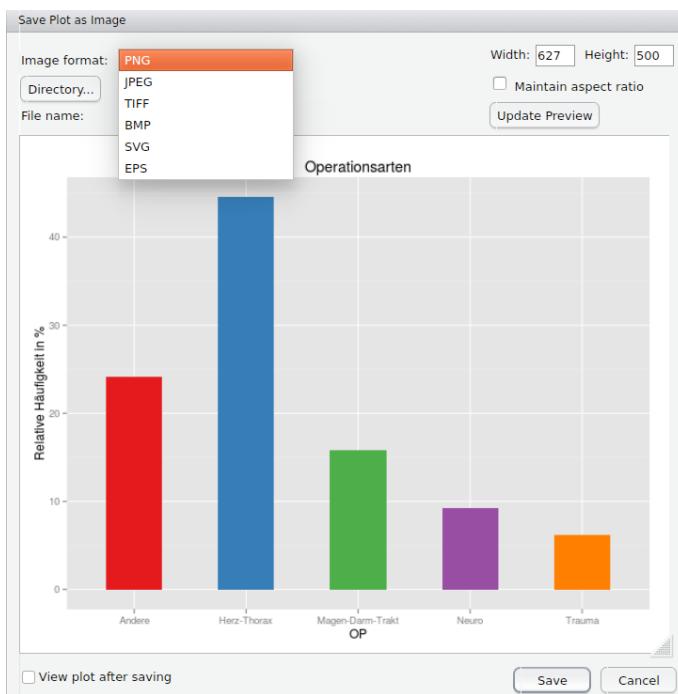


Abbildung 3.5: RStudio Fenster zum Abspeichern eines Plots als Bild.

Auch hier kann die Größe, der Dateiname und das Verzeichnis eingestellt werden. Für ein schnelles Ein-

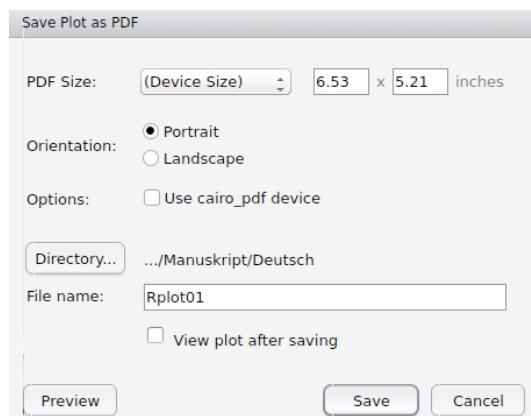


Abbildung 3.6: RStudio Fenster zum Abspeichern eines Plots als pdf-Datei.

fügen des Plots etwa in ein Textbearbeitungsprogramm oder eine E-Mail kann der Menüpunkt *Copy to Clipboard...* verwendet werden. In den meisten Fällen ist es jedoch besser, die Grafik zunächst als Bild oder pdf abzuspeichern und dann diese Datei in ein anderes Programm zu importieren.

Die oben beschriebenen Möglichkeiten sind sehr komfortabel und schnell und führen in den meisten Fällen zum gewünschten Ergebnis. Aber gerade bei komplexeren Grafiken kann es unter Umständen zu

Problemen kommen. Auch ist es manchmal nötig, beim Export weitere Parameter wie etwa die Auflösung, die Kompressionsrate oder auch die Schriftgröße anzupassen. In diesem Fall muss man für den Export direkt auf die dafür vorgesehenen Exportfunktionen von R zurückgreifen. Die zur Verfügung stehenden Grafikgeräte (devices) hängen, wie bereits oben erwähnt, vom Betriebssystem und evtl. zusätzlich installierter Software ab. Die Hauptfunktionalität wird durch das Grundpaket "grDevices" (R Core Team (2022a)) zur Verfügung gestellt. Darüber hinaus gibt es auch einige Erweiterungspakete dafür. In Tabelle 3.1 findet sich eine Übersicht über gängige Grafikgeräte (devices), die von R unterstützt werden.

Funktionsname	Beschreibung
bmp	Bitmap (bmp) ist ein Standardformat für Rastergrafiken unter Microsoft Windows.
jpeg	Komprimierte Bilddateien für Rastergrafiken, sehr verbreitet im Internet.
png	Portable network graphics (png) für verlustfrei komprimierte Bilddateien für Rastergrafiken, meist besser für statistische Grafiken geeignet als jpeg.
tiff	Tagged image file format (tiff) wird vor allem für hochauflöste druckfähige Rastergrafiken verwendet.
pdf	Portable document format (pdf) ein sehr weit verbreitetes Dateiformat, in dem Graphiken als Vektorgrafiken eingebettet sind.
postscript	PostScript (ps) ein Vektorgrafikformat, wird oft noch für den Druck verwendet, insbesondere die Weiterentwicklung Encapsulated PostScript (eps) ist für Grafiken interessant.
svg	Scalable vector graphics (svg) ein Vektorgraphikformat für Webbrowser.

Tabelle 3.1: Übersicht über Grafikgeräte (devices), die von R unterstützt werden.

Anmerkung:

Rastergrafiken basieren auf einem Pixelraster, weshalb sie auch Pixelgrafiken genannt werden. Dabei wird jedem Bildpunkt (Pixel) ein Farbwert zugeordnet. Die optimale Darstellung derartiger Bilder ist in der Auflösung, in der sie erzeugt wurden. Bei einer Reskalierung, insbesondere Vergrößerung, verlieren diese Bilder deutlich an Qualität. Im Unterschied dazu basieren Vektorgrafiken auf einer Bildbeschreibung und lassen sich daher problemlos skalieren. Darüber hinaus benötigen entsprechende Bilder oft deutlich weniger Speicher.

Der Export eines Plots in eine Datei erfolgt immer in den folgenden drei Schritten

1. Öffnen des gewünschten Grafikgerätes (device).
2. Erzeugen des Plots.
3. Beenden des Grafikgerätes (device) mit Hilfe der Funktion `dev.off()`.

Wir erzeugen beispielhaft ein png-Bild.

```

1 ## 1. Öffnen des Grafikgerätes png
2 ## height und width in Pixel
3 png(file = "Beispiel_Bild.png", height = 640, width = 640)
4 ## 2. Erzeugen des Plots
5 barplot(table(ITSDaten$OP), main = "Operationsarten",
6         ylab = "Absolute Häufigkeiten", col = farben)
7 ## 3. Beenden des Grafikgerätes
8 dev.off()

```

Nachdem Sie diesen R Code ausgeführt haben, finden Sie im aktuellen Arbeitsverzeichnis eine Bilddatei mit dem Namen `Beispiel_Bild.png`, welches den erzeugten Plot enthält.

Neben einzelnen Bildern können mit R sogar ganze Filme erstellt werden. Hierfür bieten etwa die Pakete `"animation"` (Xie (2013)) und `"ganimate"` (Pedersen and Robinson (2022)) verschiedene Möglichkeiten und stellen auch einige interessante Beispiele zur Verfügung.

3.3 Diagramme

Das Negativbeispiel aus Abschnitt 3.1 (vgl. Abbildung 3.1) bestätigt die Aussage von Abschnitt 2.5.1, dass Kuchendiagramme nicht die beste Möglichkeit für die Darstellung von Information sind. Versuchen Sie doch einmal die Kategorien in der Abbildung 3.7 der Größe nach zu sortieren oder gar die dargestellten Anteile zu ermitteln. Noch schlimmer wird das Ganze durch Einfügen einer zusätzlichen Dimension

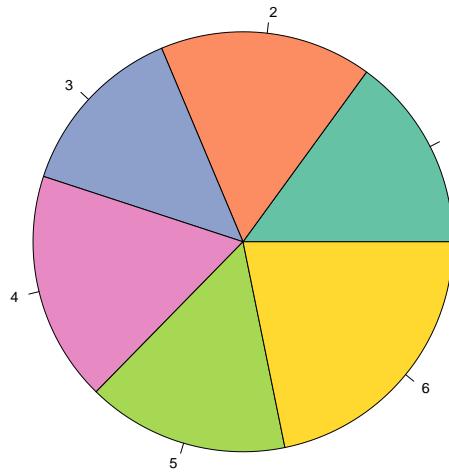


Abbildung 3.7: Sortiere die Kategorien der Größe nach!

in Form eines dreidimensionalen Kuchendiagramms; siehe Abbildung 3.8.

Anmerkung:

Das Problem mit den heute häufig anzutreffenden dreidimensionalen Kuchen- oder Balkendiagrammen ist, dass es durch die zusätzliche dritte Dimension zu einer perspektivischen Verzerrung kommt. Insbesondere widerspricht diese Darstellungsform einer der unten angegebenen Empfehlungen von E. Tufte, da die Anzahl der informationstragenden Dimensionen (= 3) größer ist als die Anzahl der Dimensionen in den Daten (= 2).

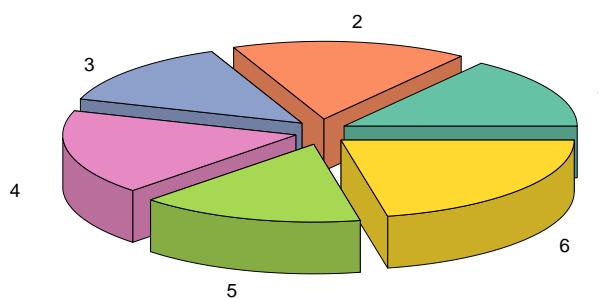


Abbildung 3.8: Noch einmal: Sortiere die Kategorien der Größe nach!

Die Reihenfolge der Kategorien ist für das Auge sofort erkennbar, wenn wir zu einem entsprechenden Balkendiagramm übergehen wie in Abbildung 3.9.

Die unten aufgeführten Empfehlungen für die Erstellung von Diagrammen gehen auf Eduard Tufte zurück (vgl. Globus (1994)):

- Die Zahlen, die man von der Grafik abmessen kann, sollten direkt proportional zu den numerischen Mengen sein, die davon repräsentiert werden.
- Man sollte eine klare, detaillierte und vollständige Beschriftung vornehmen, um eine graphische Verzerrung und Mehrdeutigkeit zu unterbinden.
- Erklärungen der Daten sollten direkt in der Grafik selbst aufgeschrieben sein.
- Wichtige Ereignisse in den Daten sollten beschriftet sein.
- Es ist wichtig, die Variation der Daten und nicht etwa des Designs darzustellen.

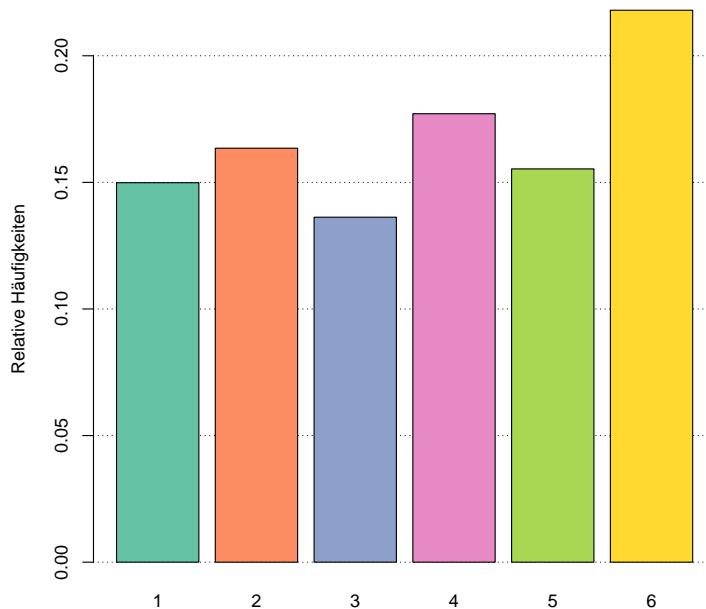
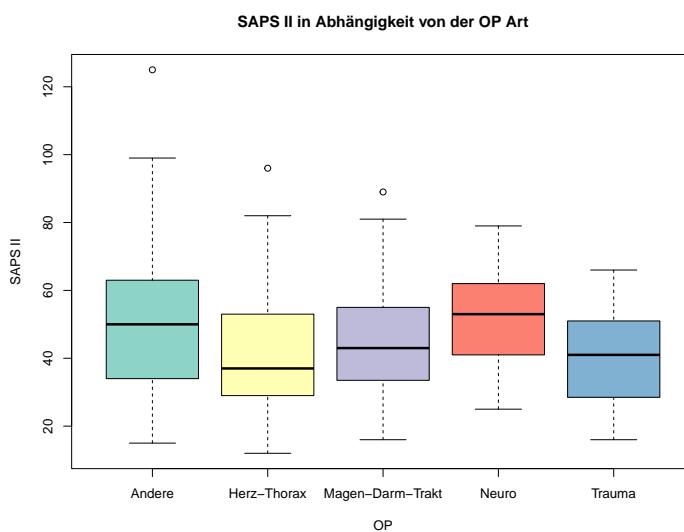


Abbildung 3.9: Und noch einmal: Sortiere die Kategorien der Größe nach!

- Die Anzahl der informationstragenden Dimensionen, die dargestellt sind, sollte die Anzahl der Dimensionen in den Daten nicht überschreiten.
- Grafiken dürfen nie außerhalb ihres Kontexts angegeben werden.

Im verbleibenden Teil dieses Abschnitts stellen wir einige weitere Beispiele für den Einsatz von Diagrammen in Kombination mit Farben dar. Wir beginnen mit einer Darstellung des SAPS II Scores für die verschiedenen OP Arten, wobei wir Box-und-Whisker Plots verwenden. Da für uns keine Anordnung der OP-Arten ersichtlich ist, wählen wir eine qualitative Farbpalette, in diesem Fall Set3 von ColorBrewer (Harrower and Brewer (2003)). Zunächst verwenden wir die Funktion `boxplot`.

```
1 farben <- brewer.pal(n = 5, name = "Set3")
2 boxplot(SAPS.II ~ OP, data = ITSDaten, ylab = "SAPS II",
3          main = "SAPS II in Abhängigkeit von der OP Art", col = farben)
```



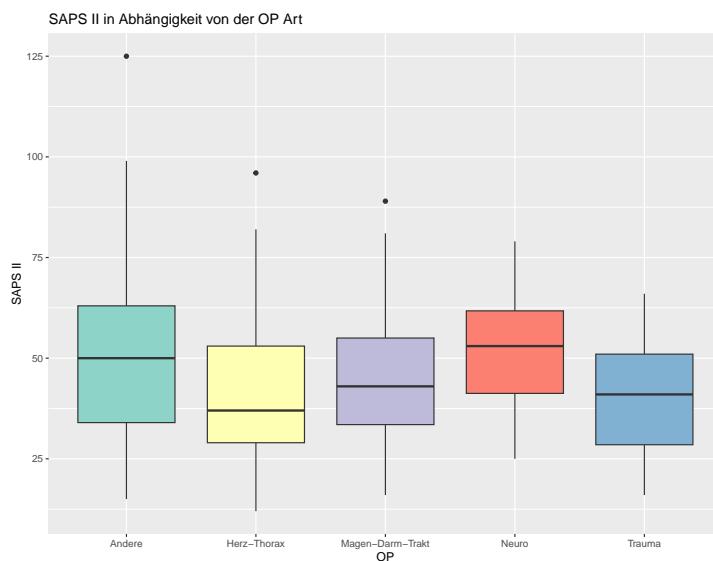
Für die Aufteilung des Scores in die OP-Arten haben wir eine sogenannte Formel verwendet. Der Ausdruck $\text{SAPS.II} \sim \text{OP}$ bedeutet demnach, dass die linke Seite SAPS.II in Abhängigkeit von der rechten Seite OP zu betrachten ist. Nicht überraschend sehen wir die größte Bandbreite bei den anderen OPs und tendenziell recht hohe Werte bei neurologischen OPs.

Wir wiederholen die Darstellung mit Hilfe des Pakets "ggplot2" (Wickham (2009)). Die Farben werden über das Argument `fill` der Funktion `geom_boxplot` gesteuert.

```

1 ## Anlegen der Daten
2 ggplot(ITSData, aes(x = OP, y = SAPS.II)) +
  ## Box-und-Whisker Plot mit den vorgegebenen Farben
4 geom_boxplot(fill = farben) +
5 ## Beschriftung
6 ylab("SAPS II") + ggtitle("SAPS II in Abhängigkeit von der OP Art")

```



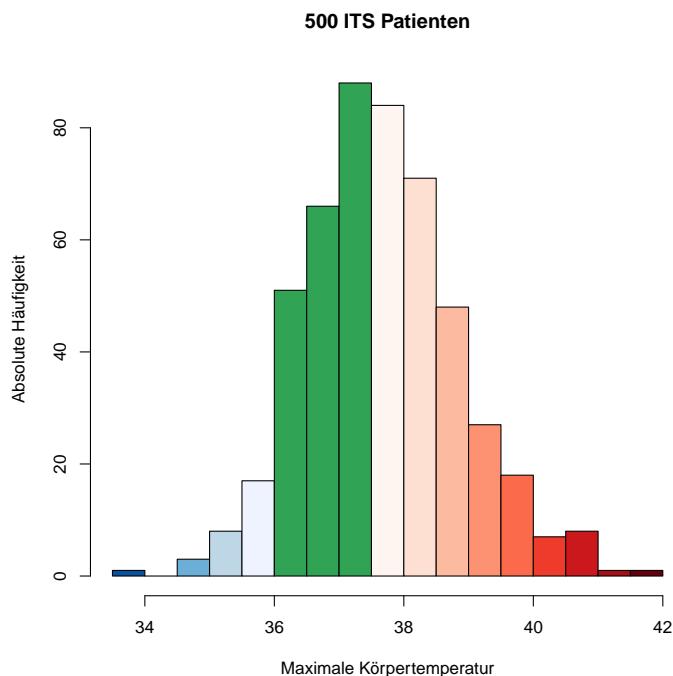
Auch im Fall von Histogrammen bietet sich oft der Einsatz von Farben an. Wir wiederholen das Histogramm der maximalen Körpertemperatur aus Abschnitt 2.6.1. Wir unterteilen die maximale Körper-

temperatur in die folgenden drei Bereiche: $< 36^{\circ}\text{C}$ (zu niedrig), $36 - 37.5^{\circ}\text{C}$ (normal), $> 37.5^{\circ}\text{C}$ (zu hoch). Für den ersten Bereich, der aus fünf Intervallen besteht, verwenden wir die ColorBrewer Palette Blues, die wir mit Hilfe der Funktion `rev` in umgekehrter Reihenfolge erhalten. Für den Normalbereich, der aus drei Intervallen besteht, verwenden wir dreimal die Farbe grün (genauer: #31A354). Das dreimalige Wiederholen erreichen wir durch den Einsatz der Funktion `rep`. Für den dritten Bereich, der aus neun Intervallen besteht, kommen die Rottöne der ColorBrewer Palette Reds zum Einsatz. Zur besseren Übersichtlichkeit entfernen wir außerdem den extremen Wert von Patient 398.

```

1 farben1 <- rev(brewer.pal(5, "Blues"))
2 farben2 <- rep("#31A354", 3)
3 farben3 <- brewer.pal(9, "Reds")
4 hist(ITSDaten$Temperatur[-398], breaks = seq(from = 33.5, to = 42, by = 0.5),
5      main = "500 ITS Patienten", ylab = "Absolute Häufigkeit",
6      xlab = "Maximale Körpertemperatur", col = c(farben1, farben2, farben3))

```

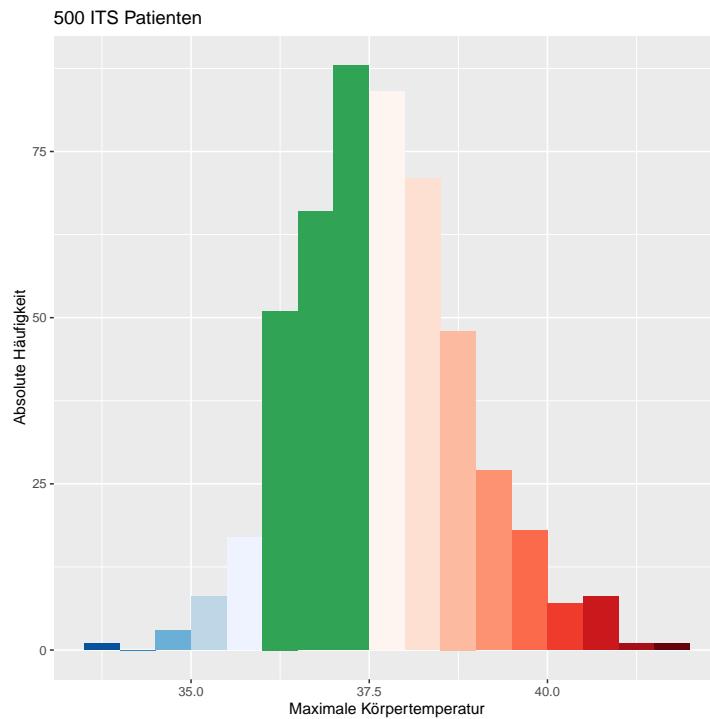


Wir erzeugen eine vergleichbare Abbildung mit Hilfe des Pakets "ggplot2" (Wickham (2009)).

```

1 ggplot(ITSDaten[-398,], aes(x=Temperatur)) +
2   geom_histogram(breaks = seq(from = 33.5, to = 42, by = 0.5),
3                 fill = c(farben1, farben2, farben3)) +
4   ylab("Absolute Häufigkeit") + xlab("Maximale Körpertemperatur") +
5   ggttitle("500 ITS Patienten")

```



Der Einsatz von Farben ist auch bei Streudiagrammen hilfreich und kann zum Beispiel dazu verwendet werden, neben der Variable auf der x- und y-Achse noch eine dritte Variable graphisch darzustellen. Wir verwenden zunächst die Funktion `plot`. Wir legen einen Farbvektor an, der anhand des Geschlechts für jeden Patienten entweder den Eintrag rot (genauer: #E41A1C) für weiblich oder blau (genauer: #377EB8) für männlich hat. Wir beginnen mit einem leeren Vektor, den wir mit Hilfe der Funktion `character` erzeugen. Es handelt sich folglich um einen Vektor, der Buchstaben bzw. Zeichenketten enthält. Mit Hilfe der eckigen Klammern [erhält der Vektor an allen Stellen, an denen das Geschlecht weiblich ist, den Eintrag für rot und an allen Stellen, an denen das Geschlecht männlich ist, den Eintrag für blau. Bei == handelt es sich um einen sogenannten **logischen Operator**, mit dem man feststellen kann, ob Gleichheit vorliegt.

```

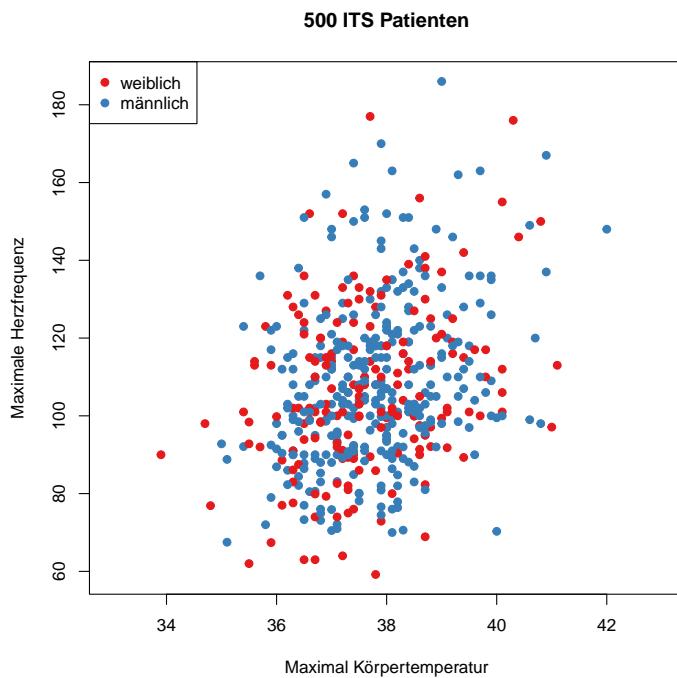
1 ## Anlegen eines leeren Vektors
2 farbenGeschlecht <- character(nrow(ITSDaten))
3 ## Auffüllen mit Farben
4 farbenGeschlecht[ITSDaten$Geschlecht == "weiblich"] <- "#E41A1C"
5 farbenGeschlecht[ITSDaten$Geschlecht == "männlich"] <- "#377EB8"
```

Zur Erinnerung: die in R möglichen Symbole für Punkte (`pch`: point character) sind in der Hilfeseite der Funktion `points` angegeben. Zur besseren Übersichtlichkeit schränken wir die x-Achse zusätzlich auf das Intervall [33, 43] ein. Damit die Farzuordnung zum Geschlecht klar wird, ergänzen wir außerdem mit Hilfe der Funktion `legend` eine Legende.

```

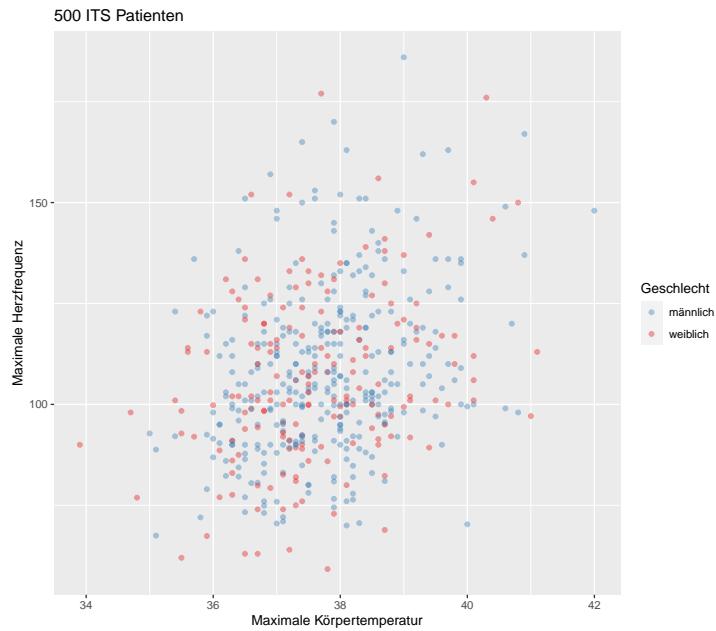
1 plot(x = ITSDaten$Temperatur, y = ITSDaten$Herzfrequenz, pch = 19,
2      xlab = "Maximal Körpertemperatur", ylab = "Maximale Herzfrequenz",
3      main = "500 ITS Patienten",
4      col = farbenGeschlecht, xlim = c(33,43))
```

```
5 legend(x = "topleft", legend = c("weiblich", "männlich"), pch = 19,
6       col = c("#E41A1C", "#377EB8"))
```



Die Werte für Männer und Frauen verteilen sich relativ gleichmäßig über das gesamte Streudiagramm, was darauf hindeutet, dass das Geschlecht keinen Einfluss besitzt. Im Fall des Pakets "ggplot2" (Wickham (2009)) können wir zusätzlich noch sehr einfach das Alpha Blending benutzen. Die Zuweisung der Farbwerte zum Geschlecht ist hier einfacher und kann über die Funktion `scale_colour_manual` durchgeführt werden. Die Reihenfolge der Farben muss dabei zur Reihenfolge der Geschlechter passen. Diese ist standardmäßig alphabetisch; d.h., die erste Farbe wird männlich und die zweite Farbe weiblich zugeordnet.

```
1 ggplot(ITSData[ -398 ,], aes(x=Temperatur, y=Herzfrequenz, colour=Geschlecht)) +
2   ## shape = 19: etwas größerer Punkt
3   ## alpha = 0.4: Stärke des Blendings
4   geom_point(shape=19, alpha=0.4) +
5   ## Werte für die Farbgebung
6   scale_colour_manual(values = c("#377EB8", "#E41A1C")) +
7   ## Beschriftung
8   ggttitle("500 ITS Patienten") + xlab("Maximale Körpertemperatur") +
9   ylab("Maximale Herzfrequenz")
```



3.4 Übungsaufgaben

Verwenden Sie den ITS-Datensatz.

1. Erzeugen Sie ein Balkendiagramm, um die relativen Häufigkeiten für die Variable Ergebnis darzustellen. Verwenden Sie hierfür sowohl die Standardfunktion `barplot` und als auch die entsprechenden Funktionen aus dem Paket "ggplot2" (Wickham (2009)). Als Farbpalette benutzen Sie bitte `Set2` aus dem Paket "RColorBrewer" (Neuwirth (2014)). Speichern Sie die Plots in jpeg-Dateien ab.
2. Erzeugen Sie ein Balkendiagramm, um die relativen Häufigkeiten für die Variable Geschlecht darzustellen. Verwenden Sie hierfür sowohl die Standardfunktion `barplot` und als auch die entsprechenden Funktionen aus dem Paket "ggplot2" (Wickham (2009)). Als Farbpalette benutzen Sie bitte eine geeignete Farbpalette aus dem Paket "RColorBrewer" (Neuwirth (2014)). Speichern Sie die Plots in jpeg-Dateien ab.
3. Erzeugen Sie ein Balkendiagramm, um die relativen Häufigkeiten für die Variable Leberversagen darzustellen. Verwenden Sie hierfür sowohl die Standardfunktion `barplot` und als auch die entsprechenden Funktionen aus dem Paket "ggplot2" (Wickham (2009)). Als Farbpalette benutzen Sie bitte eine geeignete Farbpalette aus dem Paket "ggsci" (Xiao (2018)). Speichern Sie die Plots in jpeg-Dateien ab.
4. Zeichnen Sie Box-und-Whisker Plots für das Alter und unterscheiden Sie dabei zwischen den verschiedenen Merkmalsarten der Variable Ergebnis. Verwenden Sie hierfür sowohl die Standardfunktion `boxplot` als auch die Funktionen aus dem Paket "ggplot2" (Wickham (2009)). Färben Sie die Boxen mit geeigneten Farben. Speichern Sie die Plots in svg-Dateien ab.

5. Zeichnen Sie Box-und-Whisker Plots für die Variable Bilirubin und unterscheiden Sie dabei zwischen den Patienten mit und ohne Leberversagen. Verwenden Sie hierfür sowohl die Standardfunktion `boxplot` als auch die Funktionen aus dem Paket "ggplot2" (Wickham (2009)). Färben Sie die Boxen mit geeigneten Farben. Speichern Sie die Plots in svg-Dateien ab.
6. Zeichnen Sie Box-und-Whisker Plots für die Aufenthaltsdauer (LOS) und unterscheiden Sie dabei weiblichen und männlichen Patienten. Verwenden Sie hierfür sowohl die Standardfunktion `boxplot` als auch die Funktionen aus dem Paket "ggplot2" (Wickham (2009)). Färben Sie die Boxen mit geeigneten Farben. Speichern Sie die Plots in svg-Dateien ab.
7. Erzeugen Sie ein Histogramm für die Variable Herzfrequenz. Betrachten Sie dabei eine Frequenz zwischen 70 und 100 als normal. Verwenden Sie geeignete Farben für das Histogramm und verwenden Sie zum einen die Standardfunktion `hist` und zum anderen die Funktionen aus dem Paket "ggplot2" (Wickham (2009)). Speichern Sie die Plots in png-Dateien ab.
8. Erzeugen Sie ein Histogramm für die Variable Bilirubin. Betrachten Sie dabei eine Konzentration zwischen 1 und 20 als normal. Verwenden Sie geeignete Farben für das Histogramm und verwenden Sie zum einen die Standardfunktion `hist` und zum anderen die Funktionen aus dem Paket "ggplot2" (Wickham (2009)). Speichern Sie die Plots in png-Dateien ab.
9. Erzeugen Sie ein Histogramm/Balkendiagramm für die Variable SAPS-II. Färben Sie die Balken so ein, dass klar wird, dass der Schweregrad der Erkrankung mit steigendem SAPS-II Score zunimmt. Verwenden Sie geeignete Farben für das Histogramm/Balkendiagramm und verwenden Sie zum einen die Standardfunktion `hist` und zum anderen die Funktionen aus dem Paket "ggplot2" (Wickham (2009)). Speichern Sie die Plots in png-Dateien ab.
10. Stellen Sie in einem Streudiagramm die Herzfrequenz in Abhängigkeit vom Alter dar und kennzeichnen Sie zusätzlich Männer und Frauen farblich. Verwenden sowohl die Standardfunktion `plot` als auch die Funktionen aus dem Paket "ggplot2" (Wickham (2009)). Speichern Sie die Plots in pdf-Dateien ab.
11. Stellen Sie in einem Streudiagramm die Bilirubinkonzentration in Abhängigkeit vom Alter dar und kennzeichnen Sie zusätzlich Patienten mit und ohne Leberversagen farblich. Verwenden sowohl die Standardfunktion `plot` als auch die Funktionen aus dem Paket "ggplot2" (Wickham (2009)). Speichern Sie die Plots in pdf-Dateien ab.
12. Stellen Sie in einem Streudiagramm die Herzfrequenz in Abhängigkeit von der Körpertemperatur dar und kennzeichnen Sie zusätzlich das Outcome (Variable Ergebnis) der Patienten farblich. Verwenden sowohl die Standardfunktion `plot` als auch die Funktionen aus dem Paket "ggplot2" (Wickham (2009)). Speichern Sie die Plots in pdf-Dateien ab.

4 Wahrscheinlichkeitsverteilungen

Für den Rückschluss von einer Stichprobe auf die zugrunde liegende Population benötigen wir Modelle aus der Wahrscheinlichkeitstheorie (vgl. Abschnitt 2.1). Die Grundlage dieser Modelle sind Wahrscheinlichkeitsverteilungen, wobei im einfachsten Fall die Wahrscheinlichkeitsverteilungen selbst bereits die Modelle sind, die man untersucht. Das Ziel der induktiven (parametrischen) Statistik ist es in diesem Fall, mit Hilfe der vorliegenden Daten auf die unbekannten Parameter der angenommenen Wahrscheinlichkeitsverteilung zu schließen.

In diesem Kapitel lernen wir die Wahrscheinlichkeits-, Verteilungs- und Quantilfunktionen von diskreten und (absolut) stetigen Wahrscheinlichkeitsverteilungen kennen. Im einzelnen werden die folgenden Wahrscheinlichkeitsverteilungen behandelt:

- Bernoulli-Verteilung $\text{Bernoulli}(p)$
- Binomialverteilung $\text{Binom}(m, p)$
- Hypergeometrische Verteilung $\text{Hyper}(m, n, k)$
- Negative Binomialverteilung $\text{Nbinom}(r, p)$
Spezialfälle: Pascal-Verteilung, Pólya-Verteilung, geometrische Verteilung
- Poisson-Verteilung $\text{Pois}(\lambda)$
- Normalverteilung $\text{Norm}(\mu, \sigma^2)$
- log-Normalverteilung $\text{Lnorm}(\mu, \sigma)$
- Gamma-Verteilung $\text{Gamma}(\sigma, \alpha)$
Spezialfälle: Exponentialverteilung, Erlang-Verteilung, χ^2 -Verteilung
- Weibull-Verteilung $\text{Weibull}(\sigma, \alpha)$
- Verteilungen, die im Zusammenhang mit der Normalverteilung entstehen: χ^2 -Verteilung $\text{Chisq}(n)$, t-Verteilung $t(n)$, F-Verteilung $F(m, n)$

Den R Code für dieses Kapitel finden Sie in der Datei `WSVerteilungen.Rmd`, die Sie von meinem GitHub-Account herunterladen können (Link: <https://github.com/stamats/ESDR/blob/master/WSVerteilungen.Rmd>). Klicken Sie mit der rechten Maustaste auf `Raw`. Dann können Sie das Ziel speichern unter.... Am wenigsten Schwierigkeiten ergeben sich, wenn Sie meine R Markdown Dateien im selben Ordner wie die Daten, welche in den jeweiligen Kapiteln verwendet werden, speichern.

Wir installieren zunächst die in diesem Kapitel benötigten Pakete. Wir beginnen mit dem Paket "distr" (Ruckdeschel et al. (2006)).

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

Ein weiteres interessantes Paket ist das Paket "distr6" (Sonabend and Kiraly (2022)), welches sich leider nicht (mehr) auf CRAN befindet. Wir können aber den üblichen Installationsweg wählen, indem wir das Repository von "distr6" zu den eingestellten Repositorien hinzufügen.

```
1 # Füge Repository raphaelsl hinzu
2 options(repos = c(raphaelsl = "https://raphaelsl.r-universe.dev",
3                  CRAN = "https://cloud.r-project.org"))
4 # Installiere distr6
5 install.packages("distr6")
```

Alternativ können wir das Paket direkt von GitHub installieren, indem wir das Paket "remotes" (Csárdi et al. (2021)) verwenden.

```
1 install.packages("remotes")
2 remotes::install_github("alan-turing-institute/distr6",
3                         upgrade = "never", build = FALSE)
```

Durch `remotes::install_github` wird die Funktion `install_github` aus dem Paket "remotes" aufgerufen, ohne das Paket "remotes" explizit zu laden.

Achten Sie darauf, dass Sie die Pakete der vorhergehenden Kapitel 2 und 3 bereits installiert haben. Wir laden alle Pakete, die wir in diesem Kapitel verwenden werden.

```
1 library(distr)
2 library(distr6)
```

Wie bereits in Abschnitt 2.4 erklärt, ist ein wiederholtes Ausführen von `library` unproblematisch.

4.1 Diskrete Verteilungen

Wir betrachten eine Funktion X , die ihre Werte im Raum der natürlichen Zahlen mit gewissen Wahrscheinlichkeiten annimmt. Eine derartige Funktion X nennt man auch eine **diskrete Zufallsvariable**. Die Werte einer Zufallsvariable heißen **Realisationen**.

Wir können die **diskrete Wahrscheinlichkeitsverteilung** oder kurz die **diskrete Verteilung** der Werte von X eindeutig beschreiben, indem wir zu jedem möglichen Wert $k \in \mathbb{N}$ von X die zugehörige **Wahrscheinlichkeit** $P(X = k)$ angeben. Die Funktion

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

wird die **Wahrscheinlichkeitsfunktion** von X genannt. Die Funktion

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

heißt die **Verteilungsfunktion** von X . Die Umkehrung der Verteilungsfunktion führt uns zur **Quantilfunktion**

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

Wichtige Kenngrößen einer Verteilung, die auch zu deren Charakterisierung herangezogen werden können, sind der **Erwartungswert** und die **Varianz**. Der Erwartungswert von X , kurz $E(X)$, ist der Wert, den wir von X im Mittel erwarten. Es gilt

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

d.h., die möglichen Ausprägungen von X werden mit ihren Wahrscheinlichkeiten multipliziert und aufsummiert. Die Varianz von X , kurz $\text{Var}(X)$, erhält man als den Erwartungswert der quadratischen Abweichungen vom Erwartungswert

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

Häufig betrachtet man auch die Quadratwurzel der Varianz, welche die **Standardabweichung** von X genannt wird, kurz $\sigma_X = \sqrt{\text{Var}(X)}$.

In diesem Abschnitt werden wir einige wichtige diskrete Verteilungen kennenlernen.

Bernoulli-Verteilung

Die einfachste diskrete Verteilung ist die sogenannte Bernoulli-Verteilung, für die im folgenden Beispiel zwei Anwendungsfälle skizziert werden.

Beispiel 4.1. (a) Wir betrachten eine Produktion von Glühbirnen, bei der die Birnen in 1% der Fälle defekt sind. D.h., wir können den Produktionsprozess durch eine diskrete Zufallsvariable X beschreiben, welche die Werte 0 = defekt und 1 = nicht defekt annehmen kann. Dies führt uns auf die folgende Wahrscheinlichkeitsfunktion

$$P(X = 0) = 0.01 \quad \text{und} \quad P(X = 1) = 1 - 0.01 = 0.99 \quad (4.6)$$

(b) In einer randomisierten und kontrollierten klinischen Studie werden zwei Interventionen miteinander verglichen. Hierfür werden 65% der Patienten zufällig der Intervention I zugeordnet und entsprechend 35% der Intervention II. Dieser Vorgang kann durch die diskrete Zufallsvariable X beschrieben werden, die den Wert 0 = Intervention I mit Wahrscheinlichkeit 65% und den Wert 1 = Intervention II mit Wahrscheinlichkeit 35% annimmt. Es ergibt sich die folgende Wahrscheinlichkeitsfunktion

$$P(X = 0) = 0.65 \quad \text{und} \quad P(X = 1) = 1 - 0.65 = 0.35 \quad (4.7)$$

Den beiden vorangegangenen Beispielen liegt die folgende Wahrscheinlichkeitsverteilung zugrunde.

Definition 4.2 (Bernoulli-Verteilung). Gegeben sei eine diskrete Zufallsvariable X , welche nur die Werte 0 und 1 annehmen kann. Dann kann die Wahrscheinlichkeitsfunktion der Verteilung von X geschrieben werden als

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

wobei $p \in [0, 1]$. Diese Verteilung heißt **Bernoulli-Verteilung** mit Parameter p , kurz: $X \sim \text{Bernoulli}(p)$.

Binomialverteilung

Die Verallgemeinerung der Bernoulli Verteilung ist die sogenannte Binomialverteilung. Das folgende Beispiel zeigt zwei mögliche Anwendungsfälle.

Beispiel 4.3. (a) Wir betrachten wieder die Produktion von Glühbirnen, wobei 1% der Birnen defekt sind und wir die Qualität der letzten Charge überprüfen wollen. Dazu ziehen wir zufällig **mit Zurücklegen** eine Stichprobe von $m = 20$ Glühbirnen aus dieser Charge (=Population). Die Funktion X sei die Zufallsvariable, welche die Anzahl der defekten Glühbirnen beschreibt. Mit Hilfe der Verteilung von X können wir dann zum Beispiel angeben, wie wahrscheinlich es ist, dass genau eine defekte Glühbirne gezogen wird, nämlich

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

Es gibt wegen der 20 Züge, 20 Möglichkeiten die eine defekte Glühbirne zu ziehen, wobei dies mit der Wahrscheinlichkeit 0.01 geschieht. Im Fall der anderen 19 Züge wird eine funktionierende Glühbirne jeweils mit Wahrscheinlichkeit 0.99 gezogen. Aufgrund der (stochastischen) Unabhängigkeit der Züge werden die Wahrscheinlichkeiten miteinander multipliziert.

(b) Die Prävalenz von Diabetes betrug 2014 bei den Erwachsenen weltweit ca. 9% (WHO (2015b)). Unter **Prävalenz** versteht man den Anteil der zum Untersuchungszeitpunkt kranken Personen. Wir führen eine Studie durch und ziehen zufällig **mit Zurücklegen** eine Stichprobe von 50 Personen. Die Anzahl der Personen mit Diabetes in unserer Stichprobe wird durch die Zufallsvariable X gegeben. Wie wahrscheinlich ist es, dass unsere Stichprobe mindestens zwei Personen mit Diabetes enthält? In diesem Fall ist es einfacher sich das sogenannte **Gegenereignis** anzusehen: Die Stichprobe enthält keine oder genau eine Person mit Diabetes. Wir erhalten

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

Die gesuchte Wahrscheinlichkeit ist demnach

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

Im Allgemeinen erhalten wir die folgende diskrete Wahrscheinlichkeitsverteilung.

Definition 4.4 (Binomialverteilung). Gegeben sei eine Schachtel (Urne) mit schwarzen und weißen Bällen, wobei der Anteil der weißen Bälle gleich $p \in [0, 1]$ ist. Wir ziehen zufällig m -mal ($m \in \mathbb{N}$) mit Zurücklegen aus dieser Schachtel und beschreiben mit der Zufallsvariable X die Anzahl $k \in \{1, 2, \dots, n\}$ der gezogenen weißen Bälle. Dann lautet die Wahrscheinlichkeitsfunktion von X

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

wobei

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

der Binomialkoeffizient ist und ! Fakultäten anzeigen. Diese Verteilung heißt **Binomialverteilung** mit Parametern m und p , kurz: $X \sim \text{Binom}(m, p)$.

Wir geben einige zusätzliche Erläuterungen.

Bemerkung 4.5. (a) Die Fakultät von $k \in \mathbb{N}$ ist definiert als

$$k! = \begin{cases} 1 & \text{falls } k = 0 \\ 1 \cdot 2 \cdot \dots \cdot k & \text{falls } k \geq 1 \end{cases} \quad (4.14)$$

(b) Ein Blick auf die Wahrscheinlichkeitsfunktionen der Bernoulli- und Binomialverteilung zeigt, dass $\text{Bernoulli}(p) = \text{Binom}(1, p)$.

(c) Als Erwartungswert und Varianz von $\text{Binom}(m, p)$ erhält man

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

In R stehen für viele diskrete Wahrscheinlichkeitsverteilungen die Wahrscheinlichkeits-, Verteilungs- und Quantilfunktion zur Verfügung. Generell setzen sich die Namen dieser grundlegenden Funktionen in R immer aus einem Präfix und einem Kürzel für die Wahrscheinlichkeitsverteilung zusammen. Die möglichen Präfixe sind

d: Wahrscheinlichkeitsfunktion

p: Verteilungsfunktion

q: Quantilfunktion

r: Funktion zur Erzeugung von (Pseudo-)Zufallszahlen

Es genügt demnach das Kürzel für die Verteilung zu kennen, um die Funktionen anwenden zu können.

Im Fall der Binomialverteilung lautet das Kürzel `binom`, und die Funktionen entsprechend: `dbinom`, `pbinom`, `qbinom` und `rbinom`. Die Parameter m und p der Binomialverteilung heißen in R `size` und `prob`.

Wir berechnen die Wahrscheinlichkeiten aus Beispiel 4.3 mit Hilfe von R.

```
1 ## a) Genau eine defekte Glühbirne
2 dbinom(1, size = 20, prob = 0.01)
```

```
[1] 0.1652337
```

```
1 ## b) Keine Person mit Diabetes
2 dbinom(0, size = 50, prob = 0.09)
```

```
[1] 0.008955083
```

```
1 ## b) Genau eine Person mit Diabetes
2 dbinom(1, size = 50, prob = 0.09)
```

```
[1] 0.04428338
```

Für die Bestimmung der Wahrscheinlichkeit, dass mindestens zwei Personen mit Diabetes gezogen werden, ist es einfacher die Verteilungsfunktion zu verwenden.

```
1 ## b) Mindestens zwei Personen mit Diabetes: 1 - P(X ≤ 1)
2 1 - pbinom(1, size = 50, prob = 0.09)
```

```
[1] 0.9467615
```

Alternativ und numerisch noch etwas exakter erhalten wir diese Wahrscheinlichkeit auch, indem wir das Argument `lower.tail = FALSE` verwenden. Dann wird anstelle von $P(X \leq k)$ die Wahrscheinlichkeit $P(X > k)$ berechnet. Im vorliegenden Beispiel gilt $P(X \geq 2) = P(X > 1)$ und wir erhalten.

```
1 ## b) Mindestens zwei Personen mit Diabetes: P(X > 1)
2 pbinom(1, size = 50, prob = 0.09, lower.tail = FALSE)
```

```
[1] 0.9467615
```

Mit Hilfe der Quantilfunktion können wir zum Beispiel berechnen, wie viele defekte Glühbirnen wir mit einer Wahrscheinlichkeit von **mindestens 99% höchstens** erwarten können.

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

```
[1] 2
```

Sollten folglich im Rahmen der Qualitätskontrolle mehr als zwei defekte Glühbirnen gezogen werden, so könnte das auf ein Qualitätsproblem – eine größere Defektwahrscheinlichkeit – hindeuten. Denn bei einer Defektwahrscheinlichkeit von 1% ist das Ziehen von drei oder mehr defekten Glühbirnen sehr unwahrscheinlich (< 1%). Die Wahrscheinlichkeit liegt tatsächlich bei nur ca. 0.1%.

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

```
[1] 0.001003576
```

Die Funktion `rbinom` können wir verwenden, um Zufallszahlen zu erzeugen. Wenden wir dies in unserem Diabetesbeispiel an, so steht jede Zufallszahl für eine Studie, genauer für die Anzahl der Personen mit Diabetes in dieser Studie. Mit dem folgenden R Code, simulieren wir demnach zehn Studien.

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

```
[1] 3 4 2 4 3 8 4 3 9 3
```

Anmerkung:

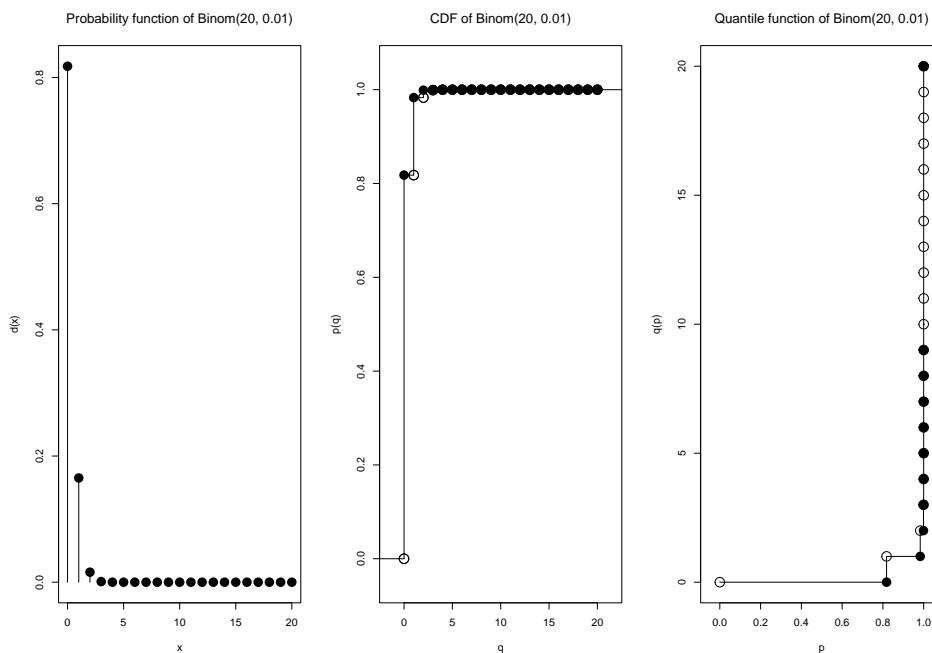
Durch die gewachsene Leistung der Computer haben Simulationen in den letzten Jahrzehnten enorm an Bedeutung gewonnen. Man kann heute immer komplexere Wahrscheinlichkeitsmodelle ztausende Male simulieren. Simulationen werden z.B. für die Fallzahlplanung klinischer Studien, die Wettervorhersage oder auch zur Vorhersage des Verlaufs der COVID19-Pandemie eingesetzt.

Wir können die Wahrscheinlichkeits-, Verteilungs- und Quantilfunktion der Binomialverteilung auch graphisch darstellen. Hierfür bietet sich die Verwendung des Paketes "distr" (Ruckdeschel et al. (2006)) an, welches eine objektorientierte (S4) Implementation von Wahrscheinlichkeitsverteilungen enthält. Wir laden das Paket und erzeugen mit Hilfe der Funktion `Binom` eine Zufallsvariable X mit der Verteilung $\text{Binom}(20, 0.01)$, was unserem Glühbirnenbeispiel entspricht.

```
1 X <- Binom(size = 20, prob = 0.01)
```

Mit Hilfe der Funktion `plot` können die Wahrscheinlichkeits- (PDF = probability density/mass function), die Verteilungs- (CDF = cumulative distribution function) und die Quantilfunktion (quantile function) einer gegebenen Zufallsvariable graphisch dargestellt werden.

```
1 plot(X)
```

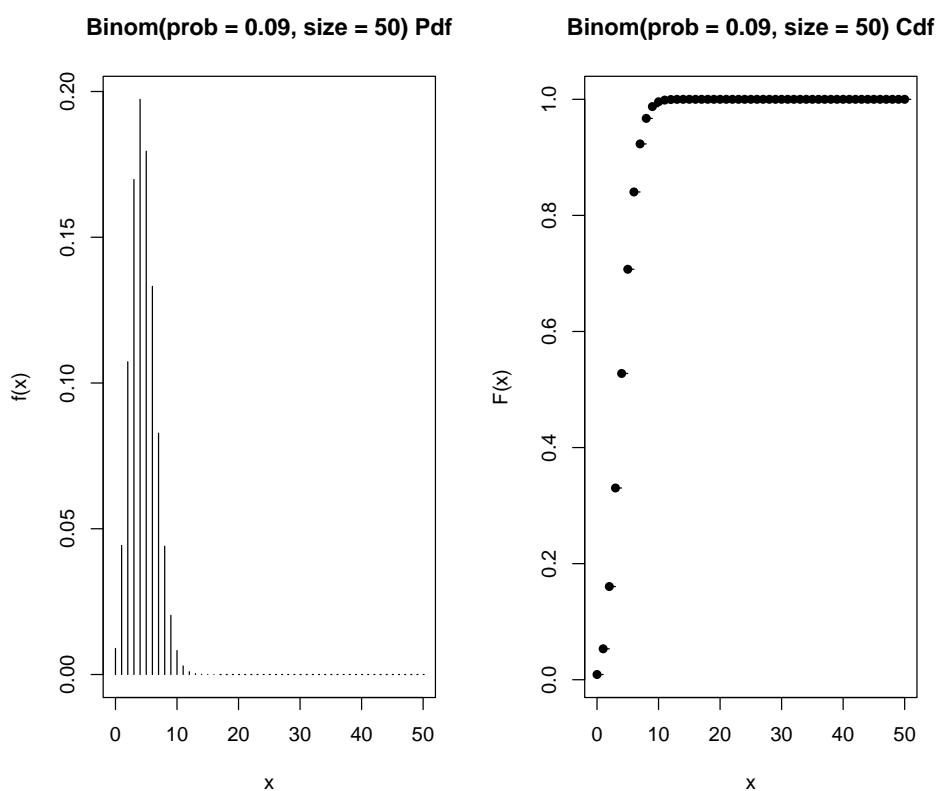


Das Paket "distr6" (Sonabend and Kiraly (2022)) kann in einem gewissen Sinn als eine Weiterentwicklung des Paketes "distr" angesehen werden. Es verwendet anstelle der sogenannten S4-Objektorientierung die sogenannte R6-Objektorientierung. Wir laden das Paket und legen eine Zufallsvariable X_6 mit der Verteilung $\text{Binom}(50, 0.09)$ an, was unserem Diabetesbeispiel entspricht.

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

Mit Hilfe der Funktion `plot` können in diesem Fall die Wahrscheinlichkeits- (PDF) und die Verteilungsfunktion (CDF) einer gegebenen Zufallsvariable graphisch dargestellt werden.

```
1 plot(X6)
```



Mit Hilfe der Funktion `summary` (genauer der entsprechenden Methode für `distr6`-Objekte) bekommt man einen Überblick über die wichtigsten Eigenschaften der entsprechenden Zufallsvariable.

```
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: NO
Traits: discrete; univariate
Properties: asymmetric; leptokurtic; positive skew

```

Hypergeometrische Verteilung

Führt man anstelle des Ziehens mit Zurücklegen ein Ziehen ohne Zurücklegen durch, führt dies auf die hypergeometrische Verteilung, die wir im Folgenden einführen werden. Das folgende Beispiel ist sehr ähnlich zu Beispiel 4.3.

Beispiel 4.6. (a) Wir betrachten eine Schachtel (= Population) mit $m + n = 500$ Glühbirnen, von denen $m = 5$ defekt sind. Wir ziehen zufällig **ohne** Zurücklegen eine Stichprobe von $k = 20$ Glühbirnen aus der Schachtel. Die Funktion X sei die Zufallsvariable, welche die Anzahl der defekten Glühbirnen in unserer Stichprobe beschreibt. Mit Hilfe der Verteilung von X können wir dann zum Beispiel angeben, wie wahrscheinlich es ist, dass keine defekte Glühbirne gezogen wird, nämlich

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

Es gibt 20 Züge, in denen jeweils eine funktionierende Glühbirne gezogen und beiseite gelegt wird, womit sich Zähler und Nenner nach jedem Zug um eins reduzieren; d.h., die Anteile in der Population ändern sich mit jedem Zug.

(b) Die Bevölkerungszahl in Andorra lag im Jahr 2000 bei ca. 66000 (Wikipedia contributors (2022a)), wobei ca. 6000 Einwohner (WHO (2015a)) einen Diabetes hatten. Wir führen in Andorra eine Studie durch und ziehen zufällig **ohne** Zurücklegen 50 Einwohner. Die Anzahl der Personen mit Diabetes in unserer Stichprobe wird durch die Zufallsvariable X beschrieben. Wie wahrscheinlich ist es, dass unsere Stichprobe mindestens eine Person mit Diabetes enthält? Wir betrachten das Gegenereignis: Die Stichprobe enthält keine Person mit Diabetes und erhalten

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

Die gesuchte Wahrscheinlichkeit ist demnach

$$P(X \geq 1) = 1 - P(X = 0) = 1 - 0.009 = 99.1\% \quad (4.18)$$

Wir geben die Definition der hypergeometrischen Verteilung an.

Definition 4.7 (Hypergeometrische Verteilung). *Wir betrachten eine Schachtel (Urne) mit $m \in \mathbb{N}$ weißen und $n \in \mathbb{N}$ schwarzen Kugeln und ziehen zufällig $k \in \mathbb{N}$ Kugeln ohne Zurücklegen ($k < m + n$). Die Zufallsvariable X , welche die Anzahl j der weißen Kugeln in der Stichprobe angibt ($j \leq m$), besitzt die folgende Wahrscheinlichkeitsfunktion*

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

Diese Verteilung heißt **hypergeometrische Verteilung** mit Parametern m , n und k , abgekürzt: $X \sim \text{Hyper}(m, n, k)$.

Wir geben einige zusätzliche Erläuterungen an.

Bemerkung 4.8. (a) Für große Populationen und Stichproben ist die Berechnung der Binomialkoeffizienten in der Definition der hypergeometrischen Verteilung schwierig. Das liegt daran, dass Fakultäten von großen Zahlen zu berechnen sind, wobei die Fakultät proportional zur Exponentialfunktion wächst. (b) Die hypergeometrische Verteilung ist in der Praxis nur selten verwendbar, da man hierfür eine genaue Kenntnis der Größe der Population und der Zusammensetzung der Population benötigt. Beides ist in den allermeisten Fällen unbekannt. Jedoch ist bereits ab einer moderaten Größe der Population und falls die Stichprobe im Verhältnis zur Population nicht zu groß ist, der Unterschied zwischen hypergeometrischer und Binomialverteilung sehr klein. In diesem Fall passiert es nämlich nur mit einer kleinen Wahrscheinlichkeit, dass beim Ziehen mit Zurücklegen die gleiche Kugel mehrfach gezogen wird. Entsprechend kann in praktischen Anwendungen in den allermeisten Fällen die Binomialverteilung als eine Näherung (Approximation) verwendet werden.

(c) Als Erwartungswert und Varianz von $\text{Hyper}(m, n, k)$ erhält man

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

In diesen Formeln lässt sich eine gewisse Analogie zur Binomialverteilung erkennen. Der Faktor $\frac{m+n-k}{m+n-1}$, der den wesentlichen Unterschied zur Binomialverteilung ausmacht und den wir in Beispiel 5.13 nochmals kennenlernen werden, heißt auch **Endlichkeitskorrektur**.

Die hypergeometrische Verteilung wird in R mit `hyper` abgekürzt. Dies führt auf die Funktionen `dhyper`, `phyper`, `qhyper` und `rhyper`. Wir berechnen die Wahrscheinlichkeiten aus dem Beispiel 4.6 mit R.

```
1 ## a) keine defekte Glühbirne
2 dhyper(0, m = 5, n = 495, k = 20)
```

```
[1] 0.8146893
```

```
1 ## b) keine Person mit Diabetes
2 dhyper(0, m=6000, n=60000, k = 50)
```

```
[1] 0.008502747
```

Die Wahrscheinlichkeit von mindestens einer Person mit Diabetes können wir auch direkt mit `phyper` unter Verwendung des Arguments `lower.tail = FALSE` berechnen.

```
1 ## b) mind. eine Person mit Diabetes
2 phyper(0, m=6000, n=60000, k=50, lower.tail = FALSE)
```

```
[1] 0.9914973
```

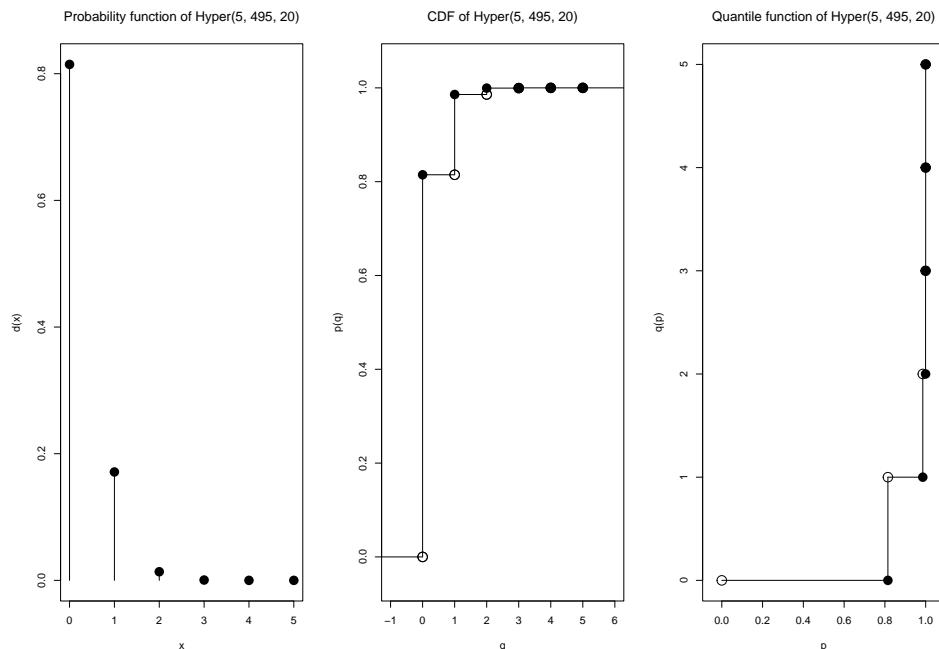
Wir simulieren 10 Stichproben der Größe 50 aus dem Diabetesbeispiel.

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

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

Jeder der Zahlen steht also für die Anzahl der Diabetespatienten für den Fall, dass man 50 Personen zufällig auswählt. Wir stellen die Verteilung $\text{Hyper}(5, 495, 20)$ aus dem Glühbirnenbeispiel mit Hilfe des Pakets "distr" (Ruckdeschel et al. (2006)) graphisch dar.

```
1 X <- Hyper(m=5, n=495, k=20)
2 plot(X)
```



Außerdem verwenden wir die Funktion `summary` des Paketes "distr6" (Sonabend and Kiraly (2022)), um die Verteilung des Glühbirnenbeispiels zusammenzufassen.

```
1 X6 <- Hypergeometric$new(size = 500, successes = 5, draws = 20)
2 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: NO
Traits: discrete; univariate
Properties: asymmetric; leptokurtic; positive skew

```

Wie der folgende Vergleich zeigt, liefern die hypergeometrische und die Binomialverteilung in unserem Glühbirnenbeispiel trotz der recht kleinen Population bereits recht ähnliche Wahrscheinlichkeitswerte.

```

1 ## Wahrscheinlichkeit für 0, 1, 2, 3 defekte Glühbirnen
2 ## mit Zurücklegen
3 round(dbinom(0:3, size=20, prob=0.01), 4)

```

```
[1] 0.8179 0.1652 0.0159 0.0010
```

```

1 ## ohne Zurücklegen
2 round(dhyper(0:3, m=5, n=495, k=20), 4)

```

```
[1] 0.8147 0.1712 0.0136 0.0005
```

Wir haben hierbei den Operator : verwendet, mit dem wir schnell Zahlenfolgen ganzer Zahlen erzeugen können; d.h., 0:3 erzeugt den Vektor mit den Zahlen 0, 1, 2, 3. Die d, p und q Funktionen können nicht nur einzelne Zahlen, sondern auch Zahlenvektoren verarbeiten.

```
1 0:3
```

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

Negative Binomialverteilung

Eine weitere wichtige diskrete Verteilung, die in gewisser Weise mit der Binomialverteilung zusammenhängt, ist die negative Binomialverteilung. Wir beginnen mit einem einführenden Beispiel mit möglichen Anwendungen dieser Verteilung.

Beispiel 4.9. (a) Wir betrachten wieder die Glühbirnenproduktion mit einer Ausschusswahrscheinlichkeit von 1%. Die Zufallsvariable X beschreibt die Anzahl funktionierenden Glühbirnen, die man (mit Zurücklegen) zieht bis man die erste defekte Glühbirne erhält. Wie wahrscheinlich ist es, dass genau die 20. Glühbirne die erste defekte Glühbirne ist? D.h., wir müssen 19 funktionierende Glühbirnen lang warten und erhalten

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

(b) Die Prävalenz (Krankheitshäufigkeit) von Diabetes betrug 2014 bei den Erwachsenen weltweit ca. 9% (WHO (2015b)). Wir führen eine Studie durch und ziehen (mit Zurücklegen) eine Stichprobe von 250 Personen. Damit unsere Studie die notwendige Aussagekraft (Power) hat, benötigen wir wenigstens 20 Personen mit Diabetes. Wie groß ist die Wahrscheinlichkeit, dass wir spätestens mit Einschluss der 250.

Person die notwendige Anzahl von Diabetespatienten erreichen? D.h., dass wir maximal 230 Personen ohne Diabetes ziehen müssen. Die Antwort lautet, wie wir in Kürze sehen werden,

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

D.h., wir können mit ca. 74% Wahrscheinlichkeit davon ausgehen, dass uns der Einschluss von 20 Diabetespatienten gelingen wird.

Wir geben die Definition der negativen Binomialverteilung an.

Definition 4.10 (Negative Binomialverteilung). *Gegeben sei eine Schachtel (Urne) mit schwarzen und weißen Bällen, wobei der Anteil der weißen Bälle gleich $p \in [0, 1]$ ist. Wir ziehen solange zufällig mit Zurücklegen aus der Schachtel, bis wir $r \in \mathbb{N}$ weiße Bälle erhalten haben. Die Zufallsvariable X beschreibt die Anzahl $k \in \mathbb{N}_0 = \{0, 1, 2, \dots\}$ der schwarzen Bälle, die man erhält bis zum ersten Mal r weiße Bälle gezogen wurden. Die Wahrscheinlichkeitsfunktion von X ist*

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

Diese Verteilung heißt **negative Binomialverteilung** mit Parametern r und p , kurz: $X \sim Nbinom(r, p)$.

Wir geben einige zusätzliche Erläuterung.

Bemerkung 4.11. (a) Die negative Binomialverteilung ist eine sogenannte **Wartezeitverteilung**. Wir können damit angeben, wie viele Fehlversuche oder ereignislose Zeitintervalle wir abwarten müssen, bis die gewünschte Anzahl von Erfolgen oder Ereignissen eingetreten ist.

(b) Man kann die negative Binomialverteilung auch noch allgemeiner definieren und den Parameter r von den natürlichen Zahlen auf die positiven reellen Zahlen erweitern.

(c) Die negative Binomialverteilung wird manchmal auch **Pascal-Verteilung** oder **Pólya-Verteilung** genannt. Dies geschieht vor allem dann, wenn man auf den Wertebereich von r hinweisen möchte. So wird der Name Pascal-Verteilung verwendet, falls $r \in \mathbb{N}$ und der Name Pólya-Verteilung, falls $r \in (0, \infty)$. Im Fall $r = 1$ spricht man auch von der **geometrischen Verteilung**.

(d) Als Erwartungswert und Varianz von $Nbinom(r, p)$ ergeben sich

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

In R wird die negative Binomialverteilung durch `nbinom` abgekürzt und die Parameter heißen `size` und `prob` wie im Fall der Binomialverteilung. Wir erhalten demnach die Funktionen `dnb`, `pnbinom`, `qnbinom` und `rnb`. Wir berechnen die Wahrscheinlichkeiten aus dem obigen Beispiel.

```
1 ## a) 20. Glühbirne = 1. defekte Glühbirne
2 dnb(19, size = 1, prob = 0.01)
```

```
[1] 0.008261686
```

```
1 ## b) Max. 250 Personen für 20 Personen mit Diabetes
2 pnbinom(230, size = 20, prob = 0.09)
```

```
[1] 0.7407983
```

Für das Diabetesbeispiel könnten wir auch die Quantilfunktion heranziehen. D.h., wir legen die Größe der Stichprobe so fest, dass wir mit einer vorgegebenen (hohen) Wahrscheinlichkeit unser Ziel von 20 Diabetespatienten erreichen. In solchen Fällen verwendet man meist 90%, 95% oder auch 99%. Für eine Sicherheit von **mindestens** 95% erhalten wir

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

```
[1] 286
```

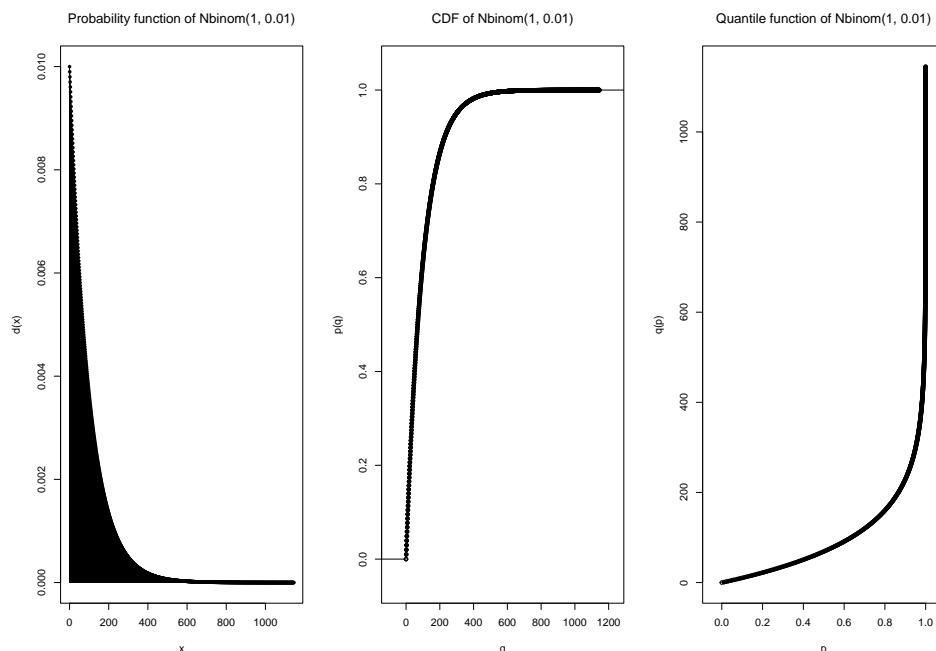
Dies ist die Anzahl der Personen ohne Diabetes; d.h., wir sollten insgesamt 306 Personen zufällig auswählen. Wir simulieren 10 Diabetesstudien.

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

```
[1] 205 222 192 230 159 289 187 239 150 205
```

Jede der obigen Zahlen gibt an, wie viele Personen ohne Diabetes gezogen wurden, bis man die notwendige Anzahl von 20 Personen mit Diabetes erreicht hatte. Wir stellen die negative Binomialverteilung aus unserem Glühbirnenbeispiel mit Hilfe des Paketes "distr" (Ruckdeschel et al. (2006)) graphisch dar.

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



Durch den Parameter `cex.points` reduzieren wir die Größe der angezeigten Punkte etwas. Außerdem fassen wir die Verteilung mit Hilfe der Funktion `summary` des Paketes "distr6" (Sonabend and Kiraly (2022)) zusammen.

```
1 X6 ← NegativeBinomial$new(size = 1, prob = 0.01)
2 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: NO   Scientific Type: NO

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

Poisson-Verteilung

Als letzte diskrete Verteilung werden wir im Folgenden die Poisson-Verteilung vorstellen, die vielfältige Anwendungen besitzt.

Beispiel 4.12. (a) Eine herkömmliche Glühbirne hält heute durchschnittlich (im Median) 1000 Stunden. Unter der Annahme, dass die Anzahl der funktionierenden Glühbirnen exponentiell abnimmt, erhalten wir

$$50\% = 0.5 = P(\text{Zeit bis Defekt} > 1000h) = e^{-1000\lambda} \quad (4.25)$$

was auf eine Ausfallrate von ca. $\lambda = 0.0007$ pro Stunde führt. Angenommen Sie haben in Ihrem Haushalt 20 Glühbirnen, die im Monat jeweils 100 Stunden betrieben werden. Die Zufallsvariable X sei die Anzahl Glühbirnen, die pro Monat ausfallen. Wie wahrscheinlich ist es, dass Sie mindestens eine Glühbirne im Monat tauschen müssen? Wir erhalten

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

(b) Die **Inzidenz** oder **Inzidenzrate** bezeichnet den Anteil der Personen, die in einem bestimmten Zeitraum neu erkranken. In Finnland ist die Inzidenzrate für Typ-1-Diabetes für Kinder bis zum Alter

von 15 Jahren weltweit am höchsten. Dort erkranken im Mittel pro Jahr 55 von 100 000 Kindern in diesem Alter neu an Typ-1-Diabetes (Harjutsalo et al. (2013)), was einer Rate von $\lambda = 0.00055$ entspricht. Laut Wikipedia contributors (2022c) leben in Finnland ca. 900 000 Kinder in diesem Alter; d.h., im Mittel ist in Finnland mit 495 Neuerkrankungen pro Jahr zu rechnen. Es sei X die Anzahl an Neuerkrankungen pro Jahr. Wie wahrscheinlich ist es, dass es in Finnland in einem Jahr zu mehr als 450 Neuerkrankungen kommt? Wie wir in Kürze sehen werden, erhalten wir

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

Wir geben die Definition der Poisson-Verteilung an.

Definition 4.13 (Poisson-Verteilung). *Die Zufallsvariable X folgt einer **Poisson-Verteilung** mit Parameter $\lambda \in (0, \infty)$, falls sie die folgende Wahrscheinlichkeitsfunktion besitzt*

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

kurz: $X \sim \text{Pois}(\lambda)$.

Wir geben einige zusätzliche Erläuterungen.

Bemerkung 4.14. (a) Der Parameter λ beschreibt, wie viele Ereignisse im Mittel in einem vorgegebenem Zeitintervall zu erwarten sind.

(b) Die Poisson-Verteilung hat eine Vielzahl von Anwendungen und kann auch als Approximation der Binomialverteilung verwendet werden, nämlich dann, wenn eine kleine Wahrscheinlichkeit p und eine große Fallzahl n vorliegen. Als Parameter λ der approximierenden Poisson-Verteilung erhält man in diesem Fall $\lambda = np$. Man nennt die Poisson-Verteilung daher auch die Verteilung der seltenen Ereignisse.

(c) Als Erwartungswert und Varianz von $\text{Pois}(\lambda)$ ergibt sich

$$E(X) = \lambda \quad \text{Var}(X) = \lambda \quad (4.28)$$

Das Kürzel in R für die Poisson-Verteilung lautet `pois`, was auf die Funktionen `dpois`, `ppois`, `qpois` und `rpois` führt. Wir berechnen die Wahrscheinlichkeiten aus dem obigen Beispiel mit Hilfe von R.

```
1 ## a) Keine defekte Glühbirne
2 dpois(0, lambda = 1.4)
```

```
[1] 0.246597
```

Mit Hilfe von `ppois` und `lower.tail = FALSE` erhalten wir

```
1 ## a) Mind. eine defekte Glühbirne
2 ppois(0, lambda = 1.4, lower.tail = FALSE)
```

```
[1] 0.753403
```

```
1 ## b) Mehr als 450 Neuerkrankungen
2 ppois(450, lambda = 495, lower.tail = FALSE)

[1] 0.9784977
```

Mit Hilfe der Quantilfunktion können wir ausrechnen, wie viele Glühbirnen wir mit hoher Sicherheit (hier mindestens 99%) höchstens im Monat austauschen müssen.

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

[1] 5
```

D.h., ein Vorrat von fünf Glühbirnen sollte mit hoher Sicherheit ($\geq 99\%$) für mehr als einen Monat reichen. Die exakte Wahrscheinlichkeit ist

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

[1] 0.9967989
```

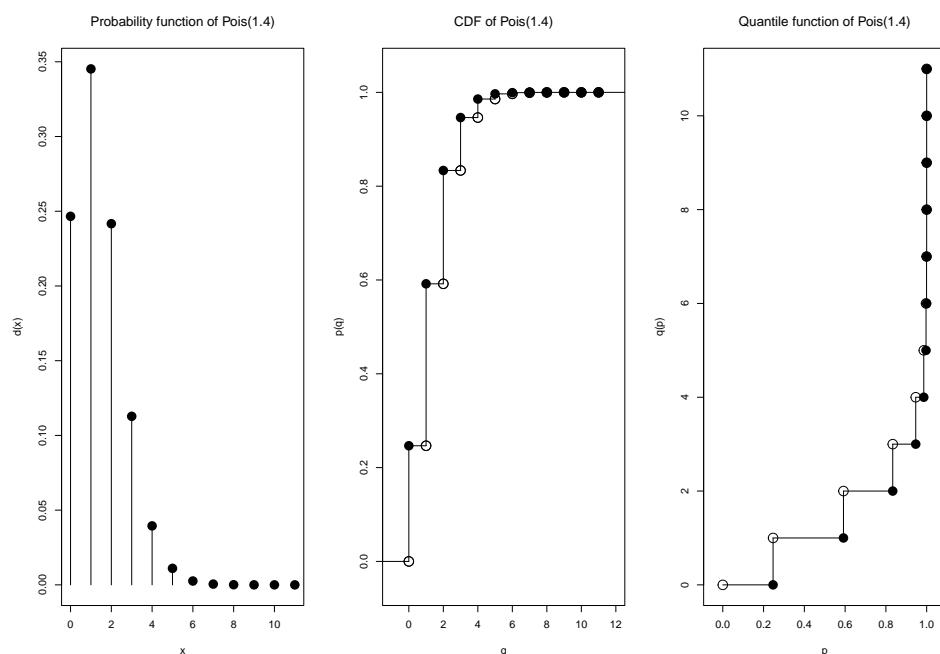
Wir simulieren für zehn Jahre die Anzahl der Neuerkrankungen von Typ-1-Diabetes in Finnland.

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

[1] 469 487 462 477 481 504 501 537 490 490
```

Wir stellen die Verteilung für das Glühbirnenbeispiel mit Hilfe des Pakets "distr" (Ruckdeschel et al. (2006)) graphisch dar.

```
1 X <- Pois(lambda = 1.4)
2 plot(X)
```



Schließlich fassen wir noch die Verteilung mit Hilfe der Funktion `summary` des Paketes "distr6" (Sonabend and Kiraly (2022)) zusammen.

```
1 X6 ← Poisson$new(rate = 1.4)
2 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 Stetige Verteilungen

Eine Zufallsvariable X , welche alle Werte in einem reellen Intervall $I \subset \mathbb{R}$ annehmen kann, heißt eine **stetige Zufallsvariable**.

Anmerkung:

Dieser Begriff der Stetigkeit spiegelt keine Eigenschaft der Funktion X wider; d.h., die Zufallsvariable X muss keine stetige Funktion sein. Stattdessen leitet sich dieser Begriff der Stetigkeit – genauer der absoluten Stetigkeit – von der Verteilung von X ab. Dies ist gleichbedeutend damit, dass die Verteilungsfunktion p von X (fast überall) eine Ableitung $d = p'$ besitzt bzw. umgekehrt p die Stammfunktion von d ist

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

Wir können die **stetige Wahrscheinlichkeitsverteilung** oder kurz die **stetige Verteilung** der Zufallsvariable X mit Hilfe der sogenannten **Wahrscheinlichkeitsdichte** oder kurz **Dichte** d beschreiben, wobei gelten muss

$$d(x) \geq 0 \quad \text{für (fast) alle } x \in \mathbb{R} \text{ und} \quad \int_{-\infty}^{\infty} d(x) dx = 1 \quad (4.30)$$

Wir erhalten die Wahrscheinlichkeit $P(X \in (a, b])$ für ein Intervall $(a, b] \in \mathbb{R}$ durch

$$P(X \in (a, b]) = \int_a^b d(x) dx = p(b) - p(a) \quad (4.31)$$

Die Wahrscheinlichkeit entspricht demnach der Fläche unter der Dichte. Insbesondere folgt daraus

$$P(X = x) = 0 \quad (4.32)$$

d.h., einzelne Punkte besitzen die Wahrscheinlichkeit 0. Folglich gilt

$$P(X \in (a, b)) = P(X \in (a, b]) = P(X \in [a, b)) = P(X \in [a, b]) \quad (4.33)$$

Es macht also keinen Unterschied, ob wir offene, halboffene oder abgeschlossene Intervalle betrachten. Die Quantilfunktion erhalten wir ähnlich wie im diskreten Fall allgemein als

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

Jede Verteilungsfunktion ist monoton wachsend, wenn diese sogar streng monoton wachsend (also injektiv) ist, was auf viele stetige Verteilungen zutrifft, so erhält man die Quantilfunktion als die übliche Umkehrfunktion der Verteilungsfunktion.

Wie im Fall der Berechnung von Wahrscheinlichkeiten muss auch für die Berechnung von Erwartungswert und Varianz im Fall stetiger Zufallsvariablen integriert werden. Wir erhalten den Erwartungswert als

$$\mathbb{E}(X) = \int_{-\infty}^{\infty} x d(x) dx \quad (4.35)$$

und die Varianz als

$$\text{Var}(X) = \int_{-\infty}^{\infty} (x - \mathbb{E}(X))^2 d(x) dx \quad (4.36)$$

Anmerkung:

Streng genommen existieren in der Praxis keine stetigen Zufallsvariablen, da alle Messungen, die wir vornehmen, nur mit einer beschränkten Genauigkeit möglich sind und somit nur auf endlich viele Ergebnisse führen können. Die stetigen Zufallsvariablen stellen somit eine abstrakte Beschreibung der Realität dar, in der wir die beschränkte Genauigkeit unserer Messungen ignorieren. Nichtsdestotrotz sind diese sehr hilfreich, da sich dadurch Berechnungen vereinfachen und sie außerdem hinreichend genaue Beschreibungen für viele praktische Anwendungen liefern.

Normalverteilung

In Folgenden werden wir einige wichtige stetige Verteilungen kennenlernen. Wir beginnen mit der wohl wichtigsten Verteilung in der Statistik, der Normal- oder Gauß-Verteilung.

Definition 4.15 (Normalverteilung). Eine reelle Zufallsvariable X folgt einer **Normal- oder Gauß-Verteilung** mit Mittelwert $\mu \in \mathbb{R}$ und Standardabweichung $\sigma \in (0, \infty)$, falls sie folgende Dichte besitzt

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

Wir schreiben kurz: $X \sim Norm(\mu, \sigma^2)$.

Wir geben einige zusätzliche Erklärungen.

Bemerkung 4.16. (a) Die zentrale Bedeutung der Normalverteilung folgt aus der Tatsache, dass eine Überlagerung (Summe) von unabhängigen Einflussfaktoren, unter recht schwachen Voraussetzungen zumindest näherungsweise durch diese Verteilung beschrieben werden kann. Es ist dies die Umschreibung der Aussage eines der wichtigsten Sätze der Wahrscheinlichkeitstheorie, nämlich des **zentralen Grenzwertsatzes**. Die Normalverteilung dient aber nicht nur als ein Modell zur Beschreibung erhobener Daten, sondern auch als eine Verteilung, mit der man statistische Verfahren (meist näherungsweise) charakterisieren kann.

(b) Bei Vorliegen einer Normalverteilung können mit Hilfe von Mittelwert und Standardabweichung auch recht präzise Aussagen zu den Wahrscheinlichkeiten gewisser Intervalle gemacht werden. Es gilt

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

Daraus ergibt sich die oftmals praktische und recht leicht zu merkende **2σ -Regel**: Innerhalb eines Abstands von 2σ um den Mittelwert (Erwartungswert) liegen ca. 95% der Werte. Die 2σ -Regel ist relativ robust und trifft näherungsweise auch auf recht viele andere Verteilungen zu.

(c) Die Normalverteilung spielt auch eine wichtige Rolle in der Qualitäts- und Prozesskontrolle. So ist der Name eines der bekanntesten Qualitätsmanagementsysteme – Six Sigma – im Hinblick auf die Normalverteilung zu verstehen ($\pm 6\sigma$). Das Ziel ist demnach eine äußerst geringe Ausfallwahrscheinlichkeit.

(d) Wie die Namen der Parameter bereits andeuten, erhalten wir als Erwartungswert und Varianz von $Norm(\mu, \sigma^2)$

$$E(X) = \mu \quad Var(X) = \sigma^2 \quad (4.39)$$

(e) Falls $X \sim Norm(\mu, \sigma^2)$, dann gilt

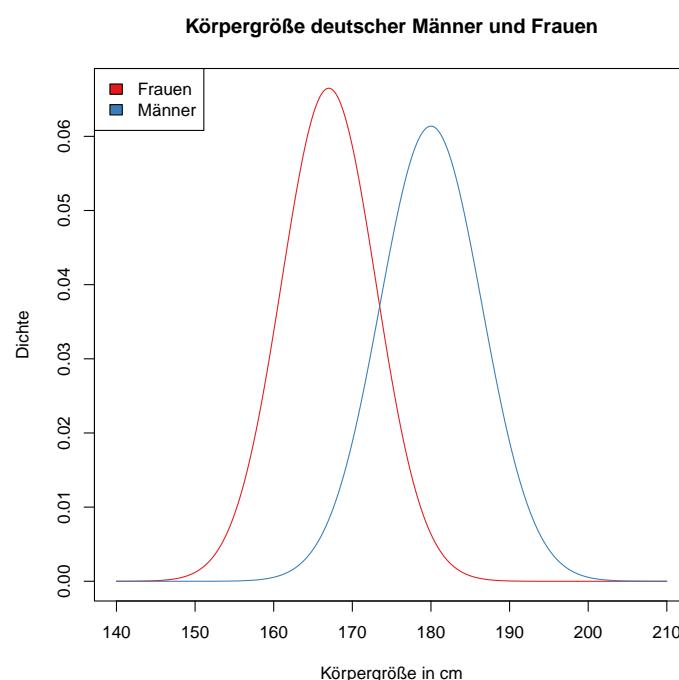
$$Z = \frac{X - \mu}{\sigma} \sim Norm(0, 1) \quad (4.40)$$

und man nennt $Norm(0, 1)$ auch die **Standardnormalverteilung**. Der Buchstabe Z wurde hier bewusst gewählt, da man manchmal auch von Z -Verteilung und entsprechend von z-Transformation oder z-Score spricht (vgl. Bemerkung 2.23)).

Die Normalverteilung wird in R mit `norm` abgekürzt, was auf die Funktionen `dnorm` (Dichte), `pnorm`, `qnorm` und `rnorm` führt. Die Namen der Parameter sind `mean` und `sd`. Wir geben zwei Beispiele für die Normalverteilung an.

Beispiel 4.17. (a) Die Körpergröße der Erwachsenen eines Landes lässt sich sehr gut durch die Normalverteilung beschreiben. Im Fall der deutschen Frauen erhalten wir einen Mittelwert von ca. 167 cm und eine Standardabweichung von ca. 6.0 cm, bei den deutschen Männern einen Mittelwert von ca. 180 cm und eine Standardabweichung von ca. 6.5 cm (Wikipedia (2015)). Wir stellen die Dichtefunktionen für Männer und Frauen graphisch dar. Wir verwenden hierzu die Funktion `curve`.

```
1 curve(expr = dnorm(x, mean = 167, sd = 6.0), from = 140, to = 210, n = 501,
2       col = "#E41A1C", xlab = "Körpergröße in cm", ylab = "Dichte",
3       main = "Körpergröße deutscher Männer und Frauen")
4 curve(expr = dnorm(x, mean = 180, sd = 6.5), from = 140, to = 210, n = 501,
5       add = TRUE, col = "#377EB8")
6 legend("topleft", legend = c("Frauen", "Männer"), fill = c("#E41A1C", "#377EB8"))
```



Der Parameter `expr` gibt den R Ausdruck wieder, der dargestellt werden soll. Mit `from` und `to` wird der zu plottende Wertebereich der x-Achse vorgegeben, wobei der Ausdruck `expr` auf einem Gitter aus `n` äquidistanten Punkten ausgewertet und gezeichnet wird. Schließlich können wir durch Verwendung von `add = TRUE` weitere Kurven zu einem bestehenden Plot hinzufügen. Der Anteil der Frauen, die größer als 175 cm sind, beträgt demnach

```
1 pnorm(175, mean = 167, sd = 6.0, lower.tail = FALSE)
[1] 0.09121122
```

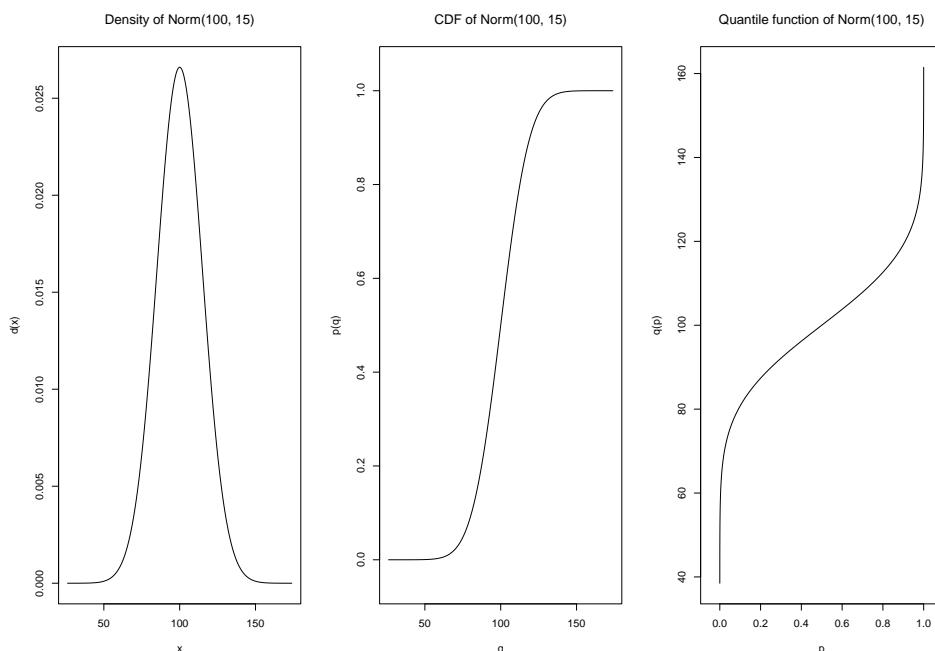
Die größten 5% der Männer sind größer als

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

```
[1] 190.6915
```

(b) Der Intelligenzquotient (IQ) kann ebenfalls sehr gut durch eine Normalverteilung beschrieben werden. Für die sogenannte Abweichungs-IQ-Skala gilt ein Mittelwert von 100 und eine Standardabweichung von 15 (Wikipedia contributors (2022d)). Wir stellen die Normalverteilung mit Hilfe des Pakets "distr" (Ruckdeschel et al. (2006)) graphisch dar.

```
1 X <- Norm(mean = 100, sd = 15)
2 plot(X)
```



Die 2σ -Regel sagt uns, dass demnach ca. 95% der Bevölkerung einen IQ zwischen 70 und 130 haben. Jeweils ca. 2.5% der Bevölkerung haben umgekehrt einen IQ von kleiner 70 oder größer 130. Wir fassen schließlich noch die Verteilung mit Hilfe der Funktion `summary` des Paketes "distr6" (Sonabend and Kiraly (2022)) zusammen.

```
1 X6 <- Normal$new(mean = 100, sd = 15)
2 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-Normalverteilung

Die zweite wichtige stetige Verteilung ist eng mit der Normalverteilung verbunden und wird log-Normalverteilung genannt. Wir geben zunächst die Definition an.

Definition 4.18 (log-Normalverteilung). *Eine reelle Zufallsvariable X , welche nur positive Werte annehmen kann, folgt einer **log-Normalverteilung** mit Mittelwert $\mu \in \mathbb{R}$ und Standardabweichung $\sigma \in (0, \infty)$, falls sie folgende Dichte besitzt*

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

Wir schreiben kurz: $X \sim \text{Lnorm}(\mu, \sigma)$.

Wir geben einige zusätzliche Erklärungen.

Bemerkung 4.19. (a) Die log-Normalverteilung tritt in vielen wissenschaftlichen Disziplinen auf. Insbesondere laufen viele biologische Prozesse auf einer exponentiellen Skala ab, weshalb sich viele Parameter, die in Biologie und Medizin gemessen werden durch eine log-Normalverteilung beschreiben lassen. D.h., in ähnlicher Weise wie eine additive Überlagerung von Zufallsvariablen im Sinne des zentralen Grenzwertsatzes zu einer Normalverteilung führt, liefert eine multiplikative Überlagerung eine log-Normalverteilung.

(b) Falls X eine log-normalverteilte Zufallsvariable ist, so ist $\log(X)$ normalverteilt. Im Hinblick auf (a) können wir dies so interpretieren, dass aus einer multiplikativen Überlagerung durch Anwendung der Logarithmusfunktion eine additive Überlagerung wird.

(c) Die Parameter der log-Normalverteilung entsprechen gerade dem Erwartungswert und der Varianz von $\log(X)$. Für die Zufallsvariable X selbst gilt

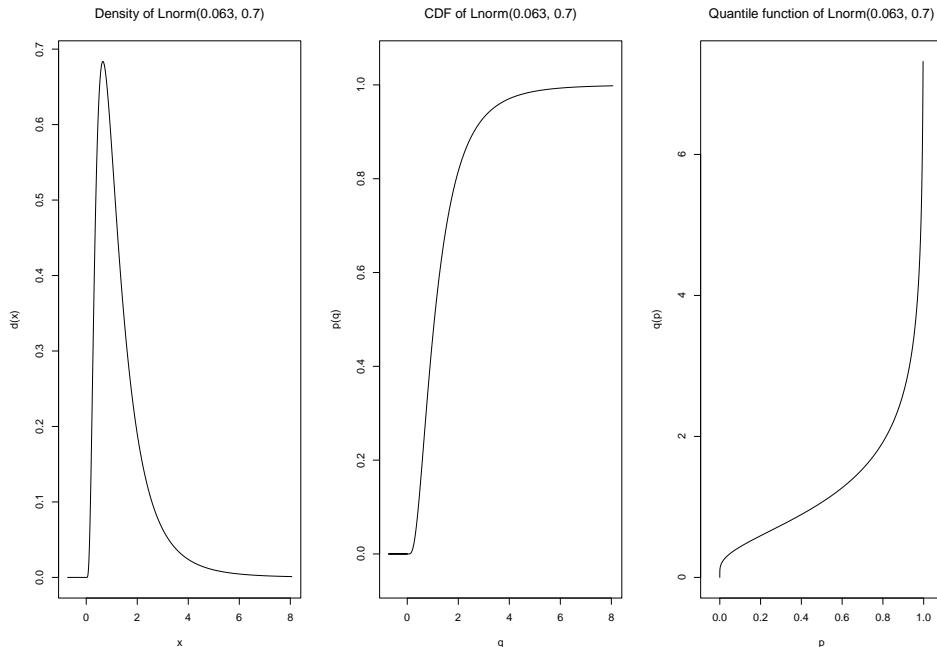
$$E(X) = e^{\mu + \frac{\sigma^2}{2}} \quad \text{Var}(X) = e^{2\mu + \sigma^2} (e^{\sigma^2} - 1) \quad (4.42)$$

Die log-Normalverteilung wird in R mit `lnorm` abgekürzt. Entsprechend heißen die zugehörigen Funktionen `dlnorm`, `plnorm`, `qlnorm` und `rlnorm`, wobei die Namen der Parameter `meanlog` und `sdlog` lauten. Wir stellen ein Beispiel für die Anwendung der log-Normalverteilung vor.

Beispiel 4.20. (a) Zur Untersuchung der Funktion der Schilddrüse wird die Konzentration von Thyreotropin (TSH) im Blut untersucht. Dessen Konzentration bei Personen mit einer normalen Schilddrüsenfunktion lässt sich durch eine log-Normalverteilung beschreiben (Hamilton et al. (2008)). Die Angaben für den Normbereich (Referenzbereich) schwanken insbesondere hinsichtlich der oberen Grenze des Normbereichs je nach Quelle. Wir verwenden als Normbereich für Erwachsene $0.27\text{-}4.2 \mu\text{U}/\text{ml}$ (Hagemann (2014)). Für die Bestimmung der genauen Verteilung von TSH bei Personen mit normaler Schilddrüsenfunktion benutzen wir zum einen den Zusammenhang zur Normalverteilung und zum anderen die Information, dass der **Norm- oder Referenzbereich** eines Parameters, so gewählt ist, dass jeweils nur 2.5% der gesunden Personen höhere oder niedrigere Werte aufweisen können (Wikipedia contributors (2022f)). Im Fall der Normalverteilung entspricht der Normbereich also genähert dem 2σ Intervall.

Nach log-Transformation erhalten wir als Normbereich $[-1.309, 1.435]$. Aufgrund der Symmetrie der Normalverteilung ist der Erwartungswert gerade der Intervallmittelpunkt; d.h., $\mu = 0.063$. Die Länge des Intervalls entspricht grob 4σ , genauer sind es 3.92σ . Ausgehend von einer Intervalllänge von 2.744 ergibt die Division durch 3.92 entsprechend $\sigma = 0.7$. Wir erhalten damit als Verteilung für die log-TSH-Werte $\text{Norm}(0.063, 0.7^2)$ und folglich für die TSH-Werte $\text{Lnorm}(0.063, 0.7)$. Wir stellen die Verteilung der TSH-Werte mit Hilfe des Pakets "distr" (Ruckdeschel et al. (2006)) graphisch dar.

```
1 X ← Lnorm(meanlog = 0.063, sdlog = 0.7)
2 plot(X)
```



Abschließend fassen wir die Verteilung mit Hilfe der Funktion `summary` des Paketes "distr6" (Sonabend and Kiraly (2022)) zusammen.

```
1 X6 ← Lognormal$new(meanlog = 0.063, sdlog = 0.7)
2 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) Eine ganze Reihe von Beispielen zur log-Normalverteilung aus den verschiedensten wissenschaftlichen Disziplinen finden sich in Limpert et al. (2001) und Limpert and Stahel (2011). Insbesondere geben beide Artikel auch Empfehlungen für den Umgang mit log-normalverteilten Daten in der Praxis.

Gamma-Verteilung

Eine sehr flexible Verteilung mit vielen Anwendungen ist die Gamma-Verteilung.

Definition 4.21 (Gamma-Verteilung). *Eine reelle Zufallsvariable X , welche nur positive Werte annehmen kann, folgt einer **Gamma-Verteilung** mit Skalenparameter $\sigma \in (0, \infty)$ und Gestaltparameter $\alpha \in (0, \infty)$, falls sie folgende Dichte besitzt*

$$d(x) = \begin{cases} \frac{1}{\sigma^\alpha \Gamma(\alpha)} x^{\alpha-1} e^{-\frac{x}{\sigma}} & \text{falls } x > 0 \\ 0 & \text{sonst} \end{cases} \quad (4.43)$$

wobei die **Gamma-Funktion** Γ gegeben ist durch

$$\Gamma(x) = \int_0^\infty t^{x-1} e^{-t} dt \quad (4.44)$$

Wir schreiben kurz: $X \sim \text{Gamma}(\sigma, \alpha)$.

Wir geben einige zusätzliche Erklärungen.

Bemerkung 4.22. (a) Durch den Gestaltparameter ist die Gamma-Verteilung sehr flexibel und besitzt daher vielfältige Anwendungen etwa in der Versicherungsmathematik, Genetik oder auch der Medizin.

(b) Ein wichtiger Spezialfall der Gamma-Verteilung ist die **Exponentialverteilung**, die man für $\alpha = 1$ erhält. Als Parameter verwendet man dabei üblicherweise die Rate $\lambda = \frac{1}{\sigma}$, was auf die folgende Dichte führt

$$d(x) = \begin{cases} \lambda e^{-\lambda x} & \text{falls } x > 0 \\ 0 & \text{sonst} \end{cases} \quad (4.45)$$

Wir schreiben kurz: $X \sim \text{Exp}(\lambda)$. Man kann sie als das stetige Gegenstück zur geometrischen Verteilung, einem Spezialfall der negativen Binomialverteilung, auffassen. Sie beschreibt die Zeit zwischen zwei Ereignissen eines Prozesses, in dem die Ereignisse stetig und voneinander unabhängig mit einer festen Rate auftreten. Man spricht auch von einer gedächtnislosen Verteilung, da die Wahrscheinlichkeit zukünftiger Ereignisse nicht von der Vergangenheit abhängt. Die Exponentialverteilung wird als ein einfaches Modell zum Schätzen von Überlebenswahrscheinlichkeiten oder Lebensdauern verwendet. Die Eigenschaft der Gedächtnislosigkeit bedeutet in diesem Zusammenhang, dass es keine Alterung gibt.

(c) Betrachtet man simultan $k \in \mathbb{N}$ unabhängige Prozesse, deren Ereignisse jeweils $\text{Exp}(\lambda)$ verteilt sind, so ergibt sich die **Erlang-Verteilung** als deren Summe. Die Erlang-Verteilung wiederum ist selbst ein Spezialfall der Gamma-Verteilung, wobei in diesem Fall $\alpha = k$ gilt und man üblicherweise wie im Fall der Exponentialverteilung die Rate $\lambda = \frac{1}{\sigma}$ als zweiten Parameter verwendet. Als Dichte ergibt sich damit

$$d(x) = \begin{cases} \frac{\lambda^k}{(k-1)!} x^{k-1} e^{-\lambda x} & \text{falls } x > 0 \\ 0 & \text{sonst} \end{cases} \quad (4.46)$$

Die Erlang-Verteilung kann zum Beispiel verwendet werden, um bei einem Callcenter die Zeit zwischen Anrufen zu modellieren, wobei man die Anzahl der Anrufe zum Beispiel mit einer Poisson-Verteilung beschreiben kann. Die Erlang-Verteilung ist demnach eine Wartezeitverteilung und kann als die stetige Variante der negativen Binomialverteilung angesehen werden.

(d) Ein weiterer wichtiger Spezialfall der Gamma-Verteilung ist die **χ^2 -Verteilung** mit $n \in \mathbb{N}$ Freiheitsgraden, kurz $\text{Chisq}(n)$. Es gilt $\sigma = 2$ und $\alpha = \frac{n}{2}$. Die Dichte ist damit

$$d(x) = \begin{cases} \frac{1}{2^{\frac{n}{2}} \Gamma(\frac{n}{2})} x^{\frac{n}{2}-1} e^{-\frac{1}{2}x} & \text{falls } x > 0 \\ 0 & \text{sonst} \end{cases} \quad (4.47)$$

Man kann die χ^2 -Verteilung auch aus der Normalverteilung ableiten wie wir im Verlauf dieses Abschnitts noch sehen werden.

(e) Als Erwartungswert und Varianz von $X \sim \text{Gamma}(\sigma, \alpha)$ erhält man

$$E(X) = \alpha\sigma \quad \text{Var}(X) = \alpha\sigma^2 \quad (4.48)$$

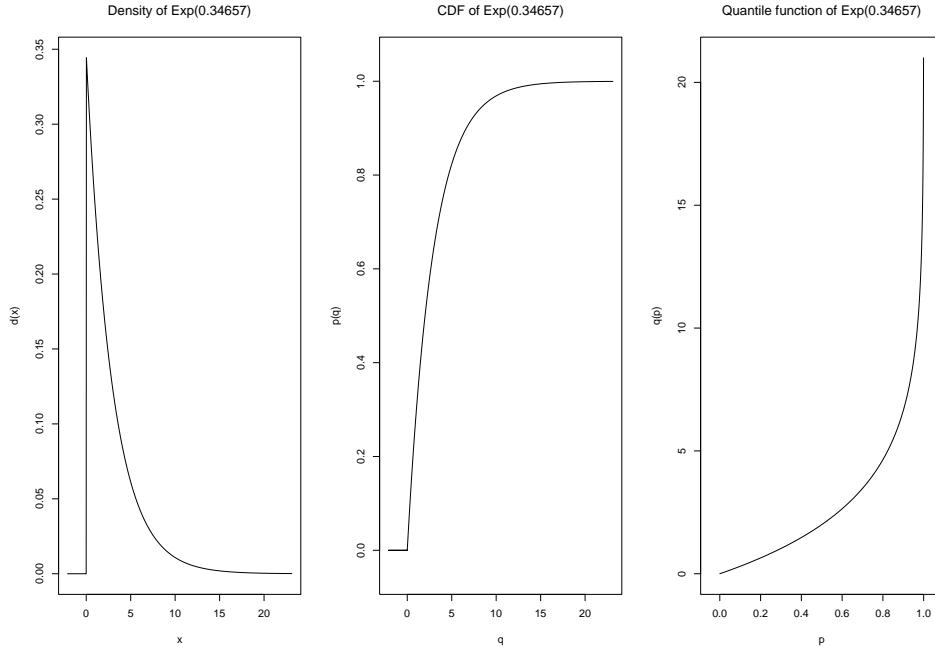
Wir stellen einige Anwendungsbeispiele der Gamma-Verteilung vor. Die Gamma-Verteilung ist in R in Form der Funktionen `dgamma`, `pgamma`, `qgamma` und `rgamma` gegeben, wobei die Parameter `scale` und `shape` lauten. Die Exponentialverteilung ist durch die Funktionen `dexp`, `pexp`, `qexp` und `rexp` implementiert mit Parameter `rate`.

Beispiel 4.23. (a) Ein moderner Akku in einem Smartphone hat eine mittlere (Median) Lebensdauer von zwei Jahren. Wir verwenden die Exponentialverteilung, um die Lebensdauer zu modellieren und erhalten

$$0.5 = P(X \leq 2\text{Jahre}) = 1 - e^{-2\text{Jahre} \cdot \lambda} \quad (4.49)$$

Dies führt auf eine Ausfallrate pro Jahr von $\lambda = 0.34657$. Wir stellen die Verteilung mit Hilfe des Pakets "distr" (Ruckdeschel et al. (2006)) graphisch dar.

```
1 X <- Exp(rate = 0.34657)
2 plot(X)
```



Wir fassen die Verteilung mit Hilfe der Funktion `summary` des Paketes "distr6" (Sonabend and Kiraly (2022)) zusammen.

```
1 X6 <- Exponential$new(rate = 0.34657)
2 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
```

Wie wahrscheinlich ist es, dass der Akku bereits im ersten Jahr defekt ist?

```
1 pexp(1, rate = 0.34657)
```

```
[1] 0.2928907
```

D.h., annähernd 30% der Akkus gehen bereits im ersten Jahr kaputt. Nach wie vielen Jahren sind dann 99% der Akkus defekt? Wir erhalten

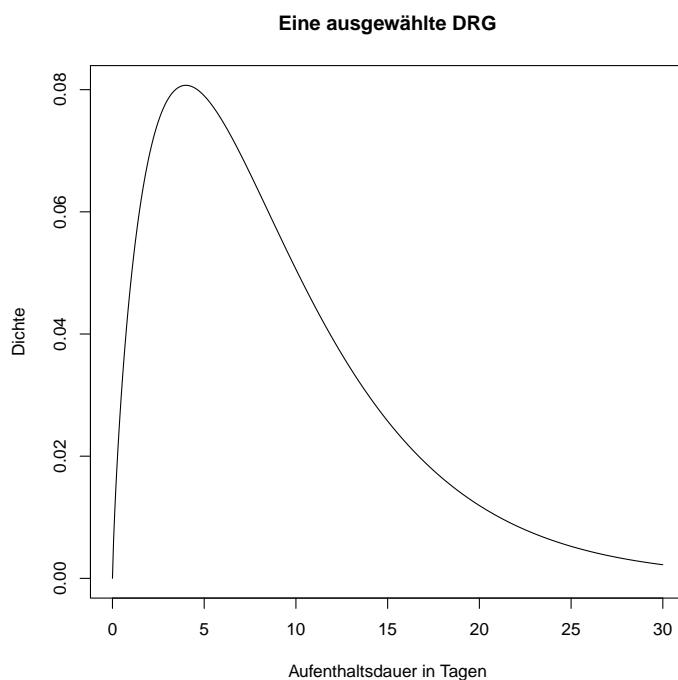
```
1 qexp(0.99, rate = 0.34657)
```

```
[1] 13.28785
```

Im Extremfall ($\leq 1\%$) kann ein Akku also theoretisch auch mehr als 13 Jahre halten.

(b) Die Gamma-Verteilung stellt eine Möglichkeit dar, die Krankenhausaufenthaltsdauer einer Gruppe von Patienten zu modellieren; zum Beispiel aller Patienten mit einer bestimmten Diagnose oder genauer die zu einer bestimmten DRG (diagnosis related group = diagnosebezogene Fallgruppe) gehören. Ausgehend von einem Skalenparameter von $\sigma = 5$ und einem Gestaltparameter von $\alpha = 1.8$ für eine ausgewählte DRG erhalten wir die folgende Dichte, die wir mit Hilfe der Funktion curve darstellen

```
1 curve(dgamma(x, scale = 5, shape = 1.8), from = 0, to = 30, n = 501,
2       xlab = "Aufenthaltsdauer in Tagen", ylab = "Dichte",
3       main = "Eine ausgewählte DRG")
```



Wir fassen die Verteilung mit Hilfe der Funktion `summary` des Paketes "distr6" (Sonabend and Kiraly (2022)) zusammen.

```
1 X6 ← Gamma$new(shape = 1.8, scale = 5)
2 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
```

Die Mehrheit der Patienten wird also innerhalb weniger Tage entlassen. Es kann aber auch vorkommen, dass Patienten deutlich länger als zwei Wochen im Krankenhaus bleiben müssen. Wie wahrscheinlich ist es, dass man mehr als zehn Tage im Krankenhaus bleiben muss? Es ergibt sich

```
1 pgamma(10, scale = 5, shape = 1.8, lower.tail = FALSE)
```

```
[1] 0.3472818
```

Demnach muss sich etwas mehr als ein Drittel der Patienten auf einen Aufenthalt von mehr als zehn Tagen einstellen. Nach wie vielen Tagen sind 99% der Patienten entlassen? Wir erhalten

```
1 qgamma(0.99, scale = 5, shape = 1.8)
```

```
[1] 31.3043
```

Innerhalb eines Monats sind folglich nahezu alle Patienten der ausgewählten DRG wieder zu Hause.

Weibull-Verteilung

Falls sich die Ausfallrate über die Zeit verändert, stellt die Weibull-Verteilung eine Möglichkeit dar, den Vorgang zu modellieren. Wir beginnen mit der Definition.

Definition 4.24 (Weibull-Verteilung). Eine reelle Zufallsvariable X , welche nur positive Werte annehmen kann, folgt einer **Weibull-Verteilung** mit Skalenparameter $\sigma \in (0, \infty)$ und Gestaltparameter $\alpha \in (0, \infty)$, falls sie folgende Dichte besitzt

$$d(x) = \begin{cases} \frac{\alpha}{\sigma} \left(\frac{x}{\sigma}\right)^{\alpha-1} e^{-\left(\frac{x}{\sigma}\right)^\alpha} & \text{falls } x > 0 \\ 0 & \text{sonst} \end{cases} \quad (4.50)$$

Wir schreiben kurz: $X \sim \text{Weibull}(\sigma, \alpha)$

Wir geben einige zusätzliche Erläuterungen.

Bemerkung 4.25. (a) Die Weibull-Verteilung spielt eine wichtige Rolle in der Zuverlässigenanalyse von Bauteilen und Komponenten zum Beispiel in der Autoindustrie. Im Unterschied zur (gedächtnislosen) Exponentialverteilung kann über den Gestaltparameter auch die Alterung mit modelliert werden.

(b) Die Weibull-Verteilung gehört zur Klasse der **Extremwertverteilungen**, genauer stellt sie den Typ III von Extremwertverteilungen dar. Nach dem Satz von Fisher–Tippett–Gnedenko entsteht sie unter gewissen Voraussetzungen, wenn das Maximum von unabhängigen Zufallsvariablen betrachtet wird.

(c) Im Fall $\alpha = 1$ entspricht die Weibull-Verteilung gerade der Exponentialverteilung.

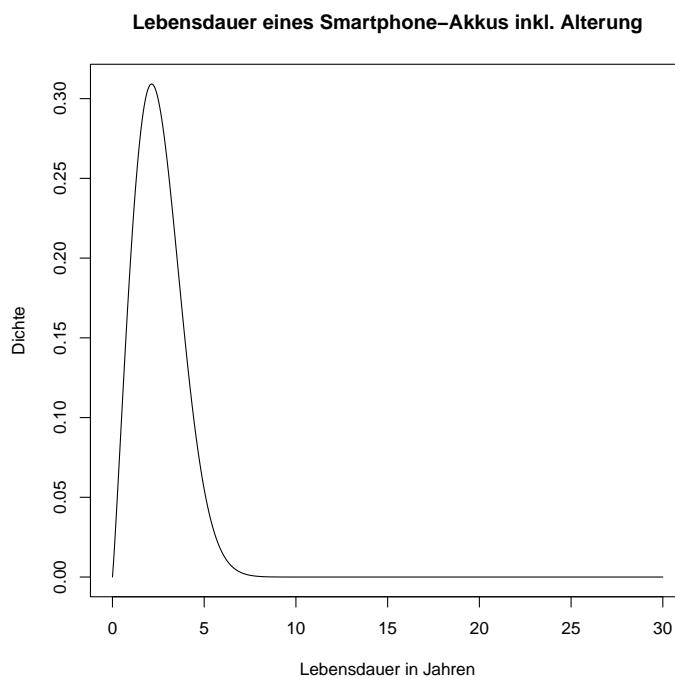
(d) Wir erhalten als Erwartungswert und Varianz

$$E(X) = \sigma \Gamma\left(1 + \frac{1}{\alpha}\right) \quad \text{Var}(X) = \sigma^2 \left(\Gamma\left(1 + \frac{2}{\alpha}\right) - \Gamma\left(1 + \frac{1}{\alpha}\right)^2 \right) \quad (4.51)$$

Wir stellen einige Beispiele vor.

Beispiel 4.26. (a) Wir betrachten wieder die Akkus von modernen Smartphones mit einer Ausfallrate von 0.34657 wie in Beispiel 4.23 (a); d.h., $\sigma = \frac{1}{0.34657}$. Zusätzlich nehmen wir an, dass sich die Alterung der Akkus durch den Gestaltparameter $\alpha = 2.12$ beschreiben lässt. Wir stellen die Dichte der entsprechenden Weibull-Verteilung mittels der Funktion `curve` dar

```
1 curve(dweibull(x, scale = 1/0.34657, shape = 2.12), from = 0, to = 30, n = 501,
2   xlab = "Lebensdauer in Jahren", ylab = "Dichte",
3   main = "Lebensdauer eines Smartphone-Akkus inkl. Alterung")
```



Wir fassen die Verteilung mit Hilfe der Funktion `summary` des Paketes "distr6" (Sonabend and Kiraly (2022)) zusammen.

```
1 X6 ← distr6::Weibull$new(shape = 2.12, scale = 1/0.34657)
2 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
```

Da es im Paket "distr" ebenfalls eine Funktion mit dem Namen `Weibull` gibt, müssen wir zur Unterscheidung zusätzlich `distr6::` angeben. Genauer ausgedrückt, dient der Operator `::` dazu, auf die

Objekte des **Namensraums** (Namespace) eines Paketes, hier "`distr6`", zuzugreifen.

Unter Berücksichtigung der Alterung sind weniger Akkus im ersten Jahr defekt als im Fall der Exponentialverteilung, nämlich

```
1 pweibull(1, scale = 1/0.34657, shape = 2.12)
[1] 0.1003672
```

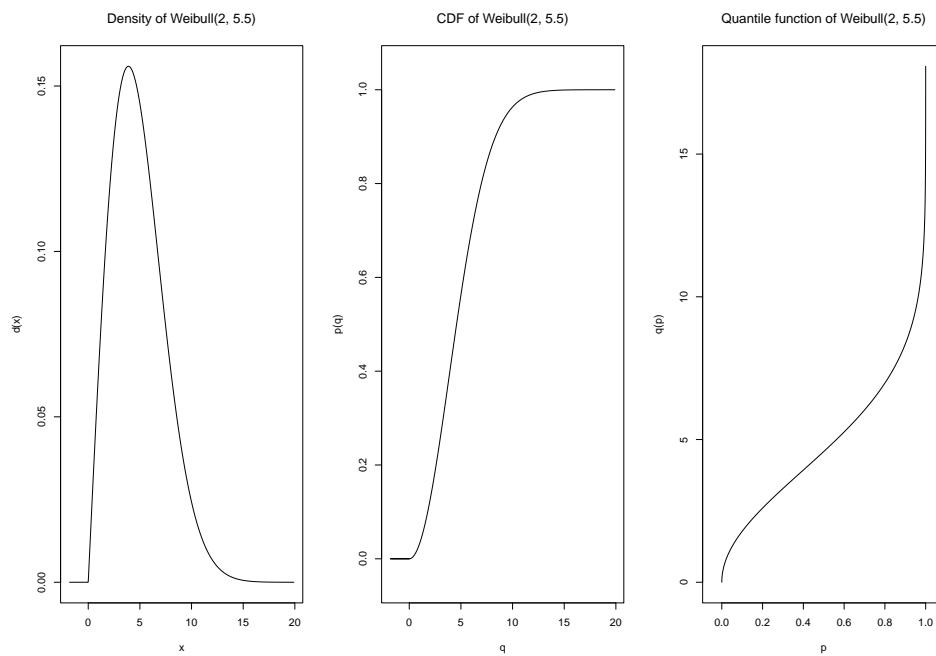
d.h., nur ca. 10%. Umgekehrt sind aber 99% der Akkus frühzeitiger defekt und halten nur

```
1 qweibull(0.99, scale = 1/0.34657, shape = 2.12)
[1] 5.930083
```

Jahre.

(b) Auch bei der Modellierung von Windgeschwindigkeiten findet die Weibull-Verteilung Anwendung. Wir nehmen an, dass sich die maximalen Windgeschwindigkeiten (in $\frac{m}{s}$) pro Tag an einem ausgewählten Ort durch eine Weibull-Verteilung mit $\sigma = 5.5$ und $\alpha = 2$ beschreiben lassen. Wir stellen diese Verteilung mit Hilfe des Pakets "`distr`" (Ruckdeschel et al. (2006)) graphisch dar.

```
1 X <- Weibull(scale = 5.5, shape = 2)
2 plot(X)
```



Wir fassen die Verteilung mit Hilfe der Funktion `summary` des Paketes "`distr6`" (Sonabend and Kiraly (2022)) zusammen.

```
1 X6 <- distr6::Weibull$new(shape = 2, scale = 5.5)
2 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
```

Wir erhalten als Median der Windgeschwindigkeiten.

```
1 qweibull(0.5, scale = 5.5, shape = 2)
```

```
[1] 4.57905
```

Der Median liegt demnach im Bereich der schwachen Brise. Mit welcher Wahrscheinlichkeit herrscht mindestens eine starker Wind; d.h., eine Windgeschwindigkeit von mindestens $11 \frac{m}{s}$? Es ergibt sich

```
1 pweibull(11, scale = 5.5, shape = 2, lower.tail = FALSE)
```

```
[1] 0.01831564
```

also an ca. 2% der Tage.

χ^2 -, t- und F-Verteilung

Abschließend führen wir kurz einige stetige Verteilungen ein, die im Zusammenhang mit der Normalverteilung entstehen und die im Rahmen der induktiven Statistik eine wichtige Rolle spielen. Wir geben zunächst die Definitionen an.

Definition 4.27 (χ^2 -, t-, F-Verteilung). (a) Eine reelle Zufallsvariable X , welche nur positive Werte annehmen kann, folgt einer **χ^2 -Verteilung** mit $n \in \mathbb{N}$ Freiheitsgraden, falls sie folgende Dichte besitzt

$$d(x) = \begin{cases} \frac{1}{2^{\frac{n}{2}} \Gamma(\frac{n}{2})} e^{-\frac{1}{2}x} x^{(\frac{n}{2}-1)} & \text{falls } x > 0 \\ 0 & \text{sonst} \end{cases} \quad (4.52)$$

kurz: $X \sim \text{Chisq}(n)$.

(b) Eine reelle Zufallsvariable X folgt einer **t-Verteilung** mit $n \in \mathbb{N}$ Freiheitsgraden, falls sie folgende Dichte besitzt

$$d(x) = \frac{\Gamma(\frac{n+1}{2})}{\Gamma(\frac{n}{2})\sqrt{\pi n}} \left(1 + \frac{x^2}{n}\right)^{-\frac{n+1}{2}} \quad (4.53)$$

kurz: $X \sim t(n)$

(c) Eine reelle Zufallsvariable X , welche nur positive Werte annehmen kann, folgt einer **F-Verteilung** mit $m \in \mathbb{N}$ und $n \in \mathbb{N}$ Freiheitsgraden, falls sie folgende Dichte besitzt

$$d(x) = \begin{cases} \frac{\Gamma(\frac{n+m}{2})}{\Gamma(\frac{n}{2})\Gamma(\frac{m}{2})} n^{\frac{n}{2}} m^{\frac{m}{2}} \frac{x^{\frac{n}{2}-1}}{(m+nx)^{\frac{n+m}{2}}} & \text{falls } x > 0 \\ 0 & \text{sonst} \end{cases} \quad (4.54)$$

kurz: $X \sim F(m, n)$.

Wir geben einige zusätzliche Erläuterungen.

Bemerkung 4.28. (a) Die χ^2 -Verteilung ergibt sich als Summe der Quadrate von n unabhängigen standardnormalverteilten Zufallsvariablen. In der induktiven Statistik tritt diese Verteilung zum Beispiel im Zusammenhang mit der Schätzung der Varianz auf.

(b) Gegeben seien eine standardnormalverteilte Zufallsvariable Z und eine unabhängige Chisq(n)-verteilte Zufallsvariable Y . Dann gilt

$$\sqrt{n} \frac{Z}{\sqrt{Y}} \sim t(n) \quad (4.55)$$

Diese Verteilung ergibt sich in der induktiven Statistik zum Beispiel bei der Betrachtung von standardisierten arithmetischen Mittelwerten wie etwa im Fall des t-Tests (vgl. Abschnitt 6.3).

(c) Gegeben seien die unabhängigen Zufallsvariablen $X \sim \text{Chisq}(m)$ und $Y \sim \text{Chisq}(n)$. Dann folgt

$$\frac{n \cdot X}{m \cdot Y} \sim F(m, n) \quad (4.56)$$

Diese Verteilung erhält man in der induktiven Statistik etwa bei der Untersuchung des Quotienten von zwei Varianzen wie etwa im Fall der ANOVA (vgl. Abschnitt 6.3).

Anmerkung:

Es gibt natürlich noch eine Vielzahl weiterer Wahrscheinlichkeitsverteilungen, insbesondere auch multivariate Verteilungen. Alle diese Verteilungen werden als (parametrische) Modelle für die verschiedenen Anwendungen eingesetzt. Außerdem dienen diese grundlegenden Verteilungen als Bausteine für komplexere Modelle, wie etwa Klassifikations- und Regressionsmodelle.

In Tabelle 4.1 findet sich eine Aufstellung wichtiger Begriffe aus der Statistik mit ihren Gegenstücken aus der Wahrscheinlichkeitsrechnung. Es gibt auch Gegenstücke zur (Stichproben-) Korrelation und Kovarianz in der Wahrscheinlichkeitsrechnung. Für deren Definition ist es nötig, die gemeinsame Verteilung zweier reeller Zufallsvariablen, also eine zweidimensionale Verteilung, zu betrachten, worauf ich aber im Rahmen dieser Einführung verzichten werde.

Statistik	Wahrscheinlichkeitsrechnung
Merkmal/Variable	Zufallsvariable
Merkmalsausprägungen	mögliche Werte einer Zufallsvariable
relative Häufigkeit	Wahrscheinlichkeit
Häufigkeitsverteilung	Wahrscheinlichkeitsfunktion
Dichteschätzung	(Wahrscheinlichkeits-)Dichte
empirische Verteilungsfunktion	Verteilungsfunktion
(Stichproben-) Quantil	Quantil
arithmetisches Mittel	Erwartungswert
(Stichproben-) Varianz	Varianz

Tabelle 4.1: Begriffe aus der Statistik und ihre Gegenstücke in der Wahrscheinlichkeitsrechnung

4.3 Übungsaufgaben

Beschreiben und interpretieren Sie jeweils detailliert die Ergebnisse.

1. Stellen Sie die Verteilung $\text{Binom}(20, p)$ für $p \in \{0.1, 0.2, \dots, 0.9\}$ mit Hilfe des Pakets "distr" (Ruckdeschel et al. (2006)) graphisch dar.
2. Berechnen Sie den Erwartungswert und die Varianz von $\text{Binom}(2, p)$ **ohne** die expliziten Formeln (4.15) für den Erwartungswert und die Varianz zu verwenden.
3. Menschen mit der Blutgruppe 0-Negativ sind Universalspender, wobei nur 7% der Bevölkerung diese Blutgruppe besitzen. Angenommen Sie führen eine Studie durch, bei der Sie zufällig 20 Personen auswählen. Wie wahrscheinlich ist es, dass darunter mindestens 3 Universalspender sind? Verwenden Sie die Binomialverteilung.
4. Die Letalität – d.h., der Anteil der Erkrankten, die an der Erkrankung versterben – von Sars-CoV-2 (COVID-19) betrug in Korea bis zum 20.03.2020 ca. 1.09% (Shim et al. (2020)). Nehmen Sie an, dass diese Letalität auch auf die ca. 83.2 Millionen Menschen in Deutschland (2019) zutrifft. Nehmen Sie weiter an, dass 70% der Einwohner in Deutschland an dem Virus erkranken müssen, bis eine Herdenimmunität entsteht. Wie viele Tote sind dann in Deutschland zu erwarten (Erwartungswert)? Wie viele Tote sind dann in Deutschland mit einer Wahrscheinlichkeit von mindestens 99% höchstens zu erwarten (Quantil). Verwenden Sie die Binomialverteilung.
5. In einem Krankenhaus ereignen sich durchschnittlich (Mittel-/Erwartungswert) 1.8 Geburten pro Stunde. Wie viele Kreißäle sollte das Krankenhaus haben, damit mit 95% Wahrscheinlichkeit jede Geburt in einem Kreißsaal stattfinden kann? Verwenden Sie die Poisson-Verteilung.
6. In der Europäischen Union spricht man von einer seltenen Erkrankung, wenn die Inzidenz weniger als 1 von 2000 Personen beträgt. Die Inzidenzrate pro Jahr von Sarkoidose beträgt ca. 5-60 Fälle pro 100 Tausend Einwohner. Wie viele Neuerkrankungen sind in Deutschland pro Jahr zu erwarten, wenn wir vom ungünstigsten Fall (60 Fälle pro 100 Tausend) und einer Bevölkerung von 83.2

Millionen in Deutschland (2019) ausgehen? Wie viele Neuerkrankte sind mit einer Wahrscheinlichkeit von mindestens 99% jährlich höchstens in Deutschland zu erwarten. Verwenden Sie die Poissonverteilung.

7. Ein Ölkonzern führt in einem Gebiet eine geologische Studie durch, in der an zufällig ausgewählten Stellen nach Öl gebohrt wird. Die Wahrscheinlichkeit auf Öl zu treffen betrage im ausgewählten Gebiet 20%. Wie wahrscheinlich ist es, dass mindestens 5 mal gebohrt werden muss, bis zum ersten Mal Öl gefunden wird? Wie oft muss der Konzern bohren, um mit 99% Sicherheit zweimal Mal Öl zu finden? Verwenden Sie die negative Binomialverteilung.
8. Die Wahrscheinlichkeit beim Eurojackpot alle Zahlen richtig zu haben, beträgt

$$\frac{1}{\binom{50}{5} \times \binom{10}{2}} \approx 1.049 \times 10^{-8}$$

Wie oft müssen Sie jeden Freitag jeweils ein Feld spielen, bis Sie mit mindestens 50% bzw. 99% Wahrscheinlichkeit zum ersten Mal alle Zahlen richtig haben? Was würde Ihnen dies kosten, wenn 1 Feld aktuell 2.20 Euro kosten? Verwenden Sie die negative Binomialverteilung.

9. Stellen Sie die Verteilung $\text{Gamma}(1, \alpha)$ für $\alpha \in \{0.1, 0.5, 1.0, 2.0, 5.0, 10.0\}$ mit Hilfe des Pakets "distr" (Ruckdeschel et al. (2006)) graphisch dar. Die Funktion für die Erzeugung Gamma-verteilter Zufallsvariablen heißt `Gammad`. Beachten Sie hierbei, dass der Gestaltsparameter α dem ersten Argument der Funktion `Gammad` entspricht.
10. Berechnen Sie den Erwartungswert und die Varianz von $\text{Exp}(1)$ **ohne** die expliziten Formeln (4.48) für den Erwartungswert und die Varianz zu verwenden.
11. Das erwartete Geburtsgewicht von gesunden Jungen liegt bei $\mu = 3.35$ kg mit einer Standardabweichung von $\sigma = 0.43$ kg. Wie wahrscheinlich ist es, dass ein gesunder Junge mit weniger als 3 kg Geburtsgewicht geboren wird? Wie lautet der Normbereich für das Geburtsgewicht von Jungen? Verwenden Sie die Normalverteilung.
12. Wir nehmen an, dass sich die Dauer in Tagen zwischen dem Auftreten der Symptome beim Überträger und beim Infizierten im Fall von Sars-CoV-2 (COVID-19) durch eine log-Normalverteilung mit $\mu = 2.78$ und $\sigma = 2.02$ beschreiben lässt. Wie lange dauert es demnach im Median bis die Paare (Überträger und Infizierter) die Symptome zeigen.
13. Sie möchten die Auswirkung von Gamma-Strahlung untersuchen und führen hierzu ein Tierexperiment mit Mäusen durch. Die Mäuse werden dabei einer Strahlung von 2.4 Gray ausgesetzt. Die Überlebenszeit der Mäuse in Wochen kann durch eine Gamma-Verteilung mit Parameter $\sigma = 7$ und $\alpha = 6$ beschrieben werden. Wie wahrscheinlich ist es, dass eine zufällig ausgewählte Maus zwischen 25 und 50 Wochen überlebt? Nach wie vielen Wochen sind 95% der Mäuse verstorben?
14. Die durchschnittliche Lebensdauer von normalen Glühbirnen liegt heute bei 1000 Stunden. Wir nehmen an, dass sich diese durch eine Weibull-Verteilung mit $\sigma = 1250$ und $\alpha = 1.8$ beschreiben lässt. Wie wahrscheinlich ist es, dass eine Glühbirne bereits in den ersten 100 Stunden defekt ist? Nach wie vielen Stunden sind 99% der Glühbirnen defekt?

15. Der maximale Pegelstand [in cm] des Neckars in Rottweil (Baden-Württemberg) lässt sich mit einer Weibull-Verteilung mit Gestaltsparameter $\alpha = 4.5$ und Skalenparameter $\sigma = 270$ beschreiben. Berechnen Sie die Werte für 2-jähriges, 10-jähriges, 50-jähriges und 100-jähriges Hochwasser. Die Werte entsprechen dem 50%, 90%, 98% und 99% Quantil der Verteilung.

5 Schätzen

Dieses Kapitel behandelt das Schätzen von Parametern von einfachen parametrischen Modellen. Es werden folgende Themen behandelt:

- Fragestellungen der induktiven Statistik
- Bedeutung des Schätzens im Kontext der induktiven Statistik
- Parametrische Wahrscheinlichkeitsmodelle
- Punktschätzer, Schätzer
- Biasfreiheit, Effizienz, Konsistenz
- Maximum-Likelihood-Schätzer (kurz: ML-Schätzer)
- Quantil-Quantil-Plot (kurz: qq-Plot)
- Minimum-Distanz-Schätzer (kurz: MD-Schätzer)
- Kolmogorov(-Smirnov)-MD-Schätzer (kurz: KS-MD-Schätzer)
- Cramér-von-Mises-MD-Schätzer (kurz: CvM-MD-Schätzer)
- Asymptotisch Linearer Schätzer (kurz: AL-Schätzer)
- Radius Minimax Schätzer (kurz: RMX-Schätzer)
- Intervallschätzer, Konfidenzintervall
- Konfidenzintervalle für arithmetisches Mittel und Standardabweichung
- exakte Konfidenzintervalle, asymptotische Konfidenzintervalle
- Bootstrap-Konfidenzintervalle
- Konfidenzintervalle für ML-Schätzer
- Stetigkeitskorrektur, Endlichkeitskorrektur
- Konfidenzintervalle für Median und MAD
- Konfidenzintervalle für MD-Schätzer
- Konfidenzintervalle für AL- und RMX-Schätzer

Den R Code für dieses Kapitel befindet sich in der Datei `Schaetzen.Rmd`, welche Sie von meinem GitHub-Account herunterladen können (Link: <https://github.com/stamats/ESDR/blob/master/Schaetzen.Rmd>). Klicken Sie mit der rechten Maustaste auf Raw. Dann können Sie das Ziel speichern unter.... Am wenigsten Schwierigkeiten ergeben sich, wenn Sie meine R Markdown Dateien im selben Ordner wie die Daten, welche in den jeweiligen Kapiteln verwendet werden, speichern.

Wir installieren zunächst die in diesem Kapitel benötigten Pakete. Da das Paket "RobLox" (Kohl (2019)) vom Bioconductor Paket "Biobase" (Huber et al. (2015)) abhängt, welches üblicherweise nicht automatisch mitinstalliert wird, beginnen wir damit. Wir benötigen hierfür außerdem das Paket "BiocManager" (Morgan (2019)).

```
1 install.packages("BiocManager")
2 BiocManager::install("Biobase", update = FALSE)
```

Nun können wir die benötigten Pakete installieren.

```
1 install.packages(c("distrMod", "qqplotr", "RobLox", "gridExtra", "MKinfer",
2                     "ROptEst", "RobExtremes", "MKpower", "MKclass"))
```

Außerdem installieren wir das Paket "rmx" (Kohl (2022c)) von GitHub, welches aktuell noch nicht auf CRAN ist.

```
1 ## Development Version von GitHub
2 install.packages("remotes")
3 remotes::install_github("stamats/rmx", upgrade = "never",
4                         build = FALSE, build_vignettes = TRUE)
```

Achten Sie darauf, dass Sie die Pakete der vorhergehenden Kapitel 2, 3 und 4 bereits installiert haben. Wir laden alle Pakete, die wir in diesem Kapitel verwenden werden.

```
1 library(ggplot2)
2 library(MKdescr)
3 library(distrMod)
4 library(qqplotr)
5 library(RobLox)
6 library(gridExtra)
7 library(MKinfer)
8 library(ROptEst)
9 library(RobExtremes)
10 library(MKpower)
11 library(MKclass)
12 library(rmx)
13 library(MASS)
14 library(boot)
15 library(parallel)
```

Wie bereits in Abschnitt 2.4 erklärt, ist ein wiederholtes Ausführen von `library` unproblematisch. Die Pakete "MASS" (Venables and Ripley (2002)), "boot" (Canty and Ripley (2021)) und "parallel" (R Core Team (2022a)) müssen nicht installiert werden, da diese in der Standardinstallation von R enthalten sind (vgl. Abschnitt 1.2).

5.1 Einführung

Diese Einführung enthält ein kurzes Beispiel, welches deutlich machen soll, welche Fragen wir mit Hilfe der induktiven Statistik beantworten können.

Wir betrachten eine Münze und bezeichnen der Einfachheit halber die Seiten mit 0 und 1, wobei wir ausschließen, dass die Münze bei einem Wurf auf ihrem Rand landen könnte. Die Frage, die uns interessiert, ist:

Handelt es sich um eine faire Münze?

Dabei bedeutet **fair**, dass beide Seiten mit gleicher Wahrscheinlichkeit auftreten. Wir können den Münzwurf mit Hilfe der Bernoulli-Verteilung $Bernoulli(p)$ beschreiben, wobei p die Wahrscheinlichkeit sei, dass die Seite 1 geworfen wird. Mit Hilfe dieses Wahrscheinlichkeitsmodells können wir die Fragestellung präzisieren und erhalten:

Ist die Wahrscheinlichkeit für Seite 1 gleich 50%, kurz: $p = 0.5$?

Wie können wir diese Frage beantworten? Wir könnten die Münze einer sehr genauen Materialanalyse unterziehen. Das wäre aber sicher recht aufwendig und nur mit einer geeigneten technischen Ausstattung möglich. Schneller und einfacher ist sicher ein Zufallsexperiment: Wir werfen die Münze mehrfach und notieren die Ergebnisse. Die Ergebnisse dieses Zufallsexperiments bilden unsere Stichprobe, welche die Basis für unsere Entscheidung mittels statistischer Verfahren ist.

Bevor wir dieses Zufallsexperiment durchführen, sind noch einige Fragen zu klären:

- I: Wie oft sollten wir die Münze werfen, um ein möglichst verlässliches Ergebnis zu bekommen?
- II: Wie fassen wir die Ergebnisse zusammen, damit wir in möglichst optimaler Weise auf die tatsächliche Wahrscheinlichkeit p für die Seite 1 schließen können?
- III: Liegt die beobachtete Anzahl für die Seite 1 im Bereich der erwarteten Häufigkeit für eine faire Münze oder ist diese zu klein oder zu groß?

Die Antworten der induktiven Statistik auf die obigen Fragen lauten:

- Zu I:** Wir können unter Verwendung eines Konfidenzintervalls (vgl. Abschnitt 5.3) oder eines statistischen Tests (vgl. Kapitel 6) eine sogenannte Fallzahlplanung durchführen. Damit können wir die Anzahl der Versuche so festlegen, dass wir mit vorgegebener Sicherheit erkennen, ob es sich um eine faire Münze handelt oder nicht.
- Zu II:** Mit Hilfe sogenannter Punktschätzer (vgl. Abschnitt 5.2) können wir die beobachteten Werte zusammenfassen. Im Fall der Münze könnte die beobachtete relative Häufigkeit der Würfe mit Seite 1 mit dem theoretischen Wert ($p = 0.5$) verglichen werden.

Zu III: Wir können erneut Konfidenzintervalle oder statistische Tests verwenden, um diese Frage mit vorgegebener hoher Sicherheit richtig beantworten zu können. Liegt etwa $p = 0.5$ im Inneren des berechneten Konfidenzintervalls, so gehen wir von einer fairen Münze aus.

Anmerkung:

Die Statistik ist nicht in der Lage, eine Frage mit absoluter Sicherheit zu beantworten. Die Möglichkeit einer Fehlentscheidung kann niemals ausgeschlossen werden. Das geht soweit, dass manche Wissenschaftler der Meinung sind, dass die Mehrzahl der publizierten Forschungsergebnisse falsch sind (Ioannidis (2005)). Dieser Kritik kann man mit einer sorgsamen methodischen Vorgehensweise, ausreichend großen Studien, dem korrekten Umgang mit statistischen Verfahren und der vorsichtigen Interpretation der Ergebnisse begegnen. Dies alles sollte zudem vollständig und umfassend dokumentiert werden, um die Forschung auch für andere wiederholbar und somit reproduzierbar zu machen (Gentleman and Temple Lang (2007)).

5.2 Punktschätzer

In diesem Abschnitt wollen wir die unbekannten Parameter von einfachen parametrischen Modellen bestimmen. Dieser Vorgang wird als Schätzen bezeichnet, genauer sind wir auf der Suche nach Punktschätzern für die unbekannten Parameter. Wir definieren zunächst, was wir unter einem parametrischen Modell und einem Punktschätzer verstehen.

Definition 5.1 (Parametrisches Modell, Punktschätzer). *(a) Ein **parametrisches Modell** ist eine Menge $\mathcal{P} = \{P_\theta \mid \theta \in \Theta\}$ von Wahrscheinlichkeitsverteilungen, wobei die Elemente der Menge durch ihren Parameter $\theta \in \Theta \subset \mathbb{R}^k$ ($k \in \mathbb{N}$) eindeutig identifiziert werden können. Man nennt dies auch eine **parametrische Familie**.*

*(b) Gegeben sei eine parametrische Familie von Wahrscheinlichkeitsverteilungen $\mathcal{P} = \{P_\theta \mid \theta \in \Theta\}$, wobei $\Theta \subset \mathbb{R}^k$ ($k \in \mathbb{N}$) die Menge der möglichen Parameter darstellt. Weiter sei x_1, \dots, x_n eine **repräsentative Stichprobe** der Länge $n \in \mathbb{N}$ für ein Element $P_\theta \in \mathcal{P}$ (θ unbekannt). Dann ist ein **Schätzer** S_n eine Zufallsvariable*

$$S_n : \mathbb{R}^n \rightarrow \Theta, (x_1, \dots, x_n) \mapsto S_n(x_1, \dots, x_n) =: \hat{\theta} \quad (5.1)$$

wobei $\hat{\theta}$ die **Schätzung** für θ ist. Man nennt dies auch einen **Punktschätzer** bzw. eine **Punktschätzung**.

Wir geben einige zusätzliche Erläuterungen.

Bemerkung 5.2. *(a) Der Begriff **parametrische Familie** impliziert, dass man die Elemente der Menge anhand ihres Parameters identifizieren kann. Formal betrachtet gibt es also eine Funktion, welche ein gegebenes θ einem P_θ zuordnet. Entsprechend führt jedes θ zu einem eindeutigen P_θ .*

(b) Die Beobachtungen in der repräsentativen Stichprobe entsprechen den Realisationen von unabhängigen Zufallsvariablen X_1, \dots, X_n . Für die Zufallsvariablen wiederum gilt: $X_i \sim P_\theta$ ($i = 1, \dots, n$). Man spricht daher auch von unabhängig und identisch verteilten (uiv) Zufallsvariablen.

(c) Es handelt sich bei einem Punktschätzer S_n um eine Zufallsvariable; d.h., eine zufällige Funktion. Dementsprechend besitzt der Schätzer eine Verteilung, die von der unbekannten Verteilung P_θ abhängt.

Für die Beurteilung der Güte eines Schätzers wird meist $E(S_n)$ und $\text{Var}(S_n)$ herangezogen. Ist $E(S_n) = \theta$, so nennt man den Schätzer **biasfrei** oder **erwartungstreu**. Das bedeutet anschaulich, dass der Schätzer im Mittel den richtigen Parameter schätzt. Ist zusätzlich $\text{Var}(S_n)$ minimal, wird der Schätzer **effizient** genannt. Es gibt demnach keinen erwartungstreuen Schätzer, der θ genauer schätzen kann. Da es gerade bei komplexeren Modellen üblicherweise schwierig oder zum Teil sogar unmöglich ist, effiziente Schätzer zu finden, wird in diesem Fall oft der **mittlere quadratische Fehler (MSE)** herangezogen

$$\text{MSE}(S_n) = E(S_n - \theta)^2 = (E(S_n - \theta))^2 + \text{Var}(S_n) = \text{Bias}(S_n)^2 + \text{Var}(S_n)$$

und minimiert. Anstelle der Biasfreiheit muss man sich häufig mit der sogenannten **Konsistenz** begnügen. Konsistenz bedeutet, dass sich der Schätzer zumindest für eine wachsende Anzahl von Beobachtungen (in einem wahrscheinlichkeitstheoretischen Sinn) immer mehr dem tatsächlichen (unbekannten) Parameter annähert; d.h., $\lim_{n \rightarrow \infty} S_n = \theta$ (in einem wahrscheinlichkeitstheoretischen Sinn). Die Abbildung 5.1 soll die Begriffe biasfrei und effizient veranschaulichen, wobei der Kreismittelpunkt für den gesuchten unbekannten Parameter steht. Für weitere Details verweisen wir auf Abschnitt 6.3 von Hedderich und Sachs (2018).

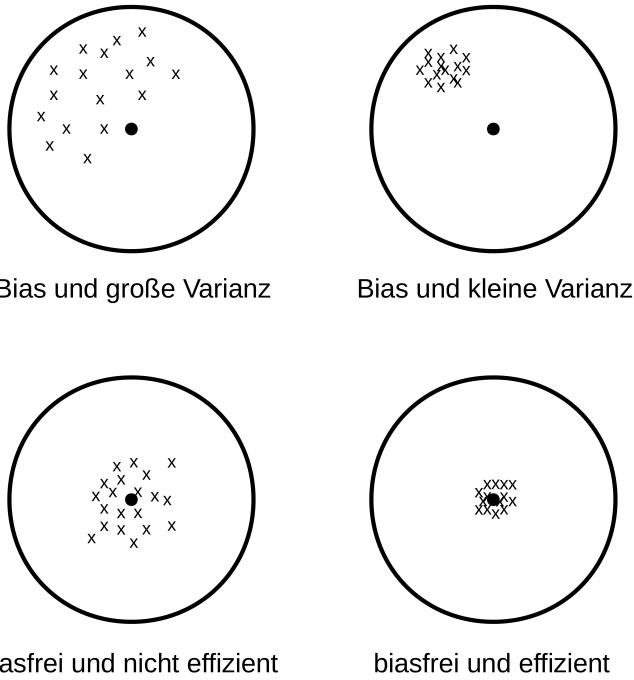


Abbildung 5.1: Veranschaulichung von biasfrei und effizient.

Im folgenden Beispiel stellen wir einige biasfreie und effiziente Schätzer vor.

Beispiel 5.3. (a) Wir betrachten das Wahrscheinlichkeitsmodell $\{\text{Bernoulli}(p) \mid p \in (0, 1)\}$. Ein biasfreier und effizienter Schätzer für die unbekannte Wahrscheinlichkeit p ist die relative Häufigkeit.

(b) Gegeben sei das Wahrscheinlichkeitsmodell $\{\mathcal{N}(\mu, \sigma^2) \mid \mu \in \mathbb{R}\}$, wobei $\sigma^2 \in (0, \infty)$ bekannt sei. Dann ist das arithmetische Mittel ein biasfreier und effizienter Schätzer für den unbekannten Erwartungswert μ .

(c) Nicht mehr ganz so einfach ist die Situation für das Modell $\{\mathcal{N}(\mu, \sigma^2) \mid \mu \in \mathbb{R}, \sigma \in (0, \infty)\}$. Die Stichprobenvarianz ist ein möglicher Schätzer für die unbekannte Varianz σ^2 . Jedoch ist diese nicht biasfrei. Der Bias beträgt $-\frac{1}{n}\sigma^2$. Wir können auch einen biasfreien Schätzer angeben, indem wir die Standardisierung $\frac{1}{n-1}$ verwenden. Wir erhalten

$$\tilde{S}_n(x_1, \dots, x_n) = \frac{1}{n-1} \sum_{i=1}^n (x_i - \text{AM}(x_1, \dots, x_n))^2 \quad (5.2)$$

Die Vermeidung eines Bias ist also der Grund, warum bei der Berechnung der empirischen Varianz oft $\frac{1}{n-1}$ anstelle von $\frac{1}{n}$ verwendet wird. Was die Genauigkeit der Schätzung betrifft, ist die Varianz der Stichprobenvarianz kleiner als die Varianz von \tilde{S}_n .

Wir verwenden den ITS-Datensatz und wollen damit die Prävalenz (Krankheitshäufigkeit) von Leberversagen für ITS-Patienten schätzen. Importieren Sie den Datensatz wie in Abschnitt 2.3 beschrieben, falls Sie dies nicht bereits getan haben. Dort finden Sie auch weitere Informationen zu den Daten. Im Unterschied zur deskriptiven Statistik ist es für die Gültigkeit der folgenden Ergebnisse notwendig, dass es sich um 500 zufällig und für die ITS-Population repräsentativ ausgewählte Patienten handelt. Wir verwenden die relative Häufigkeit wie bereits in Abschnitt 2.5.1 beschrieben.

```
1 ITSDaten <- read.csv2(file = "ITSDaten.csv", fileEncoding = "utf8",
2                         stringsAsFactors = TRUE)
3 ## biasfrei und effizient
4 table(ITSDaten$Leberversagen) / nrow(ITSDaten)
```

0	1
0.96	0.04

Demnach haben 4% der ausgewählten ITS-Patienten ein Leberversagen. Dies ist ein erster Schätzwert für alle ITS-Patienten, den wir später noch weiter absichern werden. Ein mögliches Modell für die Prävalenz von Leberversagen wäre demnach Bernoulli (0.04).

Die Analyse in Abschnitt 2.6.1 legt nahe, dass sich die maximale Körpertemperatur der ITS-Patienten – außer für stark unterkühlte (hypothermische) Patienten wie Patient 398 – recht gut durch eine Normalverteilung beschreiben lässt. Wir schätzen den Erwartungswert und die Varianz.

```
1 ## biasfrei und effizient
2 mean(ITSDaten$Temperatur[-398])
```

[1] 37.72044

```
1 ## biasfrei
2 sd(ITSDaten$Temperatur[-398])
```

[1] 1.173187

Wir haben diese Ergebnisse bereits in Abschnitt 2.6.1 erhalten. Im Unterschied dazu sehen wir die Werte jetzt aber nicht mehr als Werte zur Beschreibung der Stichprobe, sondern als Parameter eines Wahrscheinlichkeitsmodells, welches die zugrunde liegenden Population beschreibt.

Anmerkung:

Für die Interpretation der Ergebnisse und für den Rückschluss auf die ITS-Population ist es hier ganz entscheidend, ob wir stark unterkühlte Patienten wie Patient 398 mit einschließen wollen. Falls dies nicht der Fall ist, können wir $\text{Norm}(37.7, 1.2^2)$ als Modell für die maximale Körpertemperatur annehmen. Andernfalls müssen wir davon ausgehen, dass sich die maximale Körpertemperatur nicht durch eine Normalverteilung beschreiben lässt, da ein derart extremer Wert von 9.1°C in diesem Fall praktisch unmöglich ist.

Unter Verwendung des geschätzten Modells ist die Wahrscheinlichkeit für eine maximale Körpertemperatur von weniger als 10°C gerade einmal

```
1 pnorm(10, mean = 37.7, sd = 1.2)
```

```
[1] 3.404114e-118
```

Mit der nächsten Frage, mit der wir uns beschäftigen werden, ist, wie finden wir optimale oder zumindest gute Schätzer. Man nennt dies auch **Schätzerkonstruktion**. Das wohl am häufigsten verwendete Prinzip ist Maximum-Likelihood, welches wir in der folgenden Definition angeben.

Definition 5.4 (Maximum-Likelihood-Schätzer). *Es sei $\mathcal{P} = \{P_\theta \mid \theta \in \Theta\}$, $\Theta \subset \mathbb{R}^k$ ($k \in \mathbb{N}$), ein Wahrscheinlichkeitsmodell mit Wahrscheinlichkeitsfunktion bzw. Dichte d_θ . Weiter seien x_1, \dots, x_n die Realisationen von unabhängig und P_θ verteilten Zufallsvariablen X_1, \dots, X_n . Dann lautet die Likelihood-Funktion*

$$L(\theta) = \prod_{i=1}^n d_\theta(x_i) \quad (5.3)$$

und der **Maximum-Likelihood-Schätzer** (kurz: ML-Schätzer) für θ ist die Maximalstelle von $L(\theta)$.

Wir geben einige zusätzliche Erläuterungen.

Bemerkung 5.5. (a) Im Fall des ML-Schätzers wird demnach θ so gewählt, dass die beobachteten Daten die für das angenommene Wahrscheinlichkeitsmodell maximal mögliche Wahrscheinlichkeit haben.

(b) Das ML-Konstruktionsprinzip ist allgemein anwendbar und führt in der Regel auf einen (asymptotisch) biasfreien und effizienten Schätzer. Es gibt jedoch auch Wahrscheinlichkeitsmodelle, für die es nicht einsetzbar ist.

(c) In einfachen Fällen, kann man den ML-Schätzer durch analytische Berechnungen explizit bestimmen. Die numerische Berechnung der Likelihood-Funktion ist in der Praxis schwierig (Produkt vieler kleiner Zahlen) und man verwendet üblicherweise die sogenannte **Log-Likelihood-Funktion**

$$l(\theta) = \ln(L(\theta)) = \sum_{i=1}^n \ln(d_\theta(x_i)) \quad (5.4)$$

welche dieselbe Maximalstelle wie $L(\theta)$ besitzt. Dieser einfache "Trick" vereinfacht die analytischen und numerischen Berechnungen deutlich und führt zu numerisch stabileren Ergebnissen.

(d) Es gibt eine Reihe von R Paketen, welche Funktionen für die Berechnung von ML-Schätzern enthalten. Für einfache Wahrscheinlichkeitsmodelle kann man zum Beispiel die Pakete "stats4" (R Core Team (2022a)), "MASS" (Venables and Ripley (2002)), "fitdistrplus" (Delignette-Muller and Dutang (2015)) oder "distrMod" (Kohl and Ruckdeschel (2010)) verwenden.

Wir geben einige Beispiele für ML-Schätzer.

Beispiel 5.6. **(a)** Im Fall des recht einfachen Bernoulli-Modells $\mathcal{P} = \{\text{Bernoulli}(p) \mid p \in (0, 1)\}$ lässt sich der ML-Schätzer explizit über die erste Ableitung der Log-Likelihood Funktion bestimmen. Es ergibt sich als Likelihood-Funktion

$$L(p) = \prod_{i=1}^n p^{x_i} (1-p)^{1-x_i} \quad (5.5)$$

und somit als Log-Likelihood-Funktion

$$l(p) = \sum_{i=1}^n \ln [p^{x_i} (1-p)^{1-x_i}] = \sum_{i=1}^n [x_i \ln(p) + (1-x_i) \ln(1-p)] \quad (5.6)$$

Wir leiten die Log-Likelihood-Funktion ab und erhalten

$$\begin{aligned} l'(p) &= \frac{d}{dp} l(p) = \sum_{i=1}^n \left[\frac{x_i}{p} - \frac{1-x_i}{1-p} \right] = \frac{1}{p} \sum_{i=1}^n x_i - \frac{1}{1-p} \left[n - \sum_{i=1}^n x_i \right] \\ &= -\frac{n}{1-p} + \frac{(1-p)+p}{p(1-p)} \sum_{i=1}^n x_i \\ &= -\frac{n}{1-p} + \frac{1}{p(1-p)} \sum_{i=1}^n x_i \end{aligned} \quad (5.7)$$

Das Nullsetzen der ersten Ableitung ($l'(p) = 0$) führt auf

$$\frac{n}{1-p} = \frac{1}{p(1-p)} \sum_{i=1}^n x_i \quad \Leftrightarrow \quad p = \frac{1}{n} \sum_{i=1}^n x_i \quad (5.8)$$

Da x_i nur die Werte 0 und 1 annehmen kann, ist das arithmetische Mittel der x_i gerade die relative Häufigkeit von 1.

(b) Normalverteilungsmodell: Der ML-Schätzer für den Erwartungswert ist das arithmetische Mittel, für die Varianz ist es die Stichprobenvarianz (d.h. Standardisierung $\frac{1}{n}$).

(c) Poisson-Modell: Der ML-Schätzer ist das arithmetische Mittel.

(d) Exponentialmodell: Der ML-Schätzer ist der Kehrwert des arithmetischen Mittels.

Wir verwenden jetzt anstelle der expliziten Funktionen `mean` und `sd` die Funktion `fitdistr` aus dem Paket "MASS" (Venables and Ripley (2002)), um den ML-Schätzer für die maximale Körpertemperatur zu bestimmen. Wir verwenden als Modell wieder die Normalverteilung und schließen Patient 398 aus.

```
1 fitdistr(ITSDaten$Temperatur[-398], densfun = "normal")
```

mean	sd
37.72044088	1.17201119
(0.05246643)	(0.03709937)

Da der ML-Schätzer für die Varianz die Standardisierung $\frac{1}{n}$ enthält, weicht das Ergebnis etwas von der Schätzung mit Hilfe von `sd` ab. Wir erhalten zusätzlich zu den Schätzwerten noch eine weitere Ausgabe. Es handelt sich dabei um den **Standardfehler** für die Schätzwerte. Wir werden dies in Abschnitt 5.3 noch genauer besprechen.

Alternativ verwenden wir das Paket "distrMod" (Kohl and Ruckdeschel (2010)). Das Paket leitet sich vom Paket "distr" (Ruckdeschel et al. (2006)) ab. Hier erfolgt die Schätzung in zwei Schritten. Zunächst müssen wir das Wahrscheinlichkeitsmodell anlegen und anschließend können wir mit Hilfe der Funktion `MLEstimator` für das erzeugte Modell die Schätzung durchführen.

```
1 ## Ausgaboption ändern
2 distrModOptions(show.details = "minimal")
3 ## Anlegen des Modells
4 Modell <- NormLocationScaleFamily()
5 ## ML-Schätzung
6 MLEstimator(ITSData$Temperatur[-398], Modell)
```

Evaluations of Maximum likelihood estimate:	

mean	sd
37.72044088	1.17201119
(0.05246643)	(0.03709937)

Der recht komplizierte Name `NormLocationScaleFamily` für das Normalverteilungsmodell kommt daher, dass es sich dabei allgemeiner um ein **Lokations- und Skalenmodell** handelt, da sowohl der Erwartungswert (Lageparameter oder Lokation) als auch die Varianz (Streuungsparameter oder Skala) zu schätzen sind. Die ML-Schätzung (estimate) ist identisch mit dem Ergebnis von `fitdistr`.

Anmerkung:

Die abstrakte objektorientierte Implementation im Paket "distrMod" (Kohl and Ruckdeschel (2010)) ermöglicht die Berechnung einiger zusätzlicher Größen, die zum Beispiel für die Berechnung von Konfidenzintervallen (vgl. Abschnitt 5.3) oder zur Diagnostik herangezogen werden können. Mit Hilfe der Funktion `distrModOptions` und der Option `show.details = "minimal"` wurden die zugehörigen zusätzlichen Ausgaben bewusst unterdrückt. Dies dient der Übersichtlichkeit und dem besseren Verständnis der Ausgaben. Die zusätzlichen Informationen würden zum Teil Erklärungen erfordern, die den Inhalt dieses Buches deutlich übersteigen.

Zwar hat die Schätzung der unbekannten Parameter problemlos funktioniert, dies heißt aber nicht, dass man den Schätzwerten blind vertrauen kann und das gefittete Modell auch tatsächlich zu den Daten passt. Bevor statistische Ergebnisse und geschätzte Modelle weiter interpretiert werden, sollte immer eine **Modellvalidierung** durchgeführt werden. Man spricht auch von **Modelldiagnostik**. Für die Gültigkeit

jeder statistischen Analyse sind jeweils gewisse Voraussetzungen nötig, welche mehr oder weniger strikt sein können. Das Ziel dieses Schrittes ist es, die Gültigkeit dieser Annahmen und damit der statistischen Ergebnisse zu überprüfen, was in den allermeisten Fällen schwierig ist. Im Fall, dass hierzu neue Daten erhoben und zur Überprüfungen herangezogen werden, spricht man auch von **externer Validierung**. Basiert die Validierung auf den gleichen Daten, die auch für die Schätzung verwendet wurden, so nennt man dies auch eine **interne Validierung**.

Anmerkung:

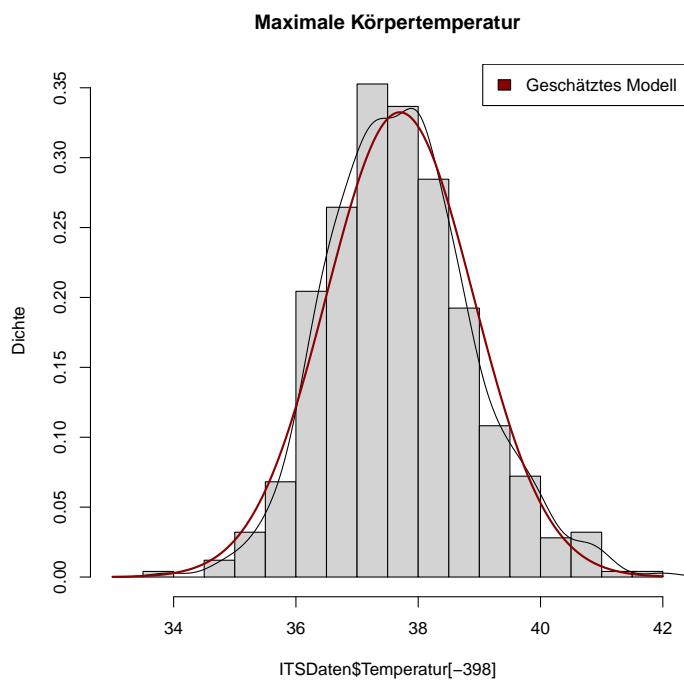
Bei der internen Validierung sollte man sehr vorsichtig vorgehen, da jedes statistische Verfahren in gewisser Weise auch versucht, das Modell bzw. die Voraussetzungen den Daten “überzustülpen”. Folglich besteht die Gefahr, die Abweichungen vom Modell bzw. den Voraussetzungen zu unterschätzen. Man spricht auch von einem **(Re-)Substitutions-Bias**. Die übliche Vorgehensweise, um diesen Bias zu reduzieren bzw. zu umgehen, ist der Einsatz von sog. **Resamplingmethoden**. Bei diesen Verfahren (z.B. Kreuzvalidierung oder Bootstrap) wählt man üblicherweise zufällig einen Teil des Datensatzes aus und führt auf diesem Teil das statistische Verfahren durch. Der verbleibende Teil dient zur Validierung. Da es sich um eine Zufallsauswahl handelt, kann man dieses Vorgehen viele Male wiederholen und bekommt so einen Eindruck von der Gültigkeit der statistischen Ergebnisse sowie auch von deren Variabilität. Bei einfachen Modellen, wie im vorliegenden Fall, wird aber meist auf Resamplingmethoden verzichtet. Daher werde auch ich diesen Ansatz hier nicht weiterverfolgen.

Manche Voraussetzungen können mit Hilfe statistischer Tests überprüft werden. Davon rate ich aber eher ab, da meist unklar ist, welche Power (vgl. Kapitel 6) diese Tests für die entsprechende Situation haben und da diese Tests selbst eigene Voraussetzungen benötigen, damit sie den Fehler 1. Art (vgl. Kapitel 6) einhalten und valide Ergebnisse liefern. Wir wollen im folgenden stattdessen diagnostische Plots verwenden, welche aus meiner Sicht die beste Möglichkeit für die Modellvalidierung darstellen. Im vorliegenden Beispiel haben wir es mit einem einfachen parametrischen Modell, nämlich der Normalverteilung, zu tun. Da die Dichte, aber auch die (kumulative) Verteilungsfunktion und die Quantilsfunktion definierende Funktionen für die Normalverteilung sind, ist es naheliegend, sich diese Funktionen für das geschätzte Modell anzusehen und mit den entsprechenden empirischen Gegenstücken, die aus den vorliegenden Daten gewonnen werden, zu vergleichen. Wir beginnen mit der Dichte und stellen hierzu die Dichte des geschätzten Normalverteilungsmodells zusammen mit einem Histogramm und der empirischen Dichte der Daten grafisch dar, wobei wir Patient 398 von der Betrachtung ausschließen.

```

1 hist(ITSDaten$Temperatur[-398], breaks = seq(from = 33, to = 42, by = 0.5),
2       main = "Maximale Körpertemperatur", ylab = "Dichte", freq = FALSE)
3 lines(density(ITSDaten$Temperatur[-398]))
4 curve(dnorm(x, mean = 37.7, sd = 1.2), col = "darkred", from = 33, to = 42,
5        n = 501, add = TRUE, lwd = 2)
6 legend("topright", fill = "darkred", legend = "Geschätztes Modell")

```

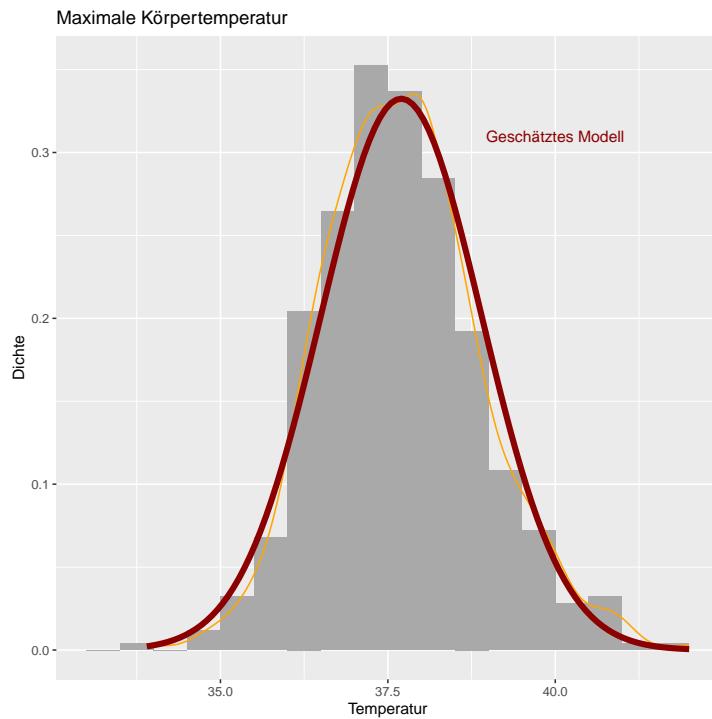


Mit dem Parameter `lwd` steuern wir die Linienstärke, wobei der Defaultwert gleich 1 ist und Werte größer 1 zu dickeren Linien führen. Wir wiederholen die Abbildung mit der Funktionen aus dem Paket "ggplot2" (Wickham (2009)). Wir benötigen hierfür neben den Funktionen `ggplot`, `geom_histogram` und `geom_density` zusätzlich die Funktion `stat_function`, mit der wir Funktionen zu einer Grafik hinzufügen können. Mit der Funktion `annotate` beschriften wir die Grafik zusätzlich.

```

1 ggplot(ITSDaten[-398,], aes(x=Temperatur)) +
2   geom_histogram(aes(y=after_stat(density)),
3                 breaks = seq(from = 33, to = 42, by = 0.5),
4                 fill = "darkgrey") +
5   geom_density(color = "orange") + ylab("Dichte") +
6   stat_function(fun = dnorm, args = list(mean = 37.7, sd = 1.2),
7                 color = "darkred", linewidth = 2) +
8   annotate("text", x = 40, y = 0.31, col = "darkred",
9           label = "Geschätztes Modell")+
10  ggttitle ("Maximale Körpertemperatur")

```



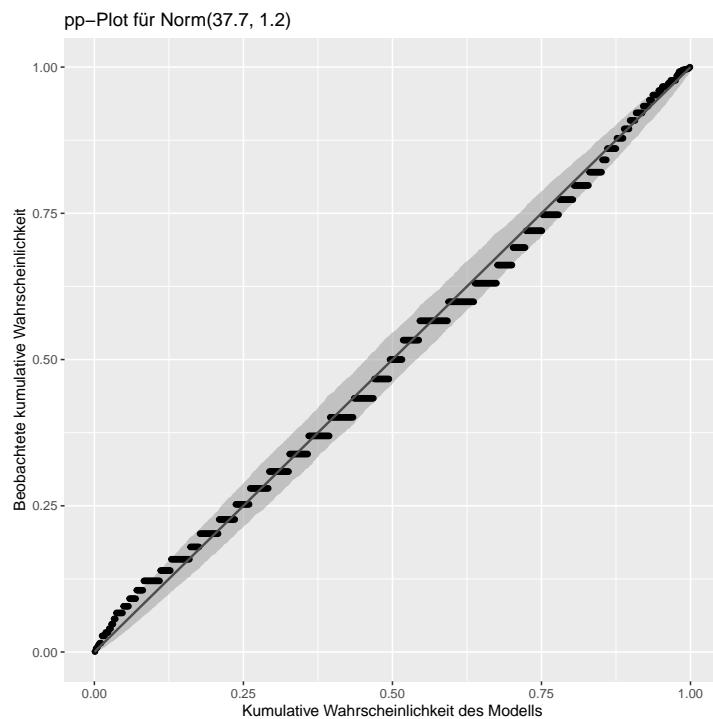
Wir sehen eine recht gute Übereinstimmung zwischen den Daten und dem Modell. Gewisse zufällige Abweichungen von den Modellannahmen sind immer zu erwarten und erschweren die Beurteilung der Gültigkeit zusätzlich. Es ist umgekehrt sogar so, dass eine zu perfekte Übereinstimmung zwischen dem Modell und den Daten ein Indiz für eine Manipulation der Daten sein kann. So wird etwa seit der Arbeit von Fisher (1936), der die Ergebnisse von Mendel (1866) in Frage stellt, diskutiert, ob Mendel seine gewonnenen Ergebnisse zur Vererbungslehre nicht etwas an seine Erwartungen angepasst hat; siehe auch Novitski (2004).

In ähnlicher Weise wie im Fall der Dichten kann man auch die empirische Verteilungsfunktion mit der Verteilungsfunktion des geschätzten Modells vergleichen. Man nennt dies einen **pp-Plot** (probability-probability-Plot). Ein entsprechender Plot kann etwa mit Hilfe des Paketes "qqplotr" (Almeida et al. (2018)) erstellt werden, welches die Funktionen `stat_pp_point`, `stat_pp_line` und `stat_pp_band` für die Verwendung in Kombination mit dem "ggplot2" (Wickham (2009)) zur Verfügung stellt.

```

1 ggplot(ITSDaten[-398,], aes(sample = Temperatur)) +
2   qqplotr:::stat_pp_band(dparams = list(mean = 37.7, sd = 1.2)) +
3   qqplotr:::stat_pp_point(dparams = list(mean = 37.7, sd = 1.2)) +
4   qqplotr:::stat_pp_line() +
5   xlab("Kumulative Wahrscheinlichkeit des Modells") +
6   ylab("Beobachtete kumulative Wahrscheinlichkeit") +
7   ggtitle("pp-Plot für Norm(37.7, 1.2)")

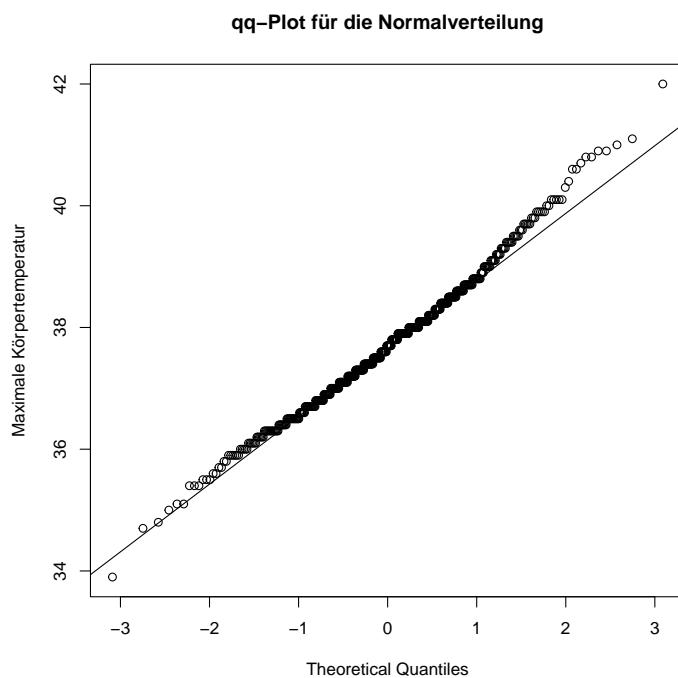
```



Wir sehen erneut eine gute Übereinstimmung zwischen dem Modell und den Daten. Die Punkte reihen sich sehr gut entlang der eingezeichneten Winkelhalbierenden auf und befinden sich zu einem sehr großen Teil innerhalb des 95% Konfidenzbandes, welches aus punktweisen Bootstrap-Konfidenzintervallen gebildet wird. Für Details zu Konfidenzintervallen und zu Bootstrap-Konfidenzintervallen im Besonderen verweisen wir auf Abschnitt 5.3.

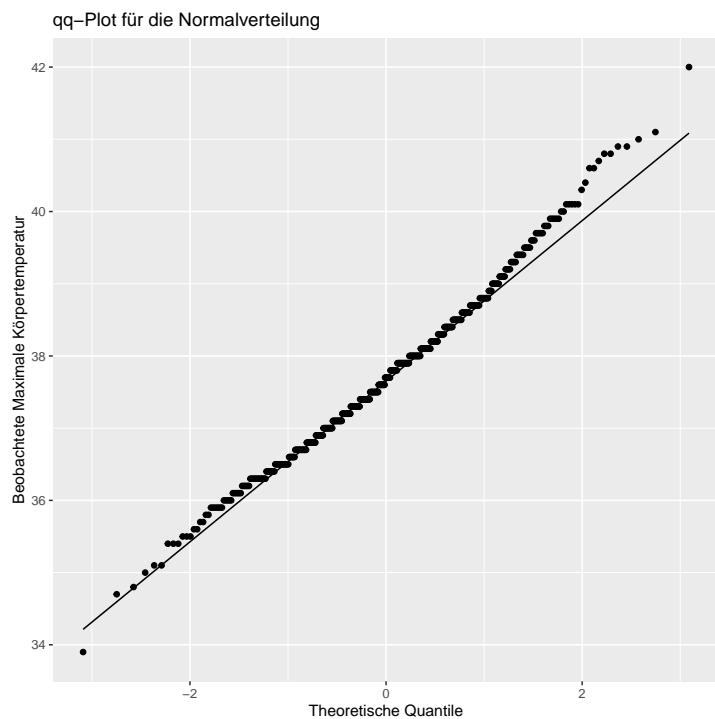
Schließlich wollen wir auch noch die Quantilfunktionen miteinander vergleichen. Der sogenannte Quantil-Quantil-Plot (kurz: **qq-Plot**) ist wohl der am häufigsten für derartige Vergleiche eingesetzte Plot. In diesem Plot werden die empirischen und theoretischen Quantile miteinander verglichen. Ähnlich wie im Fall des pp-Plots ist die Übereinstimmung zwischen dem Modell und den Daten umso besser, je näher die Punkte zur eingezeichneten Gerade liegen. Im Fall der Normalverteilung können wir in R hierfür die Funktionen `qqnorm` und `qqline` verwenden.

```
1 qqnorm(ITSDaten$Temperatur[-398], main = "qq-Plot für die Normalverteilung",
2         ylab = "Maximale Körpertemperatur")
3 qqline(ITSDaten$Temperatur[-398])
```



Alternativ können wir auch die Funktionen `stat_qq` und `stat_qq_line` aus dem Paket "ggplot2" (Wickham (2009)) verwenden. Da es auch eine Funktion `stat_qq_line` im Paket "qqplotr" (Almeida et al. (2018)) gibt, stellt die zusätzliche Angabe von `ggplot2::` sicher, dass die Funktion aus dem Paket "ggplot2" verwendet wird.

```
1 ggplot(ITS Daten[ -398 ,], aes(sample = Temperatur)) +  
2   stat_qq() + ggplot2:::stat_qq_line() +  
3   xlab("Theoretische Quantile") +  
4   ylab("Beobachtete Maximale Körpertemperatur") +  
5   ggtitle("qq-Plot für die Normalverteilung")
```

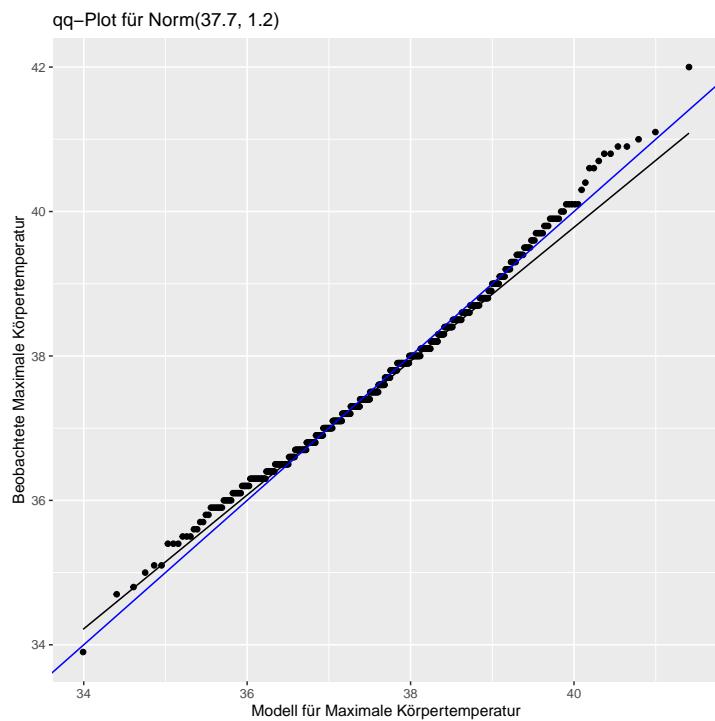


Mit diesem Plot können wir ganz generell überprüfen, ob die Daten von einer Normalverteilung kommen können. Im vorliegenden Fall sehen wir, dass wir etwas mehr erhöhte Temperaturen beobachten als wir im Fall der Normalverteilung erwarten würden. Wollen wir die Daten mit einer ganz konkreten Verteilung vergleichen, so können wir im Fall des Paketes "ggplot2" (Wickham (2009)) zusätzlich auch die entsprechenden Parameter angeben.

```

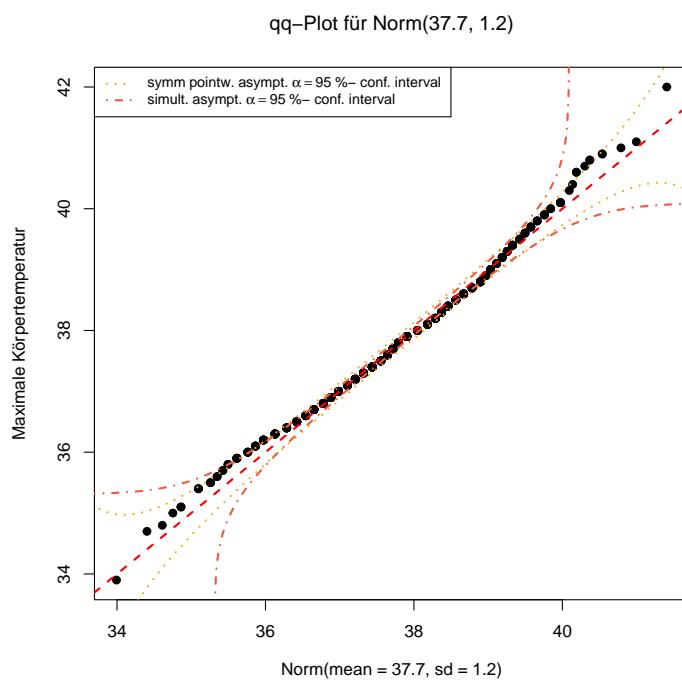
1 ggplot(ITSData[ -398 ,], aes(sample = Temperatur)) +
2   stat_qq(dparams = list(mean = 37.7 , sd = 1.2 )) +
3   ggplot2:: stat_qq_line(dparams = list(mean = 37.7 , sd = 1.2 )) +
4   geom_abline(slope = 1, color = "blue") +
5   xlab("Modell für Maximale Körpertemperatur") +
6   ylab("Beobachtete Maximale Körpertemperatur") +
7   ggtitle("qq-Plot für Norm(37.7 , 1.2 )")

```



Die zusätzlich in blau eingezeichnete Winkelhalbierende zeigt, dass die Standard-qq-Linie (basiert auf dem 1. und 3. Quartil) nicht identisch zur Winkelhalbierenden sein muss, wobei Abweichungen von der Winkelhalbierenden ebenfalls auf Abweichungen zwischen dem Modell und den Daten hinweisen. Daher empfiehlt es sich, zusätzlich die Winkelhalbierende einzuzeichnen bzw. die Punkte auf die Winkelhalbierende zu beziehen. Eine alternative Darstellung ist mit der Funktion `qqplot` aus dem Paket "distr" (Ruckdeschel et al. (2006)) möglich. In diesem Fall entspricht die eingezeichnete Gerade der Winkelhalbierenden.

```
1 qqplot(ITS Daten$Temperatur[-398], Norm(mean = 37.7, sd = 1.2),
2         xlab = "Maximale Körpertemperatur",
3         main = "qq-Plot für Norm(37.7, 1.2)")
```

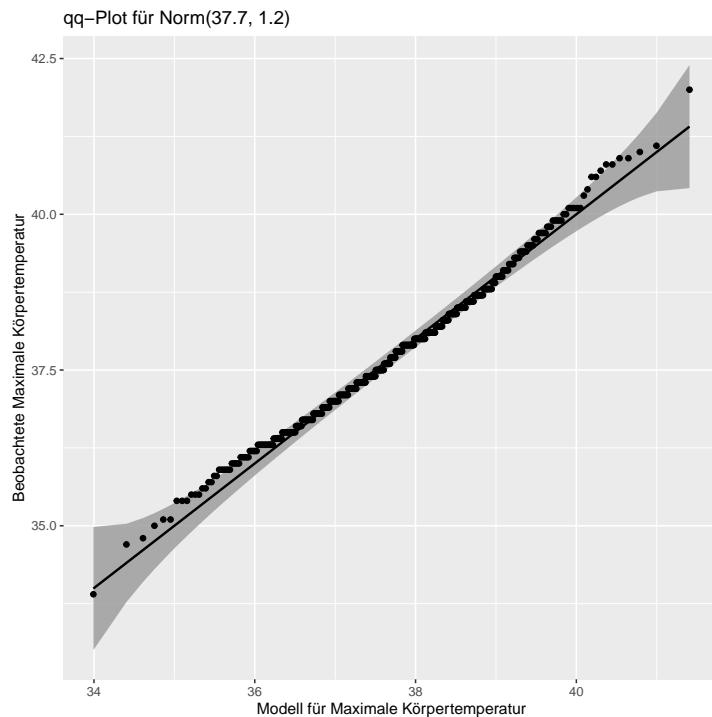


Die Daten scheinen auch bezüglich des qq-Plots in guter Übereinstimmung mit dem geschätzten Modell zu sein, wobei aber auch gewisse kleinere Abweichungen zu erkennen sind. Das Ausmaß der Abweichungen kann analog zum oben erzeugten pp-Plot durch die Angabe von (punktweisen und simultanen) Konfidenzbändern besser eingeschätzt werden. Konfidenzintervalle werden in Abschnitt 5.3 noch genauer erklärt werden. Eine ähnliche Möglichkeit bietet auch wieder das Paket "qqplotr" (Almeida et al. (2018)), welches in Form der Funktionen `stat_qq_point`, `stat_qq_line` und `stat_qq_band` zusätzliche Funktionalität für das Paket "ggplot2" (Wickham (2009)) zur Verfügung stellt. Durch Angabe von `identity = TRUE` erreichen wir außerdem eine Ausrichtung an der Winkelhalbierenden.

```

1 ggplot(ITSData[ -398 , ] , aes(sample = Temperatur)) +
2   qqplotr :: stat_qq_band(dparams = list(mean = 37.7 , sd = 1.2) , identity = TRUE) +
3   qqplotr :: stat_qq_point(dparams = list(mean = 37.7 , sd = 1.2)) +
4   qqplotr :: stat_qq_line(dparams = list(mean = 37.7 , sd = 1.2) , identity = TRUE) +
5   xlab("Modell für Maximale Körpertemperatur") +
6   ylab("Beobachtete Maximale Körpertemperatur") +
7   ggtitle("qq–Plot für Norm(37.7 , 1.2)")

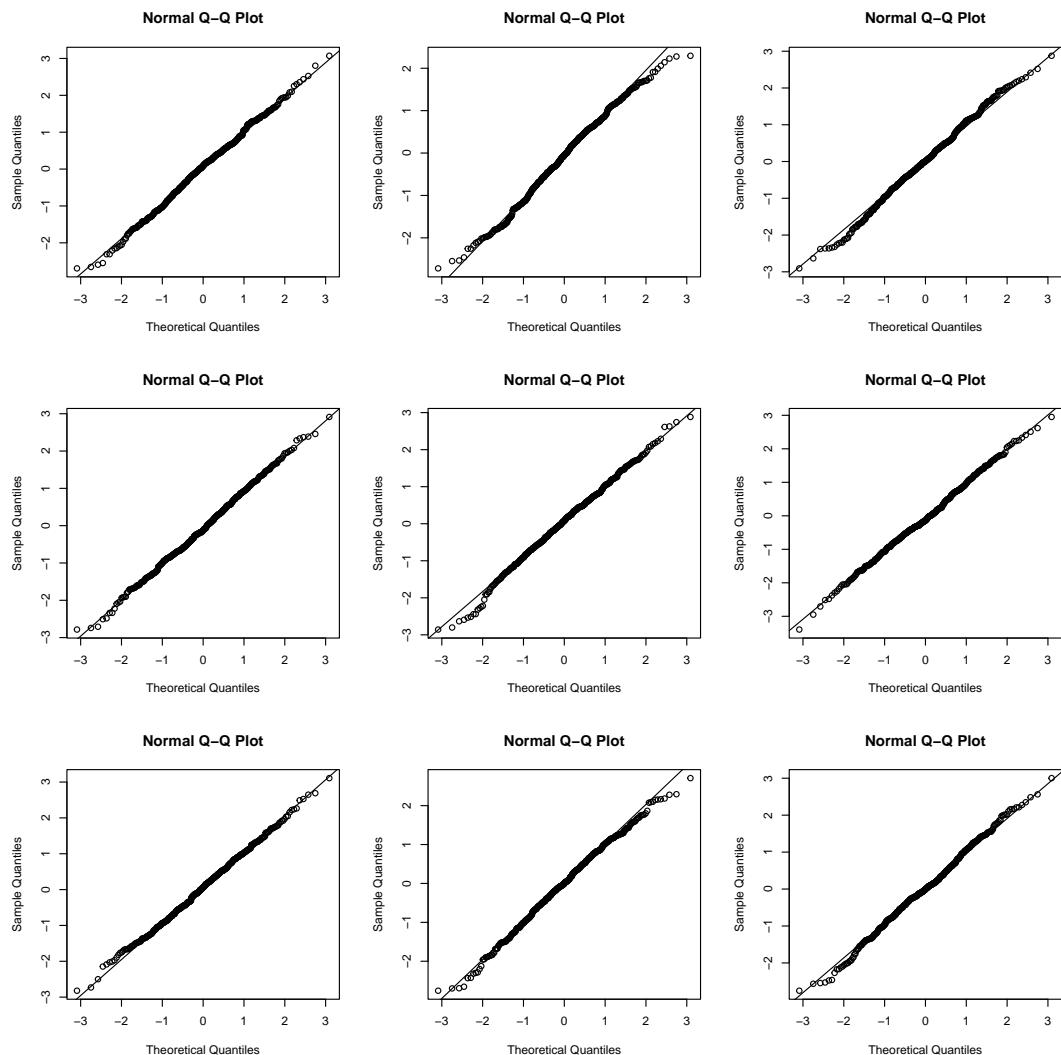
```



Für uns Menschen ist es schwer einzuschätzen, ob sich die Schwankungen, die wir sehen, noch im Rahmen der zu erwartenden stochastischen Abweichungen bewegen. Wir können diese zufälligen Abweichungen vom theoretischen Modell auch mit Hilfe simulierter Daten visualisieren. Dies hilft uns zusätzlich, die beobachteten Abweichungen besser zu beurteilen. Wir vergleichen den obigen qq-Plot mit qq-Plots, von simulierten normalverteilten Daten. Wir erzeugen im Folgenden mit Hilfe der Funktion `rnorm` standardnormalverteilte Daten (die konkreten Parameter der Normalverteilung spielen hierbei keine Rolle) und stellen diese dann mittels `qqnorm` und `qqline` in Form von qq-Plots dar. Damit wir auch einen Eindruck von den Schwankungen von Stichprobe zu Stichprobe bekommen, wiederholen wir das Ganze neun Mal. Hierzu verwenden wir eine sogenannte `for`-Schleife. Damit alle Plots in einem Grafikfenster angezeigt werden, verwenden wir außerdem die Funktion `par`, mit der man eine Vielzahl von Grafikparametern anpassen kann. Mit dem Argument `mfrow` kann ein Grafikfenster in eine gewisse Anzahl von Zeilen und Spalten unterteilt werden. In unserem Fall wählen wir drei Zeilen und drei Spalten, die dann Zeile für Zeile aufgefüllt werden.

```

1 par(mfrow=c(3,3))
2 for(i in 1:9){
3   x <- rnorm(499)
4   qqnorm(x)
5   qqline(x)
6 }
```



Wir sehen demnach auch bei normalverteilten Daten gewisse Abweichungen von der Gerade, die den Abweichungen im Fall der maximalen Körpertemperatur durchaus ähnlich sind. Dies bestätigt einmal mehr unseren ersten Eindruck, dass die maximale Körpertemperatur von ITS-Patienten (ohne stark hypothermische Patienten) recht gut durch eine Normalverteilung beschrieben werden kann.

Anmerkung:

Wie bereits oben erwähnt, werden im Fall realer und simulierter realer Daten nie alle Punkte exakt auf der Gerade zu liegen kommen. Das wäre im Gegenteil ein Hinweis darauf, dass es sich um keine realen Daten handelt bzw. die Daten entsprechend verfälscht wurden. Deutliche Abweichungen von der Gerade weisen hingegen darauf hin, dass das Modell nicht zu den Daten passt (oder umgekehrt). Die Punkte sollten demnach keine deutliche "Bananenform" (evtl. auch einseitig oder verdreht) ausbilden.

Im Fall der Normalverteilung können wir alternativ auch den Median und die geeignet standardisierten MAD und IQR als konsistente Schätzer für den Mittelwert und die Standardabweichung verwenden. Wie wir bereits in Abschnitt 2.5.1 gesehen haben, ist es in diesem Fall außerdem nicht nötig, Patient 398 zu entfernen.

```
1 median(ITSData$Temperatur)
```

```
[1] 37.7
```

```
1 mad(ITSData$Temperatur)
```

```
[1] 1.18608
```

```
1 sIQR(ITSData$Temperatur)
```

```
[1] 1.111952
```

Wir erhalten sehr ähnliche Ergebnisse wie im Fall des ML-Schätzers.

Anmerkung:

Median, MAD und IQR liefern unter sehr allgemeinen Voraussetzungen konsistente Schätzungen für den theoretischen Median, MAD und IQR. Es ist insbesondere nicht nötig, ein spezielles parametrisches Modell anzunehmen. Daher spricht man auch von nicht-parametrischen Schätzern. Diese generelle Eigenschaft und die zusätzliche Robustheit machen diese Schätzer für viele Anwendungen sehr interessant.

Ein weiteres Schätzprinzip, welches gerade bei einfachen Wahrscheinlichkeitsmodellen sehr gut funktioniert, ist die Minimum-Distanz-Schätzung.

Definition 5.7 (Minimum-Distanz-Schätzer). *Gegeben sei ein Wahrscheinlichkeitsmodell $\mathcal{P} = \{P_\theta \mid \theta \in \Theta\}$, $\Theta \subset \mathbb{R}^k$ ($k \in \mathbb{N}$). Weiter seien x_1, \dots, x_n die Realisationen von unabhängig und P_θ verteilten Zufallsvariablen X_1, \dots, X_n und \hat{F}_n deren empirische Verteilung. Dann betrachten wir*

$$D(\theta) = \text{dist}(P_\theta, \hat{F}_n) \quad (5.9)$$

wobei *dist* für einen Abstand zwischen Verteilungen steht. Der **Minimum-Distanz-Schätzer** (kurz: MD-Schätzer) für θ ist dann die Minimalstelle von $D(\theta)$.

Wir geben einige zusätzliche Erläuterungen.

Bemerkung 5.8. (a) MD-Schätzer sind in der Regel konsistente Schätzer.

(b) Wir werden im Folgenden den Cramér-von-Mises MD-Schätzer (kurz: CvM-MD-Schätzer) und den Kolmogorov(-Smirnov) MD-Schätzer (kurz: KS-MD-Schätzer) bestimmen. Die zugehörigen Distanzen werden auf Basis der Verteilungsfunktionen berechnet. Die **Cramér-von-Mises Distanz** lautet

$$\text{dist}_{CvM}(P_\theta, \hat{F}_n) = \int |P_\theta(x) - \hat{F}_n(x)|^2 Q(dx) \quad (5.10)$$

wobei die Verteilung Q , nach der integriert wird, meist als P_θ oder \hat{F}_n gewählt wird. Im Fall von \hat{F}_n wird das Integral zu einer Summe. Die **Kolmogorov(-Smirnov) Distanz** ist gegeben durch

$$\text{dist}_{KS}(P_\theta, \hat{F}_n) = \max_{x \in \mathbb{R}} |P_\theta(x) - \hat{F}_n(x)| \quad (5.11)$$

Beide MD-Schätzer sind sehr robust gegen Ausreißer und gewisse Modellabweichungen.

Wir können die Funktion `MDEstimator` aus dem Paket "distrMod" (Kohl and Ruckdeschel (2010)) für die Berechnung von MD-Schätzern verwenden. Im Fall des CvM-MD-Schätzer und des KS-MD-Schätzer stellt das Paket zusätzlich die Funktionen `CvMMDEstimator` und `KolmogorovMDEstimator` zur Verfügung. Wir betrachten wieder die maximale Körpertemperatur der ITS-Patienten und berechnen zunächst den CvM-MD-Schätzer. Wie im Fall des ML-Schätzers gehen wir dabei in zwei Schritten vor und definieren zunächst das Modell und berechnen anschließend den Schätzer. Wir erhalten ohne Patient 398

```
1 Modell ← NormLocationScaleFamily()
2 CvMMDEstimator(ITSData$Temperatur[-398], Modell)
```

```
Evaluations of Minimum CvM distance estimate ( mu = model distr. ) :
-----
mean           sd
37.67520679   1.13610915
( 0.06932949) ( 0.04461328)
```

Das Ergebnis ist demnach recht ähnlich zum ML-Schätzer. Wir wiederholen die Schätzung und verwenden dieses Mal den KS-MD-Schätzer.

```
1 KolmogorovMDEstimator(ITSData$Temperatur[-398], Modell)
```

```
Evaluations of Minimum Kolmogorov distance estimate :
-----
estimate:
mean           sd
37.679362    1.141183
```

Auch hier erhalten wir ein sehr ähnliches Ergebnis. Wir wiederholen die Schätzung, dieses Mal mit Patient 398, um die Robustheit der MD-Schätzer im Unterschied zum ML-Schätzer zu demonstrieren.

```
1 ## ML-Schätzer
2 MLEstimator(ITSData$Temperatur, Modell)
```

```
Evaluations of Maximum likelihood estimate:
-----
mean           sd
37.66320000   1.73373751
( 0.07753510) ( 0.05482559)
```

```
1 ## CvM-MD-Schätzer
2 CvMMDEstimator(ITSData$Temperatur, Modell)
```

```
Evaluations of Minimum CvM distance estimate ( mu = model distr. ) :
-----
mean           sd
37.67176542   1.13941964
( 0.06946194) ( 0.04469851)
```

```
1 ## KS-MD-Schätzer
2 KolmogorovMDEstimator(ITSData$Temperatur, Modell)
```

```
Evaluations of Minimum Kolmogorov distance estimate :
-----
estimate:
  mean        sd
37.676420  1.143365
```

Wir sehen, dass im Fall der MD-Schätzer die Ergebnisse nahezu unverändert bleiben, während beim ML-Schätzer sich insbesondere die Schätzung für die Standardabweichung deutlich verändert.

Als letzte Klasse von Schätzern betrachten wir sogenannte **optimal-robuste asymptotisch lineare Schätzer**; kurz: **AL-Schätzer**. In diesem Fall betrachtet man zusätzlich zum parametrischen Modell \mathcal{P} noch eine sogenannte schrumpfende Umgebung um die angenommene Modellverteilung herum. In den allermeisten Fällen ist dies die sogenannte **Kontaminationsumgebung**, welche auch als das gross-error Modell von Tukey bekannt ist

$$Q_n = (1 - r_n)P_\theta + r_n H_n \quad r_n = \frac{r}{\sqrt{n}} \quad (5.12)$$

Hierbei ist $r \in [0, \infty)$ der **Umgebungsradius** und H_n eine beliebige Wahrscheinlichkeitsverteilung. Q_n ist demnach die Wahrscheinlichkeitsverteilung, welche die beobachteten Daten tatsächlich erzeugt hat. Der Großteil $((1 - \varepsilon)\%, \varepsilon \in [0, 1])$ der Daten stammt von der angenommenen Modellverteilung, aber ein kleiner Teil der Daten ($\varepsilon\%$) ist fehlerhaft und wurde durch eine unbekannte Fehlerverteilung H_n erzeugt. Für eine gegebene Stichprobengröße n erhält man r dann durch $r = \sqrt{n}\varepsilon$.

Bemerkung 5.9. (a) Das Schrumpfen der Umgebung mit der Stichprobengröße n sorgt dafür, dass die Abweichungen so klein bleiben, dass diese nicht bzw. nur sehr schwer mit Hilfe statistischer Verfahren (z.B. Anpassungs- oder Ausreißertests) zuverlässig erkennbar sind.

(b) Die Umgebung sorgt nicht nur dafür, dass die AL-Schätzer robust gegenüber einem gewissen Anteil von Ausreißern sind, sondern dass diese auch bei anderen (subtileren) Modellabweichungen immer noch verlässliche Ergebnisse liefern.

AL-Schätzer minimieren den **maximalen asymptotischen MSE**, der sich in diesem Fall schreiben lässt als

$$\text{asMSE}_\theta(\psi_\theta, r) = E_\theta |\psi_\theta|^2 + r^2 \left(\sup_{P_\theta} |\psi_\theta| \right)^2$$

Hierbei werden die AL-Schätzer S_n durch ihre sogenannte **Influenzfunktion** ψ_θ beschrieben. Die Influenzfunktion gibt an, welchen Einfluss eine einzelne Beobachtung auf das Schätzergebnis hat. Sie entspricht in einem gewissen Sinne der Ableitung der Schätzfunktion. Indem man die Influenzkurve bestimmt, welche diesen maximalen asymptotischen MSE minimiert, findet man auch den zugehörigen AL-Schätzer. Man berechnet den Schätzer dann üblicherweise durch eine sogenannte 1-Schritt Konstruktion

$$S_n^{(1)} = S^{\text{start}} + \frac{1}{n} \sum_{i=1}^n \psi_\theta(x_i - S^{\text{start}}) \quad (5.13)$$

für die man einen geeigneten Startwert/Startschätzer S^{start} benötigt. Diese Konstruktion kann man auch k -fach ($k \in \mathbb{N}$) wiederholen oder wiederholen bis sich der Schätzwert nicht mehr wesentlich ändert, wobei im zweiten Schritt dann $S_n^{(1)}$ die Rolle des Startschätzer übernimmt, usw. Unserer Erfahrung nach ist es völlig ausreichend, $k = 3$ zu wählen.

Anmerkung:

Die Startschätzer sollten idealerweise sehr robuste Schätzverfahren sein; d.h., Schätzer, die auch bei einem hohen Anteil von Ausreißern noch verlässliche Ergebnisse liefern. Geeignet sind zum Beispiel Median, MAD, IQR oder auch CvM-MD-Schätzer und KS-MD-Schätzer.

Das Problem, dass in der Praxis der genaue Anteil der fehlerhaften Daten nicht bekannt ist, kann durch den Einsatz der sogenannten **Radius Minimax Schätzer**, kurz: **RMX-Schätzer**, gelöst werden. Es handelt sich hierbei um AL-Schätzer, die nicht für einen bestimmten Anteil ϵ von Abweichungen, sondern für ein ganzes Intervall $\epsilon \in [\epsilon_1, \epsilon_2]$ optimal sind. Für weitere Einzelheiten zu AL- und RMX-Schätzer verweisen wir auf Rieder (1994), Kohl (2005) und Kohl et al. (2010). AL- und RMX-Schätzer für die Normalverteilung sind im Paket "RobLox" (Kohl (2019)) implementiert, wobei Median und MAD als Startschätzer verwendet werden. Wir betrachten wieder unsere Temperatur-Daten und gehen auf Basis des qq-Plots davon aus, dass zwischen einer (Patient 398) und 5% der Beobachtungen fehlerhaft sind. Wir berechnen den entsprechenden RMX-Schätzer mit Hilfe der 3-Schritt Konstruktion.

```
1 roblox(ITSDaten$Temperatur, 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)
```

Alternativ können wir auch die Funktion `rmx` aus dem Paket "rmx" (Kohl (2022c)) verwenden, wobei hier $k = 3$ bereits als Defaulteinstellung verwendet wird und daher nicht angegeben werden muss. Das Modell wird ähnlich wie im Fall der Funktion `fitdistr` aus dem Paket "MASS" (Venables and Ripley (2002)) durch das Kürzel für die jeweilige Verteilung spezifiziert.

```
1 res.rmx ← rmx(ITSDaten$Temperatur, model = "norm",
2                  eps.lower = 1/500, eps.upper = 0.05)
3 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 = ITSDaten$Temperatur, model = "norm", eps.lower = 1/500,
     eps.upper = 0.05)
```

Das Ergebnis ist sehr ähnlich zu den obigen Ergebnissen der anderen robusten Schätzverfahren. Eine ausführlichere Zusammenfassung der Ergebnisse lässt sich mittels der Funktion `summary` erzeugen.

```
1 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 = ITSDaten$Temperatur, model = "norm", eps.lower = 1/500,
     eps.upper = 0.05)
```

Das Paket "rmx" (Kohl (2022c)) enthält auch eine Reihe von diagnostischen Funktionen. So kann etwa mit Hilfe der Funktionen `outlier` und `getOutliers` ausgegeben werden, wo die Ausreißerbereiche liegen und welche Datenpunkte entsprechend als Ausreißer anzusehen sind. Dies sind standardmäßig Datenpunkte, die kleiner als das 0.01% oder größer als das 99.9% Quantil des geschätzten Modells sind.

```
1 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 %

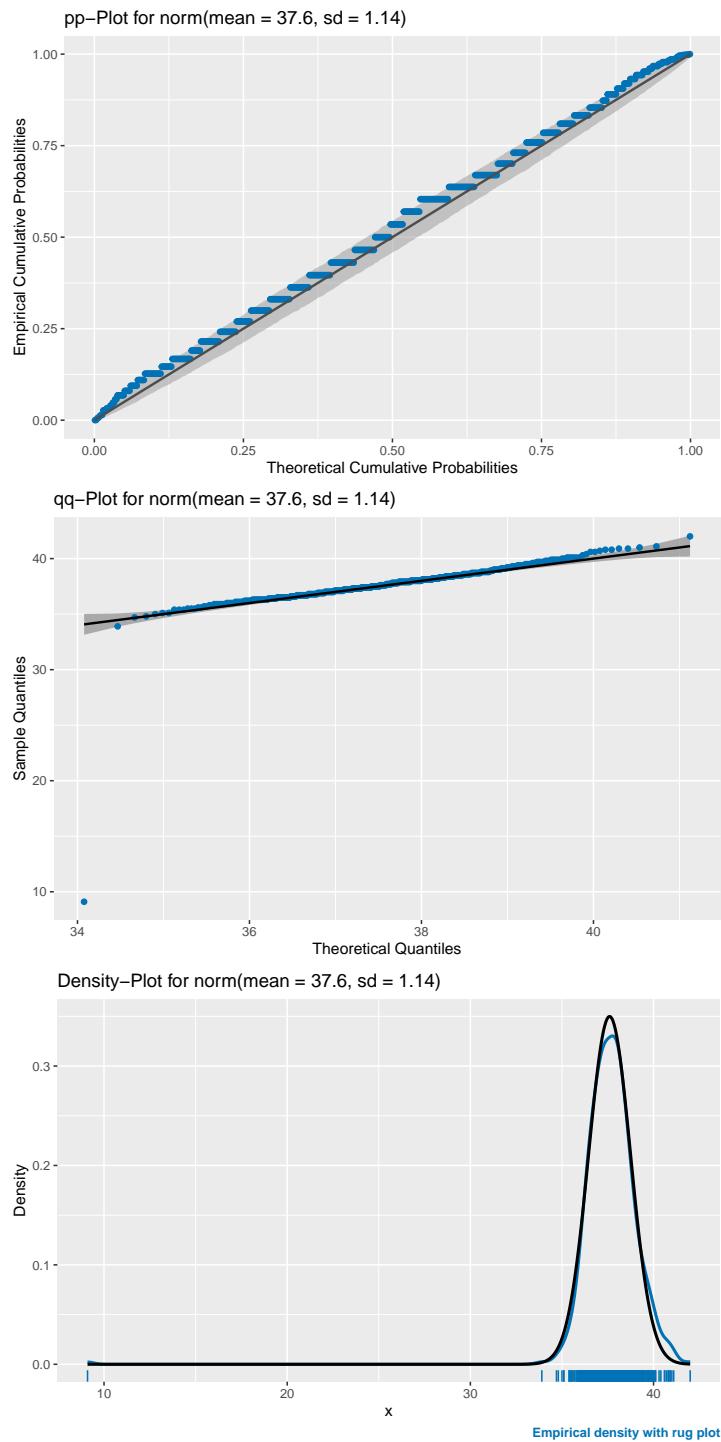
      Gross-errors for RMX = 0.2 - 5 %
```

```
1 getOutliers(res.rmx)
```

```
$values  
[1] 9.1 42.0  
  
$indices  
[1] 398 397
```

Wir erhalten demnach zwei Ausreißer. Neben dem bereits mehrfach auffälligen Patienten 398 mit einer maximalen Körpertemperatur von 9.1°C liegt auch Patient 397 mit einer maximalen Körpertemperatur von 42.0°C im Ausreißerbereich. Für die Erstellung von pp-, qq- und Dichteplot zur Modellvalidierung sind die Funktionen `ppPlot`, `qqPlot` und `dPlot` vorgesehen. Wir fügen die drei Plots mit Hilfe der Funktion `grid.arrange` aus dem Paket "gridExtra" (Auguie (2017)) zu einer Abbildung zusammen.

```
1 gg1 <- ppPlot(res.rmx)  
2 gg2 <- qqPlot(res.rmx)  
3 gg3 <- dPlot(res.rmx)  
4 grid.arrange(gg1, gg2, gg3, ncol = 1)
```



Wir sehen eine recht gute Übereinstimmung zwischen Daten und Modell, wenn man Patient 398 außer Acht lässt.

Das Paket "RobLox" (Kohl (2019)) enthält nur eine Implementation der RMX-Schätzer für die Normalverteilung, für andere Modelle kann die Funktion `roptest` aus dem Paket "ROptEst" (Kohl and Ruckdeschel (2019)) verwendet werden. Das Paket "rmx" (Kohl (2022c)) enthält aktuell eine Implementation für die Normal- und die Binomialverteilung. Es werden zukünftig aber weitere Modelle hin-

zukommen. Das Paket "RobExtremes" (Ruckdeschel et al. (2019)) enthält optimierten Code für die Berechnung von RMX-Schätzern für Extremwertverteilungen wie etwa die Weibull-Verteilung. Die Pakete "ROptEst" und "RobExtremes" enthalten einen Ordner mit R-Skripten, in denen die Anwendung der wichtigsten Funktionen demonstriert ist; siehe <https://github.com/cran/ROptEst/tree/master/inst/scripts> und <https://github.com/cran/RobExtremes/tree/master/inst/scripts>. Diese Ordner befinden sich nach der Installation der Pakete auch auf Ihrem Computer und zwar handelt es sich um Unterordner im Verzeichniss des Paketes. Das Verzeichnis, in dem ein Paket installiert ist, können Sie durch die Funktion `path.package` ermitteln; z.B.,

```
1 path.package("ROptEst")
```

Das Paket "rmx" enthält eine sogenannte Vignette, in der die wichtigsten Funktionen des Paketes vorgestellt werden. Diese kann mit Hilfe der Funktion `vignette` geöffnet werden.

```
1 vignette("rmx")
```

Anmerkung:

Es gibt noch eine Reihe anderer wichtiger Schätzerklassen wie verallgemeinerte ML-Schätzer (kurz: M-Schätzer) oder Rang-Schätzer (kurz: R-Schätzer). Ein weiteres sehr wichtiges Konstruktionsprinzip gerade bei komplexeren Modellen ist die Kleinsten-Quadrat-Schätzung (kurz: KQ-Schätzung), welche auf Gauß und Legendre zurückgeht. Dabei wird das Modell geschätzt, indem die Summe der quadratischen Abweichungen der Beobachtungen vom Modell minimiert wird.

Ein ebenfalls häufiges Schätzproblem besteht darin, **optimale Schwellenwerte** (Cut-offs) zu bestimmen. Ein Beispiel hierzu haben wir bereits kennengelernt, nämlich den Normbereich. Der **Normbereich** oder **Referenzbereich** ist ein Intervall, welches üblicherweise die Werte von 95% der untersuchten gesunden Personen enthält. Meist handelt es sich um eine zweiseitiges Intervall. In diesem Fall haben jeweils 2.5% der gesunden Personen Werte die unter- oder oberhalb dieses Intervalls liegen. Wie wir bereits gesehen haben lässt sich dieses zweiseitige Intervall im Fall der Normalverteilung (und auch einiger anderer Verteilungen) gut durch das 2σ -Intervall approximieren; siehe Bemerkung 4.16 (b). Üblicherweise wird dieses Intervall durch die entsprechenden Quantile (also 2.5% und 97.5% Quantil) bestimmt. Hierbei kann entweder ein Wahrscheinlichkeitsmodell zugrundegelegt werden, wie im Beispiel 4.20 (a), oder es werden die empirischen Quantile der erhobenen Daten herangezogen. Oftmals werden auch einseitige Intervalle benötigt. Beispiele hierfür sind etwa die Nachweisgrenzen "**limit of blank**" (**LOB**), "**limit of detection**" (**LOD**) und "**(lower/upper) limit of quantitation/quantification**" (**LOQ, LLOQ, ULOQ**) (Armbruster and Pry (2008)) für quantitative Assays. Die Nachweisgrenze LOB entspricht einem einseitigen Intervall, dessen Obergrenze das entsprechende 95% Quantil der gemessenen Leerproben ist. Legt man eine Normalverteilung zugrunde, so ergibt sich

$$\text{LOB} = \text{AM}_{\text{Leerproben}} + 1.645 \times \text{SD}_{\text{Leerproben}} \quad (5.14)$$

wobei die Konstante 1.645 dem 95% Quantil der Standardnormalverteilung entspricht.

```
1 qnorm(0.95)
```

```
[1] 1.644854
```

Erhält man Werte unterhalb dieser Grenze, so muss man davon ausgehen, dass es sich um eine Leerprobe handelt. Die Nachweisegrenze LOD erhält man als

$$\text{LOD} = \text{AM}_{\text{Leerproben}} + 3.3 \times \text{SD}_{\text{Leerproben}} \quad (5.15)$$

wobei

```
1 pnorm(3.3)
```

```
[1] 0.9995166
```

Es handelt sich also um das 99.95% Quantil, welches in diesem Fall verwendet wird. Alternativ verwendet man anstelle der Leerproben auch Proben mit einer niedrigen Konzentration und berechnet LOD als

$$\text{LOD} = \text{LOB} + 1.645 \times \text{SD}_{\text{Proben nied. Konz.}} \quad (5.16)$$

Ergeben Sie bei einer Messung Werte oberhalb des LOD, so kann man von einem positiven Signal ausgehen. Der untersuchte Analyt ist demnach in der Probe enthalten. Eine sichere Quantifizierung der Menge des untersuchten Analyten erfolgt aber genau genommen erst für Signalwerte oberhalb der (unteren) Quantifikationsgrenze LOQ bzw. genauer LLOQ. Diese berechnet sich zum Beispiel als

$$(\text{L})\text{LOQ} = \text{AM}_{\text{Leerproben}} + 10 \times \text{SD}_{\text{Leerproben}} \quad (5.17)$$

Anstelle des Faktors 10 kommen hier auch manchmal 5 oder 6 zum Einsatz. Darüber hinaus gibt es auch andere Ansätze, diese Nachweisgrenzen zu ermitteln; siehe zum Beispiel Bernal (2014).

Ein anderer Ansatz Cut-offs zu bestimmen, beruht darauf, die Suche nach dem optimalen Cut-off als ein Klassifikationsproblem für zwei Gruppen aufzufassen. Man sucht demnach nach einer Grenze, so dass man mit möglichst hoher Wahrscheinlichkeit beide Gruppen richtig klassifiziert. Wir müssen hierfür außerdem einen monotonen Zusammenhang zwischen der Messung und den Gruppen voraussetzen; d.h., niedrige Werte sprechen für das Vorliegen der ersten Gruppe ("negativ") und hohe Wert für das Vorliegen der zweiten Gruppe ("positiv"). Dies führt uns auf die Begriffe **Sensitivität** und **Spezifität**. Unter der Sensitivität versteht man die Wahrscheinlichkeit, dass ein Mitglied der positiven Gruppe als "positiv" eingestuft wird und als Spezifität entsprechend die Wahrscheinlichkeit, dass ein Mitglied der negativen Gruppe als "negativ" eingruppiert wird. Wollen wir beide Wahrscheinlichkeiten maximieren, so ist es naheliegend die Summe der beiden Kriterien zu betrachten. Dies führt uns etwa auf **Youdens J Statistik** (Youden (1950))

$$\text{Youdens J} = \text{Sensitivität} + \text{Spezifität} - 1 \quad (5.18)$$

welche vor allem in der Informatik auch als **Informiertheit** bezeichnet wird (Powers (2011)). Ebenfalls äquivalent zur Summe ist das arithmetische Mittel der beiden Kriterien, welches auch als **balancierte Genauigkeit (bACC)** (Brodersen et al. (2010)) bekannt ist

$$\text{bACC} = 0.5 \times \text{Sensitivität} + 0.5 \times \text{Spezifität} \quad (5.19)$$

Man könnte auch sagen, dass in diesem Fall beide Kriterien gleich gewichtet werden. Sind Sensitivität und Spezifität nicht gleich wichtig, so könnte man dies durch eine entsprechende **gewichtete Genauigkeit (wACC)** ausdrücken (Brodersen et al. (2010))

$$\text{wACC} = w \times \text{Sensitivität} + (1 - w) \times \text{Spezifität} \quad (5.20)$$

mit $w \in [0, 1]$. Entsprechende Berechnungen können zum Beispiel mit Hilfe der Funktion `optCutOff` aus dem Paket "MKclass" (Kohl (2020a)) durchgeführt werden, wobei neben den oben genannten Genauigkeitsmaßen alle Maße verwendet werden können, die mit der Funktion `perfMeasures` berechnet werden können.

Wir betrachten den ITS-Datensatz und wollen einen Cut-off für Bilirubin ermitteln, mit dem möglichst gut zwischen Patienten mit und ohne Leberversagen unterschieden werden kann. Für die Berechnungen runden wir die Bilirubinwerte auf eine Nachkommastelle, was realistischer Weise der Messgenauigkeit des entsprechenden Assays entsprechen könnte.

```
1 Bili <- round(ITSDaten$Bilirubin, 1)
```

Wir vermuten einen einseitigen Referenzbereich; d.h., Patienten ohne Leberversagen (Gruppe "negativ") weisen niedrige und Patienten mit Leberversagen (Gruppe "positiv") hohe Bilirubin-Werte auf.

Wir gehen zunächst wie bei der Berechnung von Normbereichen vor und bestimmen das empirische 95% Quantil für die Patienten ohne Leberversagen und berechnen anschließend mit Hilfe der Funktion `perfMeasures` aus dem Paket "MKclass" (Kohl (2020a)) die Sensitivität und Spezifität dieser empirischen Intervallobergrenze für die Vorhersage eines Leberversagens.

```
1 quantile(Bili[ITSDaten$Leberversagen == 0], probs = 0.95)
```

```
95%
49.615
```

```
1 perfMeasures(pred = Bili, truth = ITSDaten$Leberversagen,
2                 namePos = 1, cutoff = 49.615, measures = c("SENS", "SPEC"))
```

Performance Measure(s)	
Measure	Value
sensitivity (SENS)	0.75
specificity (SPEC)	0.95

Natürlich ist in diesem Fall die Spezifität gleich 95%, da wir das empirische 95% Quantil der Patienten ohne Leberversagen als Grenze herangezogen haben. Dieser Ansatz gibt uns also in einem gewissen Sinne Kontrolle über die Spezifität, wobei die Sensitivität unbekannt ist und in einem zweiten Schritt berechnet werden muss.

Alternativ können wir auch annehmen, dass die Werte der Patienten ohne Leberversagen einer log-Normalverteilung folgen und analog zum Beispiel 4.20 (a) vorgehen. Wir bilden entsprechend die log-Werte und berechnen davon Mittelwert und Standardabweichung. Hierfür bestimmen wir dann das 95% Quantil der zugehörigen log-Normalverteilung und berechnen Sensitivität und Spezifität.

```
1 mean(log(Bili[ITSDaten$Leberversagen == 0]))  
[1] 2.77395  
  
1 sd(log(Bili[ITSDaten$Leberversagen == 0]))  
[1] 0.6038337  
  
1 qlnorm(0.95, meanlog = 2.774, sdlog = 0.604)  
[1] 43.27139  
  
1 perfMeasures(pred = Bili, truth = ITSDaten$Leberversagen,  
2 namePos = 1, cutoff = 43.27, measures = c("SENS", "SPEC"))
```

Performance Measure(s)

Measure	Value
1 sensitivity (SENS)	0.7500000
2 specificity (SPEC)	0.9354167

Das Cut-off sowie Sensitivität und Spezifität sind ähnlich zur empirischen Berechnung. Im nächsten Schritt ermitteln wir den optimalen Cut-off, der die Youden J Statistik und damit auch die balancierte Genauigkeit maximiert. Wir verwenden die Funktionen `optCutoff` und `perfMeasures` aus dem Paket "MKclass" (Kohl (2020a)).

```
1 ## Youden J Statistik  
2 optCutoff(pred = Bili, truth = ITSDaten$Leberversagen, namePos = 1)  
  
Optimal Cut-off YJS  
28.4000000 0.7395833  
  
1 ## bACC  
2 optCutoff(pred = Bili, truth = ITSDaten$Leberversagen,  
3 namePos = 1, perfMeasure = "BACC")  
  
Optimal Cut-off BACC  
28.4000000 0.8697917  
  
1 perfMeasures(pred = Bili, truth = ITSDaten$Leberversagen,  
2 namePos = 1, cutoff = 28.4, measures = c("SENS", "SPEC"))
```

Performance Measure(s)	
Measure	Value
1 sensitivity (SENS)	0.9000000
2 specificity (SPEC)	0.8395833

Wir erhalten eine etwas höhere Sensitivität als im Fall der berechneten Referenzbereiche, aber auf Kosten einer etwas geringeren Spezifität.

Anmerkung:

Die Bestimmung des Cut-offs und die Anwendung des Cut-offs sollte idealerweise nicht auf dem selben Datensatz erfolgen. Da ansonsten die Gefahr eines Bias (“resubstitution bias”) bei der Schätzung von Sensitivität und Spezifität besteht. Ist dies nicht vermeidbar, empfiehlt sich der Einsatz eines Resamplingverfahrens, wie Bootstrap oder Kreuzvalidierung, um diesen Bias möglichst gering zu halten.

Wir berechnen den optimalen Cut-off mittels 1000 Bootstrapwiederholungen. Aufgrund des sehr geringen Anteils an Patienten mit Leberversagen verwenden wir eine stratifizierte Version des Bootstraps. Wir verwenden wieder die Funktionen `optCutoff` und `perfMeasures` aus dem Paket "MKclass" (Kohl (2020a)).

```

1 B <- 1000
2 n0 <- sum(ITSDaten$Leberversagen == 0)
3 ind0 <- which(ITSDaten$Leberversagen == 0)
4 n1 <- sum(ITSDaten$Leberversagen == 1)
5 ind1 <- which(ITSDaten$Leberversagen == 1)
6 cutoffs <- numeric(B)
7 for(i in 1:B){
8   auswahl0 <- sample(ind0, n0, replace = TRUE)
9   auswahl1 <- sample(ind1, n1, replace = TRUE)
10  BS.data <- ITSDaten[c(auswahl0, auswahl1),]
11  cutoffs[i] <- optCutoff(pred = round(BS.data$Bilirubin, 1),
12                            truth = BS.data$Leberversagen,
13                            namePos = 1)[1]
14}
15 summary(cutoffs)

```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
17.70	28.40	31.60	38.27	53.30	104.70

Wir erhalten einen mittleren optimalen Cut-off, der etwas höher liegt als im Fall der Berechnung auf dem vollen Datensatz.

```

1 perfMeasures(pred = Bili, truth = ITSDaten$Leberversagen,
2                 namePos = 1, cutoff = mean(cutoffs),
3                 measures = c("SENS", "SPEC"))

```

Performance Measure(s)	
Measure	Value
1 sensitivity (SENS)	0.7500000
2 specificity (SPEC)	0.9145833

Je nachdem, ob das Augenmerk mehr auf der Sensitivität oder mehr auf der Spezifität liegt, kann man das entsprechende Genauigkeitsmaß zum Beispiel durch die Verwendung von Gewichten mehr in den Fokus der Analyse rücken. Wir wählen im Folgenden eine Gewichtung von 3 zu 1 und verzichten bei dieser Demonstration auf die Absicherung mittels Bootstrap.

```
1 ## Mehr Gewicht auf Sensitivität
2 optCutoff(pred = Bili, truth = ITSDaten$Leberversagen,
3            namePos = 1, perfMeasure = "WACC", wACC = 0.75)
```

Optimal Cut-off	WACC
17.7000000	0.9005208

```
1 perfMeasures(pred = Bili, truth = ITSDaten$Leberversagen,
2                namePos = 1, cutoff = 17.7, measures = c("SENS", "SPEC"))
```

Performance Measure(s)	
Measure	Value
1 sensitivity (SENS)	1.0000000
2 specificity (SPEC)	0.6020833

```
1 ## Mehr Gewicht auf Spezifität
2 optCutoff(pred = Bili, truth = ITSDaten$Leberversagen,
3            namePos = 1, perfMeasure = "WACC", wACC = 0.25)
```

Optimal Cut-off	WACC
53.3000000	0.9078125

```
1 perfMeasures(pred = Bili, truth = ITSDaten$Leberversagen,
2                namePos = 1, cutoff = 53.3, measures = c("SENS", "SPEC"))
```

Performance Measure(s)	
Measure	Value
1 sensitivity (SENS)	0.7500000
2 specificity (SPEC)	0.9604167

Wir erhalten entsprechend veränderte optimale Cut-offs, die zu einer höheren geschätzten Sensitivität bzw. zu einer höheren geschätzten Spezifität führen. Ein Anstieg des einen Kriteriums führt gezwungener Maßen zu einer Reduktion des anderen Kritierums, da das Zielkriterium das Maximum der gewichteten Summe von Sensitivität und Spezifität war. Die zugehörige Klassifikationsfunktion bezeichnet man auch als einen Eintscheidungsbaumstumpf (“decision stump”) (Iba and Langley (1992)). Diese kann etwa mit Hilfe der Funktion `decisionStump` des Paketes "MKclass" (Kohl (2020a)) berechnet werden.

```
1 stump ← decisionStump(pred = Bili, truth = ITSDaten$Leberversagen,
2                           namePos = 1)
3 stump
```

```
Call:
decisionStump(pred = Bili, truth = ITSDaten$Leberversagen, namePos = 1)

Cut-off:
Optimal Cut-off
        28.4

Performance:
YJS
0.7396
```

Wir können damit dann Vorhersagen für neue Daten berechnen.

```
1 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
```

Anmerkung:

Es gibt noch viele andere Genauigkeitsmaße, die man für die obigen Berechnungen heranziehen könnte. Im Paket "MKclass" (Kohl (2020a)) sind aktuell 80 solcher Maße implementiert. Alle diese Genauigkeitsmaße erfordern eine Dichotomisierung, was durchaus auch kritisch gesehen werden kann (Harrell (2014, Kapitel 10)) und man sollte diese Cut-offs besser nur zu einer groben Orientierung heranziehen. In den meisten Fällen empfiehlt es sich stattdessen, eine stetige Klassifikationsfunktionen zu betrachten. Solche Funktionen kann man mittels Genauigkeitsscores wie etwa der AUC (“Area Under the receiver operating characteristic Curve”) (Hanley and McNeil (1982)) oder dem Brier Score (Brier (1950)) miteinander vergleichen.

Wir fassen Bilirubin als eine stetige Klassifikationsfunktion zur Vorhersage von Leberversagen auf und berechnen die AUC und den Brier Score. Im Fall des Brier Scores müssen die Daten zusätzlich auf das Intervall $[0, 1]$ transformiert werden. Wir verwenden für die Berechnungen die Funktion `perfScores` aus dem Paket "MKclass" (Kohl (2020a)).

```
1 perfScores(pred = Bili, truth = ITSDaten$Leberversagen, namePos = 1,
2             scores = "AUC")
```

Performance Score(s)

Score	Value
1 area under curve (AUC)	0.9372917

```
1 perfScores(pred = Bili, truth = ITSDaten$Leberversagen, namePos = 1,
2             scores = c("AUC", "BS"), transform = TRUE)
```

Performance Score(s)

Score	Value
1 area under curve (AUC)	0.93729167
2 Brier score (BS)	0.02300764

Eine AUC von mehr als 0.5 bzw. ein Brier Score von weniger als 0.25 zeigt an, dass Bilirubin Information zur Beurteilung eines Leberversagens enthält. Eine AUC von 1.0 bzw. ein Brier Score von 0.0 steht für eine fehlerfreie Klassifikation. Die obigen Berechnungen bestätigen, dass Bilirubin bei Intensivpatienten zur Unterstützung der Diagnose eines Leberversagens herangezogen werden kann, wobei die Wahrscheinlichkeit eines Leberversagens mit wachsenden Bilirubinwerten zunimmt.

5.3 Konfidenzintervalle

Im letzten Abschnitt haben wir einige Schätzverfahren kennengelernt und wissen, dass wir möglichst biasfreie (oder zumindest konsistente) und effiziente Schätzer verwenden sollen. Dies sind jedoch lediglich theoretische Eigenschaften, die uns in der Praxis nicht sagen können, wie nah unser Punktschätzer tatsächlich beim gesuchten unbekannten Parameter liegt. Eine Möglichkeit, den Punktschätzer weiter abzusichern, ist das sogenannte Konfidenz- oder Vertrauensintervall.

Definition 5.10 (Konfidenzintervall). *Gegeben sei ein Wahrscheinlichkeitsmodell $\mathcal{P} = \{P_\theta \mid \theta \in \Theta\}$, $\Theta \subset \mathbb{R}^k$ ($k \in \mathbb{N}$). Weiter seien x_1, \dots, x_n die Realisationen von unabhängig und P_θ verteilten Zufallsvariablen X_1, \dots, X_n . Dann heißt der **Intervallschätzer***

$$\hat{I}(x_1, \dots, x_n) = [S_u(x_1, \dots, x_n), S_o(x_1, \dots, x_n)] \quad (5.21)$$

ein $(1 - \alpha)$ -Konfidenzintervall, falls

$$P(\theta \in \hat{I}) \geq 1 - \alpha$$

für $\alpha \in (0, 1)$. Dabei sind S_u und S_o Schätzer für die untere und obere Grenze des Intervalls.

Wir geben einige zusätzliche Erläuterungen (vgl. auch Kohl and Münch (2022c)).

Bemerkung 5.11. (a) Die Definition lässt auch einseitige Konfidenzintervalle zu. In diesem Fall bleibt eine Grenze frei und es wird nur S_u oder S_o benötigt.

(b) Man sagt: Ein Konfidenzintervall überdeckt den wahren unbekannten Parameter mit einer Wahrscheinlichkeit von $1 - \alpha$. Dies drückt aus, dass in 95% der Fälle, in denen wir auf der Basis von Daten Konfidenzintervalle berechnen, diese Intervalle den wahren unbekannten Parameter enthalten. Die Aussage, dass der wahre unbekannte Parameter mit 95% Wahrscheinlichkeit im berechneten Konfidenzintervall liegt, ist streng genommen falsch. Denn nachdem das Konfidenzintervall festgelegt wurde, liegt der wahre unbekannte Parameter entweder im Intervall oder eben nicht (der Parameter ist keine Zufallsvariable).

(c) Die Herleitung und Berechnung von Konfidenzintervallen basiert immer auf bestimmten (theoretischen) Annahmen, von denen man in der Praxis selten bzw. nie weiß, ob diese erfüllt sind. Daher ist in der Praxis auch nie bekannt, wie groß die Überdeckungswahrscheinlichkeit des berechneten Intervals tatsächlich ist. Entsprechend sollten auch Konfidenzintervalle jeweils mit Vorsicht interpretiert werden.

(d) Wir betrachten die Bestandteile eines Konfidenzintervalls etwas genauer. Konkreter besitzen diese im Allgemeinen die folgende Form

$$\hat{I}(x_1, \dots, x_n) = [S_n(x_1, \dots, x_n) - k_1 \sigma_{S_n}, S_n(x_1, \dots, x_n) + k_2 \sigma_{S_n}] \quad (5.22)$$

Die Bestandteile sind:

- Ein Punktschätzer S_n für den wahren unbekannten Parameter θ .
- Die Standardabweichung des Punktschätzers σ_{S_n} , oft auch Standardfehler genannt.
- Zwei Konstanten $k_1, k_2 \in (0, \infty)$, die üblicherweise von α , n und der Verteilung von S_n (abhängig von P_θ) abhängen.

Darüber hinaus verwendet man die folgenden Bezeichnungen:

Konfidenzlevel: die gewählte Überdeckungswahrscheinlichkeit $1 - \alpha$ für den wahren unbekannten Parameter θ .

Ausgangspunkt: Punktschätzer für den wahren unbekannten Parameter θ , oftmals der Mittelpunkt des Intervalls.

Konfidenzschränken: untere und obere Grenze des Intervalls.

Maximaler Schätzfehler: maximaler Abstand zwischen dem Punktschätzer und den Konfidenzgrenzen.

Wir geben einige Beispiele für Konfidenzintervalle.

Beispiel 5.12. (a) Wir betrachten das Normalverteilungsmodell $\mathcal{P} = \{\mathcal{N}(\mu, \sigma^2) \mid \mu \in \mathbb{R}\}$, wobei wir annehmen, dass $\sigma^2 \in (0, \infty)$ bekannt ist. Wie wir in Abschnitt 5.2 erfahren haben, ist das arithmetische Mittel ein biasfreier und effizienter Schätzer für μ . Da wir davon ausgehen, dass die Beobachtungen

x_1, \dots, x_n Realisationen von unabhängig und identisch verteilten Zufallsvariablen X_1, \dots, X_n sind mit $X_i \sim \mathcal{N}(\mu, \sigma^2)$ ($i = 1, \dots, n$), erhalten wir

$$\text{AM}(x_1, \dots, x_n) \sim \mathcal{N}\left(\mu, \frac{1}{n}\sigma^2\right) \quad (5.23)$$

Es folgt, $\sigma_{S_n} = \frac{1}{\sqrt{n}}\sigma$, was auch der **Standardfehler** (SE = standard error) des arithmetischen Mittels (SEM = standard error of mean) genannt wird. Aufgrund der Symmetrie der Normalverteilung ergibt sich $k_1 = k_2$ und es folgt weiter $k_1 = k_2 = z_{1-\alpha/2}$, das $(1 - \alpha/2)$ -Quantil der Standardnormalverteilung. Demnach lautet das $(1 - \alpha)$ -Konfidenzintervall

$$\text{AM}(x_1, \dots, x_n) \mp z_{1-\alpha/2} \frac{\sigma}{\sqrt{n}} \quad (5.24)$$

In der Praxis ist aber in der Regel auch σ unbekannt und wir müssen es daher ebenfalls schätzen. Da es darum geht, den wahren unbekannten Wert von μ einzufangen, bietet sich als Schätzer die biasfreie Stichprobenvarianz \tilde{S} (d.h. Standardisierung $\frac{1}{n-1}$) an, wobei

$$\frac{(n-1)\tilde{S}}{\sigma} \sim \text{Chisq}(n-1) \quad (5.25)$$

Entsprechend führt uns das Mitschätzen von σ weg von der Normalverteilung und hin zur t-Verteilung mit $n - 1$ Freiheitsgraden (vgl. auch Bemerkung 4.28 (b)). Damit erhalten wir als Konfidenzintervall

$$\text{AM}(x_1, \dots, x_n) \mp t_{n-1;1-\alpha/2} \frac{\sqrt{\tilde{S}(x_1, \dots, x_n)}}{\sqrt{n}} \quad (5.26)$$

wobei $t_{n-1;1-\alpha/2}$ das $(1 - \alpha/2)$ -Quantil der t-Verteilung mit $n - 1$ Freiheitsgraden ist.

(b) Wenn wir umgekehrt das Wahrscheinlichkeitsmodell $\mathcal{P} = \{\mathcal{N}(\mu, \sigma^2) \mid \sigma \in (0, \infty)\}$ betrachten, wobei wir realistischer Weise μ als unbekannt annehmen, erhalten wir das folgende asymmetrische $(1 - \alpha)$ -Konfidenzintervall für σ^2

$$\left[\frac{(n-1)\tilde{S}}{\chi_{n-1;1-\alpha/2}^2}, \frac{(n-1)\tilde{S}}{\chi_{n-1;\alpha/2}^2} \right] \quad (5.27)$$

Hierbei sind $\chi_{n-1;1-\alpha/2}^2$ und $\chi_{n-1;\alpha/2}^2$ das $(1 - \alpha/2)$ - und das $\alpha/2$ -Quantil der χ^2 -Verteilung mit $n - 1$ Freiheitsgraden.

Anmerkung:

Die obigen Konfidenzintervalle sind nicht nur für das Normalverteilungsmodell interessant, sondern können auch als Approximationen für andere Wahrscheinlichkeitsmodelle herangezogen werden. Der Grund dafür ist der zentrale Grenzwertsatz. Dieser besagt, dass sich die Verteilung des geeignet standardisierten arithmetischen Mittels der Realisationen von recht beliebigen unabhängig und identisch verteilten Zufallsvariablen mit wachsender Stichprobengröße n immer mehr einer Normalverteilung annähert.

Wir werden jetzt die Punktschätzungen der maximalen Körpertemperatur für den Datensatz der ITS-Patienten (vgl. Abschnitt 5.2) weiter absichern und zusätzlich 95%-Konfidenzintervalle angeben (d.h. $\alpha = 0.05$). Wir lassen dabei Patient 398 unberücksichtigt. Das Konfidenzintervall für den Mittelwert μ bei unbekannter Standardabweichung σ können wir mit Hilfe der Funktion `meanCI` aus dem Paket "MKinfeR" (Kohl (2022b)) berechnen.

```
1 meanCI(ITSDaten$Temperatur[-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
```

Das Intervall sollte dann in Abhängigkeit von der Genauigkeit der Temperaturmessung gewählt werden; zum Beispiel [37.61, 37.83] oder [37.60, 37.85] oder auch [37.6, 37.9]. Jedes dieser Intervalle überdeckt den wahren unbekannten Mittelwert mit mindestens 95% Wahrscheinlichkeit. Dieses Konfidenzintervall kann auch mit Hilfe der Funktion `t.test` berechnet werden, welche wir in Kapitel 6 noch genauer kennenlernen werden. Vom Ergebnis lassen wir uns jetzt nur das Konfidenzintervall (Komponente `conf.int`) ausgeben.

```
1 t.test(ITSDaten$Temperatur[-398])$conf.int
```

```
[1] 37.61725 37.82363
attr(,"conf.level")
[1] 0.95
```

Da im vorliegenden Beispiel die Fallzahl recht groß ist, könnten wir anstelle des Quantils der t-Verteilung im Sinne des zentralen Grenzwertsatzes auch das entsprechende Quantil der Normalverteilung heranziehen. Wir vergleichen die beiden Quantile

```
1 qt(1 - 0.05 / 2, df = 499 - 1)
```

```
[1] 1.964739
```

```
1 qnorm(1 - 0.05 / 2)
```

```
[1] 1.959964
```

und erhalten eine Differenz von etwas weniger als 0.005. Die Konfidenzgrenzen des approximativen Intervalls sind folglich sehr ähnlich und die Unterschiede dürften jenseits der Messgenauigkeit liegen.

Ein ähnliches approximativer Konfidenzintervall können wir ausgehend von den ML-Schätzern bestimmen. Wir verwenden dazu die Funktion `fitdistr` aus dem Paket "MASS" (Venables and Ripley (2002)) in Kombination mit der Funktion `confint`.

```
1 ## ML-Schätzer
2 ML <- fitdistr(ITSData$Temperatur[-398], densfun = "normal")
3 ## Approximativer Konfidenzintervall
4 confint(ML)
```

```
2.5 %    97.5 %
mean 37.617609 37.823273
sd    1.099298  1.244725
```

Wir erhalten sowohl ein approximativer Konfidenzintervall für den Mittelwert μ also auch für die Standardabweichung σ . Diese Intervalle können wir auch mit Hilfe der Funktion `MLEstimator` aus dem Paket "distrMod" (Kohl and Ruckdeschel (2010)) und der Funktion `confint` berechnen.

```
1 ## Modell
2 Modell <- NormLocationScaleFamily()
3 ## ML-Schätzer
4 ML2 <- MLEstimator(ITSData$Temperatur[-398], Modell)
5 ## Approximativer Konfidenzintervall
6 distrMod::confint(ML2)
```

```
A[n] asymptotic (LAN-based) confidence interval:
2.5 %    97.5 %
mean 37.617609 37.823273
sd    1.099298  1.244725
```

Wir vergleichen das approximative Konfidenzintervall für die Standardabweichung, welches symmetrisch um den ML-Schätzer für die Standardabweichung ist, mit dem asymmetrischen Intervall, welches wir mit Hilfe der χ^2 -Verteilung und den Funktionen `sdCI` und `normCI` aus dem Paket "MKinfer" (Kohl (2022b)) berechnen können.

```
1 ## nur Standardabweichung
2 sdCI(ITSData$Temperatur[-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

```

```

1 ## Mittelwert und Standardabweichung
2 normCI(ITSDaten$Temperatur[-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

```

Das exakte (asymmetrische) Konfidenzintervall weicht etwas vom asymptotischen Intervall ab, wobei die Unterschiede aber lediglich im Promillebereich liegen.

Anmerkung:

Falls die Stichprobengröße n nicht zu klein ist, können die approximativen Konfidenzintervalle verwendet werden, die sich aus dem zentralen Grenzwertsatz ergeben. Die Abbildung 5.2 zeigt das Verhältnis des 95%-Quantil der t-Verteilung mit steigendem Freiheitsgrad zum 95%-Quantil der Standardnormalverteilung. Ab einer Fallzahl von ca. 25 ist der Unterschied zwischen den Quantilen und entsprechend den maximalen Schätzfehlern kleiner als 5%.

Neben den approximativen Intervallen auf Basis der Normalverteilung, gibt es auch noch einen sehr allgemeinen datenbasierten Ansatz, um Konfidenzintervalle zu berechnen, das sogenannte **Bootstrap**. Beim klassischen Bootstrapping werden im einfachsten Fall n (= Stichprobengröße) Beobachtungen **mit Zurücklegen** aus der Stichprobe gezogen. Dieser Zufallsvorgang wird B -Mal ($B \in \mathbb{N}$) wiederholt und für jede dieser Bootstrap-Stichproben werden die entsprechenden Schätzer berechnet. Anschaulich gesprochen tritt somit die Stichprobe an die Stelle der Population und die Bootstrap-Stichproben an die Stelle der ursprünglichen Stichprobe. Ist die ursprüngliche Stichprobe repräsentativ, dann sind es durch

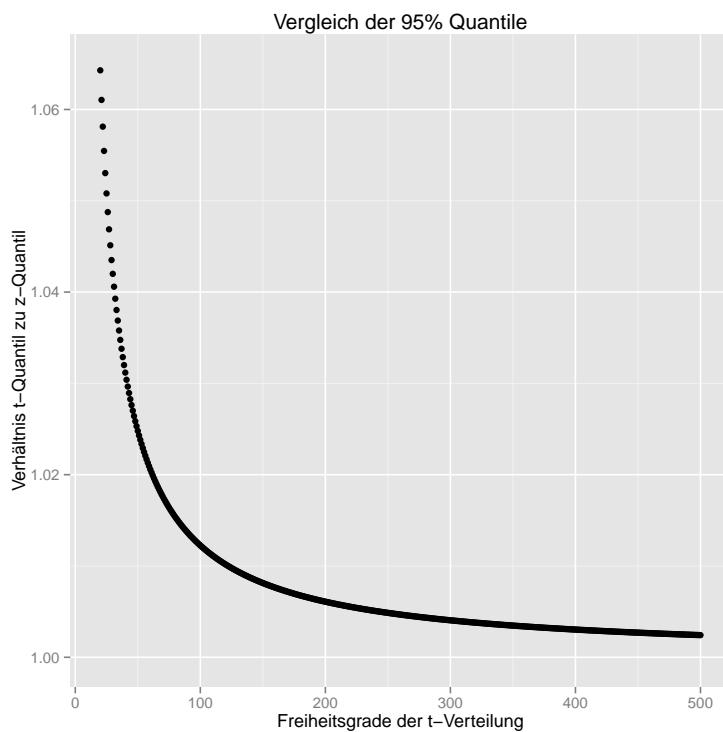


Abbildung 5.2: Verhältnis zwischen den 95%-Quantilen der t- und der Standardnormalverteilung.

die Zufallsauswahl auch die Bootstrap-Stichproben. Jede Bootstrap-Stichprobe entspricht demnach einer möglichen repräsentativen Stichprobe, die so aus der Population hätte gezogen werden können. Aus den Bootstrap-Stichproben gewinnen wir folglich viele mögliche Schätzergebnisse für die Population. Diese Schätzergebnisse geben uns ein Bild von der Variabilität der Schätzergebnisse (Varianz des Schätzers) und wir können dann (ohne bzw. mit wenigen speziellen Modellannahmen!) für den Schätzer ein Konfidenzintervall für den gesuchten Parameter ermitteln. Das Bootstrapping wird daher auch als eine nicht-parametrische Methode bezeichnet; d.h., wir benötigen für das Konfidenzintervall kein konkretes parametrisches Verteilungsmodell.

Anmerkung:

Im Fall von $n = 20$ und $B = 2000$ ist die Wahrscheinlichkeit zwei oder mehr gleiche Stichproben zu erhalten, kleiner als 5% (Abschnitt 8.1 in Chernick and LaBudde (2011)). Die Empfehlungen zur Anzahl von Bootstrap-Wiederholungen beginnen bei wenigen Hundert Wiederholungen ($B = 399$ oder $B = 599$) und enden bei Werten von $B = 10000$ oder sogar $B = 100000$. Bei R Funktionen, die mit Bootstrap oder anderen Simulationstechniken arbeiten, findet man oft voreingestellte Werte von $B = 999$ oder $B = 9999$. Im Fall von Konfidenzintervallen empfehlen sich Werte von $B = 999$ oder größer. Wenn die Rechenzeit gering ist bzw. keine Rolle spielt, sollte $B = 9999$ in den meisten Fällen sehr gute und stabile Ergebnisse liefern.

Die entsprechende Funktionalität ist im recommended Paket "boot" (Canty and Ripley (2021)) implementiert. Es gibt dort die Möglichkeit mit der Funktion `boot.ci` fünf verschiedene Arten von Bootstrap-Konfidenzintervallen zu berechnen. Diese sind "norm", "basic", "stud", "perc" und "bca". Das ein-

fachste der Intervalle ist das Perzentilintervall ("perc"). Hier besteht das Konfidenzintervall aus den empirischen Quantilen ($\alpha/2$ und $1 - \alpha/2$) der Bootstrap-Schätzungen. Die BCa-Methode ist eine korrigierte und adjustierte Variante (bias corrected and accelerated) der Perzentil-Methode, die in vielen Fällen die besten Ergebnisse liefert. Die Methode "norm" basiert auf der Formel (5.24) wobei Mittelwert und Standardabweichung aus den Bootstrap-Schätzungen berechnet werden. Im Fall der Methode "basic" wird wieder die Normalapproximation (5.24) verwendet, dieses Mal für die am Mittelwerte der Originalstichprobe zentrierten Bootstrap-Schätzungen. Die Methode "stud" schließlich verwendet die am Mittelwerte der Originalstichprobe zentrierten und durch die Standardabweichung des Schätzers (für die jeweilige Bootstrap-Stichprobe berechnet) standardisierten Bootstrap-Schätzungen (dies nennt man auch Studentisieren), man könnte also auch von einem gebootstrapten z-Score sprechen; d.h., man muss hierbei zusätzlich die Standardabweichung (oder Varianz) des Schätzers kennen und diese dann für jede Bootstrap-Stichprobe zusätzlich berechnen. Auch hinter der Methode "stud" steckt demnach die Normalapproximation. Durch die zusätzliche Standardisierung wird (zumindest in der Theorie) eine Reduktion des Fehlers erreicht (Größenordnung des Fehlers $O(n^{-1})$ anstelle von $O(n^{-1/2})$ wie im Fall des Typs "basic"). Für weitere Einzelheiten zu Bootstrap-Konfidenzintervallen verweisen wir auf Kapitel 5 von Davison and Hinkley (1997), Kapitel 3 von Chernick and LaBudde (2011) und das Supplement zu Kohl and Münch (2022c).

Die Funktionen `normCI`, `meanCI` und `sdCI` aus dem Paket "MKinfer" (Kohl (2022b)) enthalten vereinfachte Schnittstellen zu den Funktionen des Paketes "boot" (Canty and Ripley (2021)).

```
1 normCI(ITSDaten$Temperatur[-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, 37.83 )
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,   1.252 )  ( 1.093,   1.251 )  ( 1.098,   1.256 )

Level      Percentile        BCa
95%    ( 1.096,   1.253 )  ( 1.102,   1.261 )
Calculations and Intervals on Original Scale

sample estimates:
mean      sd
37.720441 1.173187

```

Die Ergebnisse sind sehr ähnlich zu den bisherigen Ergebnissen.

Anmerkung:

Bei kleinen bis moderaten Stichprobengrößen $n \leq 50$ muss man davon ausgehen, dass auf dem zentralen Grenzwertsatz basierende asymptotische Konfidenzintervalle als grobe Näherungen zu betrachten sind. Das Bootstrapping funktioniert erfahrungsgemäß auch bei kleinen Stichprobengrößen bis $n = 10$ oder sogar noch etwas kleiner sehr gut. Man sollte sich aber auch hier bewusst sein, dass Bootstrap-Konfidenzintervalle für kleine bis moderate Fallzahlen ($10 \leq n \leq 50$) und schiefe Verteilungen tendenziell etwas zu kurz sind. Chernick and LaBudde (2011) empfehlen in Abschnitt 3.7 in diesem Fall den Typ "stud" zu verwenden.

Im folgenden Beispiel behandeln wir das Bernoulli-Modell.

Beispiel 5.13. Wir betrachten das Wahrscheinlichkeitsmodell $\mathcal{P} = \{\text{Bernoulli}(p) \mid p \in (0, 1)\}$. Wie wir in Abschnitt 5.2 erfahren haben, ist die relative Häufigkeit \hat{p} der ML-Schätzer für p und ist zudem biasfrei und effizient. Da es sich bei der Bernoulli-Verteilung um eine diskrete Verteilung handelt, ist auch die Verteilung \hat{p} diskret und die Quantile diskreter Verteilungen sind nicht notwendigerweise eindeutig. Entsprechend gibt es eine ganze Reihe von Vorschlägen, wie ein Konfidenzintervall für die Wahrscheinlichkeit p berechnet werden kann; zum Beispiel das Clopper-Pearson- oder das Wilson-Intervall. Wir verzichten hier auf die Angabe der genauen Formeln und verweisen auf Abschnitt 6.6 von Hedderich and Sachs (2018).

Ein approximatives Konfidenzintervall für p ergibt sich unter Verwendung des zentralen Grenzwertsatzes

$$\left(\hat{p} \mp \frac{1}{2n} \right) \mp z_{1-\alpha/2} \sqrt{\frac{\hat{p}(1-\hat{p})}{n}} \quad (5.28)$$

Der Korrekturterm $\frac{1}{2n}$ heißt **Stetigkeitskorrektur** und führt durch eine geringe Vergrößerung des Konfidenzintervalls zu einer Verbesserung der Approximation. Weiter ist $z_{1-\alpha/2}$ das $(1 - \alpha/2)$ -Quantil der Standardnormalverteilung. Für die Überprüfung der Anwendbarkeit des asymptotischen Intervalls gibt es verschiedene Faustformeln. Eine davon lautet etwa, dass

$$n\hat{p} > 5 \quad \text{und} \quad n(1 - \hat{p}) > 5 \quad (5.29)$$

sein müssen; d.h., je mehr sich \hat{p} der Null oder der Eins nähert, desto größer muss die Stichprobe sein. Gehen wir von einem Ziehen ohne Zurücklegen aus und ist die betrachtete Population eher klein mit $N \in \mathbb{N}$ Mitgliedern, so empfiehlt es sich, das folgende leicht modifizierte Konfidenzintervall zu verwenden

$$\left(\hat{p} \mp \frac{1}{2n} \right) \mp z_{1-\alpha/2} \sqrt{\frac{\hat{p}(1-\hat{p})}{n} \frac{N-n}{N-1}} \quad (5.30)$$

Der zusätzliche Faktor $\frac{N-n}{N-1}$, den wir bereits in Bemerkung 4.8 (c) kennengelernt haben, heißt **Endlichkeitskorrektur** und spiegelt den Unterschied zwischen Ziehen mit und ohne Zurücklegen wider.

Wir betrachten die Prävalenz von Leberversagen auf der ITS und sichern die Schätzung durch ein Konfidenzintervall zusätzlich ab. Es gibt verschiedene Pakete, mit denen “exakte” Konfidenzintervalle berechnet werden können. Wir verwenden die Funktion `binomCI` aus dem Paket "MKinfer" (Kohl (2022b)). Wir berechnen das Wilson-, das Clopper-Pearson- und das Agresti-Coull-Intervall. Hierzu benötigen wir die Anzahl der Patienten mit Leberversagen und die Gesamtzahl der Patienten.

```
1 ## Häufigkeit von Leberversagen
2 table(ITSDaten$Leberversagen)
```

0	1
480	20

```
1 ## Wilson-Intervall
2 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
```

```
1 ## Clopper-Pearson-Intervall
2 binomCI(x = 20, n = 500, method = "clopper-peerson")
```

```
clopper-peerson confidence interval

95 percent confidence interval:
```

```

    2.5 %      97.5 %
prob 0.02460131 0.06110261

sample estimate:
prob
0.04

```

```

1 ## Agresti-Coull-Intervall
2 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

```

Wir erhalten leichte Unterschiede zwischen den Intervallen, insbesondere basiert das Agresti-Coull-Intervall auch auf einem anderen Schätzer für p . Das asymptotische Intervall mit und ohne Stetigkeitskorrektur können wir ebenfalls mit Hilfe der Funktion `binomCI` berechnen.

```

1 ## ohne Stetigkeitskorrektur
2 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

```

```

1 ## mit Stetigkeitskorrektur
2 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
```

Schließlich kann die Funktion `binomCI` auch dazu verwendet werden, um Bootstrap-Konfidenzintervalle zu berechnen.

```
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%   ( 0.0229,   0.0569 )   ( 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                           0.008683610
```

Wir können ein asymptotisches Konfidenzintervall auch wieder mit Hilfe der Funktion `MLEstimator` aus dem Paket "distrMod" (Kohl and Ruckdeschel (2010)) und der Funktion `confint` berechnen. Das Bernoulli-Modell definieren wir mit Hilfe der Funktion `BinomFamily` und dem Parameterwert `size = 1`.

```

1 ## Bernoulli-Modell
2 Modell <- BinomFamily(size = 1)
3 ## ML-Schätzer
4 MLp <- MLEstimator(ITSData$Leberversagen, Modell)
5 MLp

```

```

Evaluations of Maximum likelihood estimate:
-----
0.040000000
(0.008763561)

```

```

1 ## Konfidenzintervall
2 distrMod::confint(MLp)

```

```

A[n] asymptotic (LAN-based) confidence interval:
      2.5 %    97.5 %
[1,] 0.02282374 0.05717626

```

Das Ergebnis entspricht dem obigen asymptotischen Konfidenzintervall ohne die Stetigkeitskorrektur. Grob zusammengefasst, können wir für diese ausgewählte Population von Intensivpatienten mit recht hoher Sicherheit von einer Prävalenz für Leberversagen im Bereich von 2.2% bis 6.1% ausgehen.

Anmerkung:

Da es meistens mehr als eine Möglichkeit gibt, ein Konfidenzintervall für einen Parameter zu bestimmen, ist es in der Praxis zu empfehlen, neben dem Intervall selbst auch die Art des Intervalls anzugeben. Nur so kann ein Leser die Analyse und die Ergebnisse wirklich nachvollziehen.

Eine sehr interessante Möglichkeit, die Lage und Skala der Daten zu beschreiben, stellen Median und MAD dar. Zum einen sind die beiden Schätzverfahren sehr robust, zum anderen ist es nicht nötig, sich auf ein parametrisches Modell festzulegen.

Beispiel 5.14. Es seien $x_{(1)}, x_{(2)}, \dots, x_{(n)}$ die aufsteigend sortierten Beobachtungen. Dann ergibt sich das $(1 - \alpha)$ -Konfidenzintervall des Medians als

$$[x_{(k)}, x_{(n-k+1)}] \quad (5.31)$$

wobei $k \in \mathbb{N}$ so zu bestimmen ist, dass die folgende Ungleichung gilt

$$1 - 2 \sum_{i=1}^{k-1} \binom{n}{i} 0.5^n \geq 1 - \alpha \quad (5.32)$$

Dies lässt sich auch auf den MAD übertragen, indem man ihn als Median von $|x_1 - M|, \dots, |x_n - M|$ mit $M = \text{median}(x_1, \dots, x_n)$ betrachtet. Im Fall der Normalverteilung standardisiert man den MAD üblicherweise noch mit 1.4826, um einen konsistenten Schätzer für die Standardabweichung zu erhalten; vergleiche auch Gleichung (2.3).

Wir betrachten die maximale Körpertemperatur der ITS-Patienten und bestimmen 95%-Konfidenzintervalle für Median und MAD. Hierzu verwenden wir die Funktionen `medianCI` und `madCI` aus dem Paket "MKinfer" (Kohl (2022b)). Im Fall des MAD wählen wir die mit 1.4826 standardisierte Version.

```
1 ## Exaktes Konfidenzintervall für den Median
2 medianCI(ITSDaten$Temperatur)
```

```
exact confidence interval

95 percent confidence interval:
 2.5 % 97.5 %
median 37.5   37.8

sample estimate:
median
 37.7
```

```
1 ## Exaktes Konfidenzintervall für den MAD
2 madCI(ITSDaten$Temperatur)
```

```
exact confidence interval

95 percent confidence interval:
 2.5 % 97.5 %
MAD 1.03782 1.33434

sample estimate:
MAD
1.18608
```

Da die Fallzahl in unserem Beispiel recht groß ist, können wir auch auf die asymptotischen Konfidenzintervalle ausweichen. Es ergibt sich

```
1 ## Asymptotisches Konfidenzintervall für den Median
2 medianCI(ITSDaten$Temperatur, method = "asymptotic")
```

```
asymptotic confidence interval

95 percent confidence interval:
 2.5 % 97.5 %
median 37.5   37.8

sample estimate:
median
 37.7
```

```
1 ## Asymptotisches Konfidenzintervall für den MAD
2 madCI(ITSDaten$Temperatur, method = "asymptotic")
```

```
asymptotic confidence interval

95 percent confidence interval:
 2.5 % 97.5 %
MAD 1.03782 1.33434

sample estimate:
MAD
1.18608
```

Wir erhalten identische Ergebnisse wie im Fall der exakten Intervalle. Als Alternative zu den asymptotischen Intervallen eignen sich auch wieder Bootstrap-Konfidenzintervalle, die wir ebenfalls mit den Funktionen `medianCI` und `madCI` aus dem Paket "MKinfer" (Kohl (2022b)) berechnen können.

```
1 ## Bootstrap-Konfidenzintervall für den Median
2 medianCI(ITSDaten$Temperatur, 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%   (37.54, 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
```

```
1 ## Bootstrap-Konfidenzintervall für den MAD
2 madCI(ITSDaten$Temperatur, 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%   ( 1.082,  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

```

Auch die Bootstrap-Intervalle sind sehr ähnlich zu den exakten und asymptotischen Intervallen, was aufgrund der großen Fallzahl nicht überrascht.

Insgesamt sind die Intervalle für Median und MAD etwas länger als im Fall des arithmetischen Mittels und der (Stichproben-)Standardabweichung. Das ist der Preis, den wir für die nicht-parametrische Natur dieser Schätzverfahren und deren Robustheit zu bezahlen haben.

Im Folgenden werfen wir einen Blick auf die MD-Schätzer. Hier ist die Angabe von exakten oder asymptotischen Konfidenzintervallen schwierig, da die (exakte und asymptotische) Verteilung dieser Schätzer nur schwer oder gar nicht herzuleiten ist. Im Fall des CvM-MD-Schätzers können wir mit Hilfe der Funktion `CvMMDEstimator` aus dem Paket "distrMod" (Kohl and Ruckdeschel (2010)) und der Funktion `confint` ein asymptotisches Konfidenzinverall berechnen. Wir betrachten zunächst wieder die maximale Körpertemperatur der ITS-Patienten.

```

1 ## Modell
2 Modell <- NormLocationScaleFamily()
3 ## CvM-MD-Schätzer inkl. Varianz
4 MD <- CvMMDEstimator(ITSData$Temperatur, Modell)
5 ## 95%-Konfidenzintervall
6 distrMod::confint(MD)

A[n] asymptotic (LAN-based) confidence interval:
      2.5 %    97.5 %
mean 37.535623 37.807908
sd    1.051812  1.227027

```

Die Konfidenzintervalle sind etwas länger als im Fall des ML-Schätzers, wobei wir aber aufgrund der Robustheit des CvM-MD-Schätzers Patient 398 nicht von den Berechnungen ausschließen mussten.

Wir können in ähnlicher Weise auch das Konfidenzintervall für die Prävalenz von Leberversagen auf der ITS berechnen.

```

1 ## Modell
2 Modell <- BinomFamily(size = 1)
3 ## CvM-MD-Schätzer inkl. Varianz
4 MDp <- CvMMDEstimator(ITSData$Leberversagen, Modell)
5 ## 95%-Konfidenzintervall
6 distrMod::confint(MDp)

A[n] asymptotic (LAN-based) confidence interval:
      2.5 %    97.5 %
prob 0.02282334 0.05717567

```

Die Ergebnisse sind nahezu identisch zum ML-Schätzer. Auch im Fall der MD-Schätzer kann man Bootstrap-Konfidenzintervalle verwenden. In diesem Fall ist uns jedoch keine Funktion bekannt, die dies direkt erledigt. Wir verwenden stattdessen die Funktionen `boot` und `boot.ci` aus dem Paket "boot" (Canty and Ripley (2021)). Wir zeigen dies exemplarisch für den KS-MD-Schätzer, für den dies von besonderem Interesse ist, da weder ein exaktes noch ein asymptotisches Konfidenzintervall bekannt ist. Zunächst benötigen wir eine Funktion, welche den Schätzwert berechnet und diesen als Skalar (Zahl) zurückgibt. Die Funktion besitzt zwei Parameter. Der erste Parameter repräsentiert den Zahlenvektor, der die vollständigen Daten enthält. Der zweite Parameter die Indices der Einträge des Zahlenvektors, die für die jeweilige Bootstrap-Stichprobe ausgewählt wurden.

```

1 ## x: Vektor der Beobachtungen
2 ## i: Vektor der Indizes der Bootstrap-Stichprobe
3 KSMDEst <- function(x, i){
4   estimate(KolmogorovMDEstimator(x[i], ParamFamily = NormLocationScaleFamily()))
5 }

```

Mit dieser Funktion berechnen wir zunächst den KS-MD-Schätzer für 999 Bootstrap-Stichproben. Die Berechnungen dauern einige Minuten. Anschließend ermitteln wir ausgehend von diesen Schätzwerten die Bootstrap-Konfidenzintervalle.

```

1 ## Bootstrap-Schätzungen
2 boot.out <- boot(ITSData$Temperatur, statistic = KSMDEst, R = 999)
3 ## Bootstrap-Konfidenzintervalle
4 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%  (37.57, 37.79 )  (37.57, 37.79 )

Level      Percentile          BCa
95%  (37.57, 37.78 )  (37.58, 37.79 )
Calculations and Intervals on Original Scale

```

```

1 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%   ( 1.063,   1.249 )  ( 1.068,   1.254 )

Level      Percentile          BCa
95%   ( 1.033,   1.218 )  ( 1.064,   1.240 )
Calculations and Intervals on Original Scale
Some BCa intervals may be unstable

```

Wir erhalten sehr ähnliche Ergebnisse wie bei den anderen Schätzern.

Abschließend wollen wir noch die RMX-Schätzer auf die ITS-Daten anwenden. Wir berechnen zunächst den RMX-Schätzer für die Temperatur-Daten mit Hilfe der Funktion `roblox` aus dem Paket "RobLox" (Kohl (2019)) und den gleichen Einstellungen wie im Abschnitt 5.2. Danach können wir mit der Funktion `confint` das zugehörige asymptotische Konfidenzintervall berechnen.

```

1 ALest <- roblox(ITSDaten$Temperatur, eps.lower = 1/500,
2                   eps.upper = 0.05, k = 3)
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

```

Das obige Konfidenzintervall ignoriert eine möglichen Bias, der unvermeidlich ist, wenn man die beschriebenen Kontaminationsumgebungen um das parametrische Modell herum betrachtet. Wir können auch Konfidenzintervalle berechnen, die zusätzlich den maximalen (asymptotischen) Bias berücksichtigen.

```

1 distrMod::confint(ALest, symmetricBias = TRUE)

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

```

Wir sehen, dass die Konfidenzintervalle dadurch etwas länger, also konservativer, werden. Wir können alternativ auch wieder die Funktion `rmx` aus dem Paket "rmx" (Kohl (2022c)) verwenden. Auch hier können wir mit der Funktion `confint` die entsprechenden Konfidenzintervalle ausgeben.

```
1 RMXest ← rmx(ITS Daten$Temperatur, model = "norm",
2                  eps.lower = 1/500, eps.upper = 0.05)
3 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
```

```
1 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
```

Die Möglichkeit von Bootstrap-Konfidenzintervallen ist im Paket "rmx" integriert.

```
1 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)   (37.46, 37.67)   (37.45, 37.68)

Level      Percentile       BCa
95%    (37.61, 37.82)   (37.50, 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)   ( 1.020,  1.187)   ( 1.015,  1.196)

Level      Percentile       BCa
95%    ( 1.092,  1.259)   ( 1.023,  1.188)
Calculations and Intervals on Original Scale
Some BCa intervals may be unstable

RMX estimates:
      mean        sd
37.642900 1.139516

```

Alternativ können wir hier auch wieder Bootstrap-Konfidenzintervalle berechnen, indem wir die Funktionen `boot` und `boot.ci` aus dem Paket "boot" (Canty and Ripley (2021)) verwenden. Die folgende Funktion führt für die Bootstrap-Stichproben die notwendigen Berechnungen durch, wobei wir neben dem Schätzwert auch die asymptotische Varianz des Schätzers berechnen. Der Rückgabewert der Funktion ist demnach ein Vektor der Länge vier, dessen ersten beiden Einträge die Schätzwerte für Mittelwert und SD sind und dessen dritter und vierter Eintrag die asymptotische Varianz für diese beiden Schätzungen ist. Die Funktion besitzt die gleichen Parameter wie die Funktion, die wir oben für den KS-MD-Schätzer angegeben haben.

```

1 ## x: Vektor der Beobachtungen
2 ## i: Vektor der Indizes der Bootstrap-Stichprobe
3 RMXEst <- function(x, i){
4   res <- roblox(x[i], eps.lower = 1/500, eps.upper = 0.05, k = 3)
5   c(estimate(res), diag(asvar(res)))
6 }

```

Wir berechnen die Bootstrap-Konfidenzintervalle. Im Fall des studentisierten Intervalls wird folglich zusätzlich die asymptotische Varianz zur Berechnung herangezogen.

```

1 ## Bootstrap-Schätzungen
2 boot.out <- boot(ITSDaten$Temperatur, statistic = RMXEst, R = 999)
3 ## Bootstrap-Konfidenzintervall für den Mittelwert
4 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%   (37.54, 37.75)    (37.54, 37.75)    (37.54, 37.75)

Level      Percentile          BCa
95%   (37.54, 37.75)    (37.54, 37.75)
Calculations and Intervals on Original Scale

```

```

1 ## Bootstrap-Konfidenzintervall für die Standardabweichung
2 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%   ( 1.056, 1.229)    ( 1.049, 1.228)    ( 1.056, 1.235)

Level      Percentile          BCa
95%   ( 1.053, 1.231)    ( 1.056, 1.234)
Calculations and Intervals on Original Scale

```

Die Ergebnisse sind sehr ähnlich zum asymptotischen Intervall ohne Beachtung des Bias.

Anmerkung:

Neben den vorgestellten Möglichkeiten, gibt es in R eine Vielzahl weiterer Möglichkeiten Konfidenzintervalle zu berechnen. Insbesondere werden üblicherweise bei der Berechnung statistischer Tests, die wir in Kapitel 6 behandeln werden, Konfidenzintervalle mitberechnet.

Zur Verdeutlichung der in diesem Kapitel vorgestellten statistischen Verfahren werden wir im Folgenden die Schritte zur Schätzung der Parameter, zur Validierung der Modelle und zur Berechnung der zugehörigen Konfidenzintervalle nochmals an einem Beispiel wiederholen. Wir verwenden hierzu den SAPS-II Score der Patienten, die nach Hause entlassen wurden. Wir gehen davon aus, dass wir die Daten durch eine Gammaverteilung beschreiben können. Wir legen zunächst einen entsprechenden Teildatensatz an, um die einzelnen Schritte zu vereinfachen.

```
1 ITSDaten.nachHause ← ITSDaten[ITSDaten$Ergebnis == "nach Hause",]
```

Als Schätzer verwenden wir den ML-, den CvM-MD- und den RMX-Schätzer. Im Fall des RMX-Schätzers verwenden wir die Funktion `roptest` aus dem Paket "ROptEst" (Kohl and Ruckdeschel (2019)) und gehen von 0% bis 5% Fehlern aus, wie dies bei Routinedaten üblicher Weise der Fall ist.

```
1 Modell ← GammaFamily()
2 Mlest ← MLEstimator(ITSDaten.nachHause$SAPS.II, Modell)
3 Mlest
```

```
Evaluations of Maximum likelihood estimate:
-----
scale      shape
5.5804692   6.9961941
(0.6131542) (0.7414555)
```

```
1 MDest ← CvMMDEstimator(ITSDaten.nachHause$SAPS.II, Modell)
2 MDest
```

```
Evaluations of Minimum CvM distance estimate ( mu = model distr. ) :
-----
scale      shape
6.0324283   6.4628936
(0.8239082) (0.8509984)
```

```
1 RMXest ← roptest(ITSDaten.nachHause$SAPS.II, Modell,
2                      eps.lower = 0, eps.upper = 0.05, steps = 3)
3 RMXest
```

```
Evaluations of 3-step estimate:
-----
scale      shape
5.9019056   6.6572947
(0.7183910) (0.7803873)
```

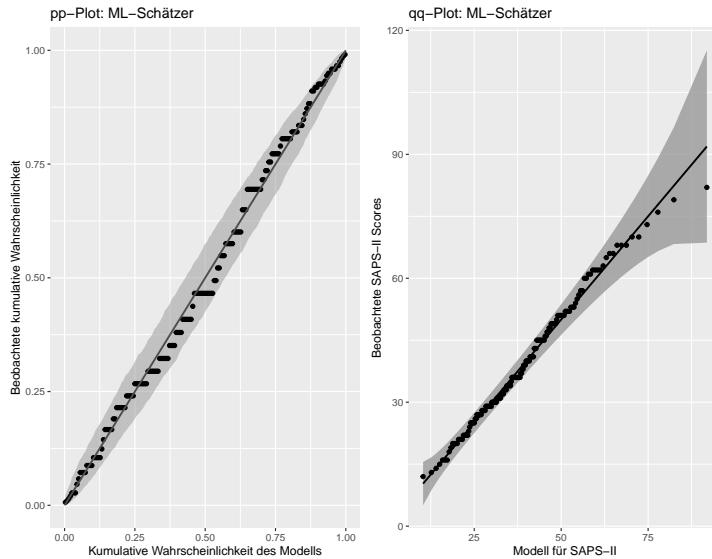
Wir verwenden pp- und qq-Plot, um die geschätzten Modelle zu validieren. Wir beginnen mit dem ML-Schätzer.

```
1 gg1 ← ggplot(ITSDaten.nachHause, aes(sample = SAPS.II)) +
2   qqplotr::stat_pp_band(dparams = list(scale = 5.58, shape = 7.0),
3                         distribution = "gamma") +
4   qqplotr::stat_pp_point(dparams = list(scale = 5.58, shape = 7.0),
5                         distribution = "gamma") +
6   qqplotr::stat_pp_line() +
7   xlab("Kumulative Wahrscheinlichkeit des Modells") +
8   ylab("Beobachtete kumulative Wahrscheinlichkeit") +
9   ggtitle("pp-Plot: ML-Schätzer")
10 gg2 ← ggplot(ITSDaten.nachHause, aes(sample = SAPS.II)) +
```

```

11 qqplotr::stat_qq_band(dparams = list(scale = 5.58, shape = 7.0),
12                         distribution = "gamma", identity = TRUE) +
13 qqplotr::stat_qq_point(dparams = list(scale = 5.58, shape = 7.0),
14                         distribution = "gamma") +
15 qqplotr::stat_qq_line(dparams = list(scale = 5.58, shape = 7.0),
16                         distribution = "gamma", identity = TRUE) +
17 xlab("Modell für SAPS-II") +
18 ylab("Beobachtete SAPS-II Scores") +
19 gtitle("qq-Plot: ML-Schätzer")
20 grid.arrange(gg1, gg2, nrow = 1)

```

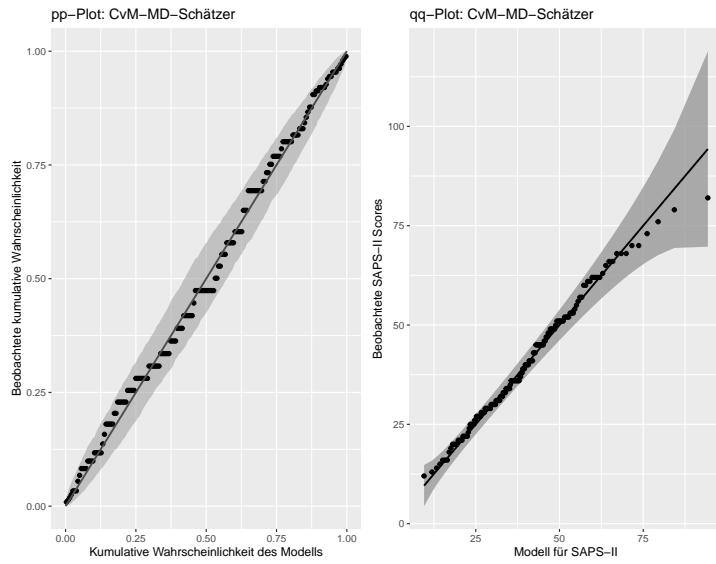


Wir erhalten eine gute Übereinstimmung von Daten und Modell. Wir untersuchen als Nächstes den CvM-MD-Schätzer.

```

1 gg1 <- ggplot(ITSDaten.nachHause, aes(sample = SAPS.II)) +
2   qqplotr::stat_pp_band(dparams = list(scale = 6.03, shape = 6.46),
3                         distribution = "gamma") +
4   qqplotr::stat_pp_point(dparams = list(scale = 6.03, shape = 6.46),
5                         distribution = "gamma") +
6   qqplotr::stat_pp_line() +
7   xlab("Kumulative Wahrscheinlichkeit des Modells") +
8   ylab("Beobachtete kumulative Wahrscheinlichkeit") +
9   gtitle("pp-Plot: CvM-MD-Schätzer")
10 gg2 <- ggplot(ITSDaten.nachHause, aes(sample = SAPS.II)) +
11   qqplotr::stat_qq_band(dparams = list(scale = 6.03, shape = 6.46),
12                         distribution = "gamma", identity = TRUE) +
13   qqplotr::stat_qq_point(dparams = list(scale = 6.03, shape = 6.46),
14                         distribution = "gamma") +
15   qqplotr::stat_qq_line(dparams = list(scale = 6.03, shape = 6.46),
16                         distribution = "gamma", identity = TRUE) +
17   xlab("Modell für SAPS-II") +
18   ylab("Beobachtete SAPS-II Scores") +
19   gtitle("qq-Plot: CvM-MD-Schätzer")
20 grid.arrange(gg1, gg2, nrow = 1)

```

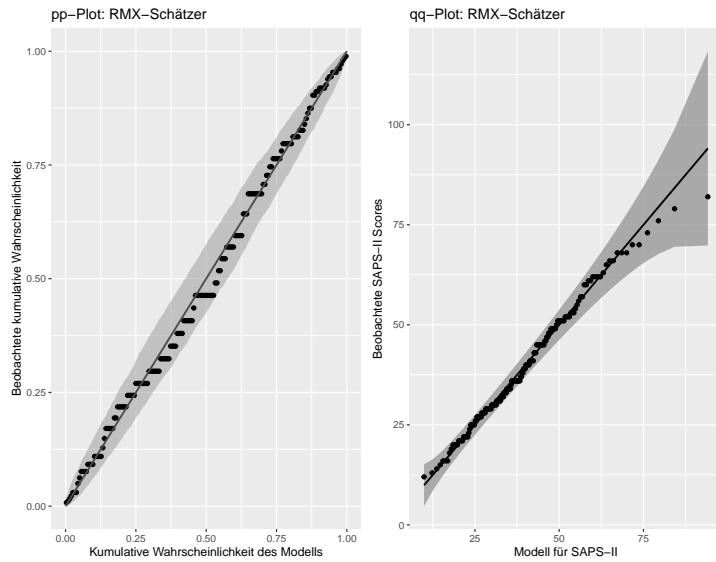


Auch hier sehen wir eine gute Übereinstimmung von Daten und Modell. Wir betrachten schließlich noch den RMX-Schätzer.

```

1 gg1 <- ggplot(ITSData.nachHause, aes(sample = SAPS.II)) +
2   qqplotr::stat_pp_band(dparams = list(scale = 5.90, shape = 6.66),
3                         distribution = "gamma") +
4   qqplotr::stat_pp_point(dparams = list(scale = 5.90, shape = 6.66),
5                         distribution = "gamma") +
6   qqplotr::stat_pp_line() +
7   xlab("Kumulative Wahrscheinlichkeit des Modells") +
8   ylab("Beobachtete kumulative Wahrscheinlichkeit") +
9   ggtitle("pp-Plot: RMX-Schätzer")
10 gg2 <- ggplot(ITSData.nachHause, aes(sample = SAPS.II)) +
11   qqplotr::stat_qq_band(dparams = list(scale = 5.90, shape = 6.66),
12                         distribution = "gamma", identity = TRUE) +
13   qqplotr::stat_qq_point(dparams = list(scale = 5.90, shape = 6.66),
14                         distribution = "gamma") +
15   qqplotr::stat_qq_line(dparams = list(scale = 5.90, shape = 6.66),
16                         distribution = "gamma", identity = TRUE) +
17   xlab("Modell für SAPS-II") +
18   ylab("Beobachtete SAPS-II Scores") +
19   ggtitle("qq-Plot: RMX-Schätzer")
20 grid.arrange(gg1, gg2, nrow = 1)

```

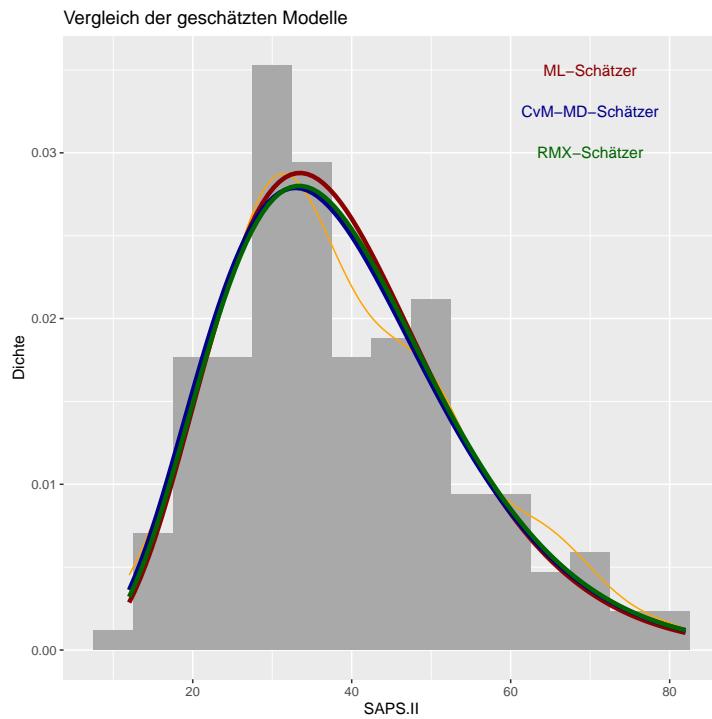


Auch im Fall des RMX-Schätzers sehen wir eine gute Übereinstimmung zwischen geschätztem Modell und den Daten. Insgesamt sehen wir nur wenig Unterschiede zwischen den drei Schätzern. Da wir insbesondere keine Unterschiede zwischen dem ML-Schätzer und den beiden robusten Schätzern sehen, können wir dies als ein Zeichen dafür interpretieren, dass die Daten keine auffälligen Abweichungen vom angenommenen Modell enthalten. Wir stellen die Dichten der geschätzten Modelle zusammen graphisch dar.

```

1 ggplot(ITSDaten.nachHause, aes(x=SAPS.II)) +
2   geom_histogram(aes(y=after_stat(density)), binwidth = 5,
3                 fill = "darkgrey") +
4   geom_density(color = "orange") + ylab("Dichte") +
5   stat_function(fun = dgamma, args = list(scale = 5.58, shape = 7.0),
6                 color = "darkred", lwd = 1.5) +
7   stat_function(fun = dgamma, args = list(scale = 6.03, shape = 6.46),
8                 color = "darkblue", lwd = 1.5) +
9   stat_function(fun = dgamma, args = list(scale = 5.90, shape = 6.66),
10                color = "darkgreen", lwd = 1.5) +
11  annotate("text", x = 70, y = 0.035, col = "darkred",
12          label = "ML-Schätzer") +
13  annotate("text", x = 70, y = 0.0325, col = "darkblue",
14          label = "CvM-MD-Schätzer") +
15  annotate("text", x = 70, y = 0.030, col = "darkgreen",
16          label = "RMX-Schätzer") +
17  ggtitle("Vergleich der geschätzten Modelle")

```



Auch hier zeigt sich eine große Ähnlichkeit zwischen den geschätzten Modellen. Wir berechnen die asymptotischen Konfidenzintervalle für die drei Schätzer.

```
1 distrMod::confint(Mlest)
```

```
A[n] asymptotic (LAN-based) confidence interval:
      2.5 %   97.5 %
scale  4.378709 6.782229
shape  5.542968 8.449420
```

```
1 distrMod::confint(MDest)
```

```
A[n] asymptotic (LAN-based) confidence interval:
      2.5 %   97.5 %
scale  4.417598 7.647259
shape  4.794967 8.130820
```

```
1 distrMod::confint(RMXest)
```

```
A[n] asymptotic (LAN-based) confidence interval:
      2.5 %   97.5 %
scale  4.493885 7.309926
shape  5.127764 8.186826
```

Die Konfidenzintervalle überlappen sich deutlich. Wir berechnen für ML- und CvM-MD-Schätzer die zugehörigen Bootstrap-Konfidenzintervalle. Im Fall des RMX-Schätzers verzichten wir aufgrund der hohen Rechenzeit auf die Berechnungen. Wir geben jedoch den entsprechenden R-Code dafür an. Wir definieren

zuerst die Funktionen, mit denen wir für die Bootstrap-Stichproben die Schätzer und die (asymptotische) Varianzen der Schätzer berechnen können.

```

1 ## x: Vektor der Beobachtungen
2 ## i: Vektor der Indizes der Bootstrap-Stichprobe
3 MLEst ← function(x, i){
4   res ← MLEstimator(x[i], ParamFamily = GammaFamily())
5   c(estimate(res), diag(asvar(res)))
6 }
7 MDEst ← function(x, i){
8   res ← CvMMDEstimator(x[i], ParamFamily = GammaFamily())
9   c(estimate(res), diag(asvar(res)))
10 }
11 RMXEst ← function(x, i){
12   res ← roptest(x[i], ParamFamily = GammaFamily(),
13                 eps.lower = 0, eps.upper = 0.05, steps = 3)
14   c(estimate(res), diag(asvar(res)))
15 }
```

Da die Berechnungen einige Zeit in Anspruch nehmen werden, wollen wir diese durch Parallelisieren beschleunigen. Auch dies ist mit der Funktion `boot` möglich. Zunächst bestimmen wir mit Hilfe der Funktion `detectCores` aus dem Paket "parallel" (R Core Team (2022a)) die Anzahl der zur Verfügung stehenden CPU Kerne. Wir verwenden für die Berechnung dann alle außer einem Kern.

```
1 nr.cpus ← detectCores()-1
```

Wir beginnen mit den Berechnungen für den ML-Schätzer.

```

1 ## Bootstrap-Schätzungen
2 boot.out ← boot(ITSDaten.nachHause$SAPS.II, statistic = MLEst, R = 999,
3                  parallel = "multicore", ncpus = nr.cpus)
4 ## Bootstrap-Konfidenzintervall für scale
5 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%    ( 4.510 ,   6.695 )    ( 4.496 ,   6.621 )    ( 4.675 ,   6.862 )

Level      Percentile          BCa
95%    ( 4.540 ,   6.664 )    ( 4.560 ,   6.741 )
Calculations and Intervals on Original Scale
```

```

1 ## Bootstrap-Konfidenzintervall für shape
2 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%   ( 5.539,   8.262 )   ( 5.436,   8.117 )   ( 5.726,   8.336 )

Level      Percentile          BCa
95%   ( 5.876,   8.556 )   ( 5.782,   8.435 )
Calculations and Intervals on Original Scale

```

Wir erhalten Konfidenzintervalle, die den asymptotischen Intervallen ähnlich sind. Wir führen die entsprechenden Berechnungen für den CvM-MD-Schätzer durch.

```

1 ## Bootstrap-Schätzungen
2 boot.out <- boot(ITSDaten.nachHause$SAPS.II, statistic = MDEst, R = 999,
3                   parallel = "multicore", ncpus = nr.ncpus)
4 ## Bootstrap-Konfidenzintervall für scale
5 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%   ( 4.458,   7.709 )   ( 4.234,   7.603 )   ( 4.652,   8.163 )

Level      Percentile          BCa
95%   ( 4.462,   7.831 )   ( 4.721,   8.201 )
Calculations and Intervals on Original Scale

```

```

1 ## Bootstrap-Konfidenzintervall für shape
2 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%    ( 4.659,   7.958 )  ( 4.377,   7.821 )  ( 4.894,   8.192 )

Level      Percentile       BCa
95%    ( 5.105,   8.549 )  ( 4.896,   8.139 )
Calculations and Intervals on Original Scale

```

Es ergeben sich erneut Ergebnisse, die nur wenig von den asymptotischen abweichen. Schließlich geben wir noch den R-Code für die Berechnungen der Bootstrap-Konfidenzintervalle für den RMX-Schätzer an. Verzichten aber, wie bereits erwähnt, wegen der sehr hohen Rechenzeit auf die Berechnungen. Wir gehen davon aus, dass wir auch hier Ergebnisse bekommen würden, die sich nicht deutlich von den asymptotischen Ergebnissen unterscheiden würden.

```

1 ## Bootstrap-Schätzungen
2 boot.out <- boot(ITSDaten.nachHause$SAPS.II, statistic = RMXEst, R = 999,
3                   parallel = "multicore", ncpus = nr.ncpus)
4 ## Bootstrap-Konfidenzintervall für scale
5 boot.ci(boot.out, index = c(1,3))
6 ## Bootstrap-Konfidenzintervall für shape
7 boot.ci(boot.out, index = c(2,4))

```

Wir haben demnach drei Modelle gefunden, mit denen sich die vorliegenden Daten gut beschreiben lassen. Welches dieser drei Modelle letztendlich am besten ist, müsste durch das Heranziehen neuer, unabhängiger Daten weiter untersucht werden.

Neben der Absicherung von Punktschätzungen wie im obigen Fall können Konfidenzintervalle auch für die Fallzahlplanung (vgl. Abschnitt 5.1) herangezogen werden. Dies wollen wir abschließend im folgenden Beispiel an einem einfachen Fall demonstrieren.

Beispiel 5.15. Wir wollen uns mit der Frage beschäftigen, wie viele Personen die Meinungsforschungs-institute bei der Sonntagsfrage befragen sollten, um möglichst zuverlässige Prognosen zu erhalten. Aufgrund der großen Population (z.B. Population eines Landes) können wir in diesem Fall die Endlichkeitskorrektur getrost vernachlässigen und können das asymptotische Konfidenzintervall aus Beispiel 5.13 heranziehen. Da uns die Abweichung vom geschätzten Wert interessiert, also der maximale Schätzfehler, müssen wir uns den folgenden Ausdruck genauer ansehen.

$$z_{1-\alpha/2} \sqrt{\frac{\hat{p}(1 - \hat{p})}{n}} \quad (5.33)$$

Offenbar variiert der Schätzfehler in Abhängigkeit von Konfidenzniveau $1 - \alpha$, der geschätzten (hier: erwarteten) Wahrscheinlichkeit \hat{p} und der Stichprobengröße n . Wir gehen von einem 95%-Konfidenzintervall aus; d.h., wir erhalten für $z_{1-\alpha/2} = z_{0.975}$

```
1 qnorm(0.975)
```

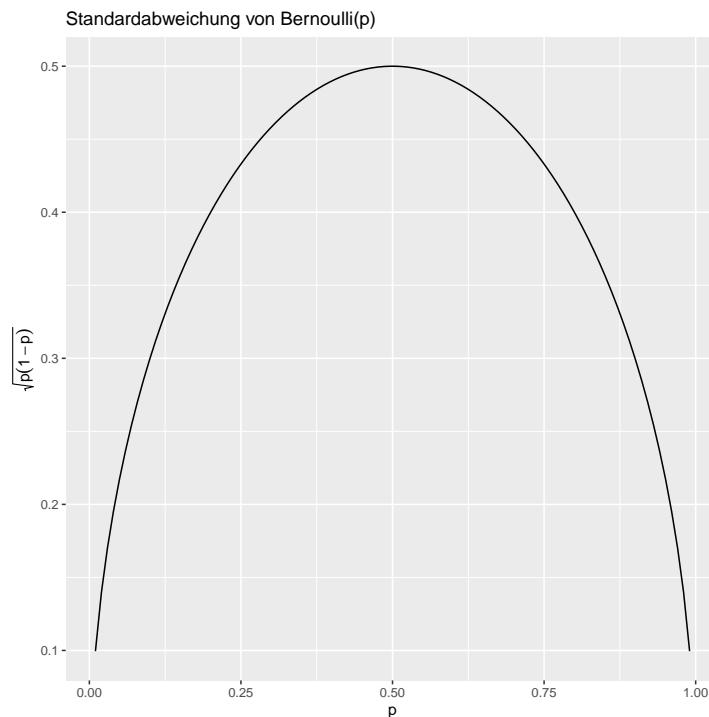
```
[1] 1.959964
```

Wir betrachten die Standardabweichung $\sqrt{p(1-p)}$ der Bernoulli-Verteilung genauer.

```

1 ## Werte für p
2 p <- seq(from = 0.01, to = 0.99, length = 100)
3 ## Standardabweichung
4 SD <- sqrt(p*(1-p))
5 ## Grafische Darstellung
6 DF <- data.frame(p, SD)
7 ggplot(DF, aes(x = p, y = SD)) + geom_line() +
8   ylab(expression(sqrt(p*(1-p)))) +
9   xlab("p") + gtitle("Standardabweichung von Bernoulli(p)")

```



Diesem Plot können wir entnehmen, dass der untersuchte Term für $p = 0.5$ am größten ist und für kleinere oder größere Wahrscheinlichkeiten jeweils abnimmt. Im Fall von $p = 0.5$ erhalten wir demnach den maximal möglichen maximalen Schätzfehler. Es folgt für das 95%-Konfidenzintervall

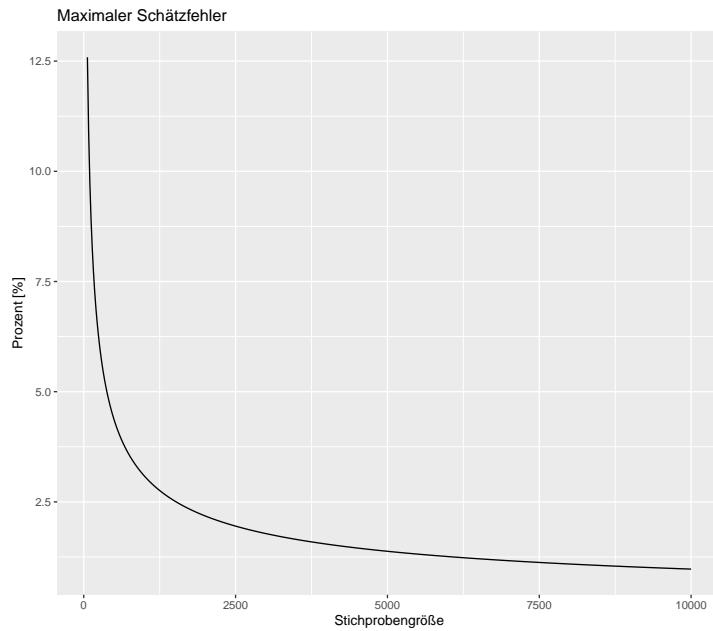
$$1.96 \times \frac{0.5}{\sqrt{n}} = \frac{0.975}{\sqrt{n}} = \frac{97.5}{\sqrt{n}} \% \quad (5.34)$$

Wir plotten diesen maximalen Schätzfehler in Abhängigkeit von der Stichprobengröße.

```

1 ## Stichprobengröße
2 n <- seq(60, 10000, by = 20)
3 ## Maximaler Schätzfehler
4 maxFehler <- 97.5 / sqrt(n)
5 ## Grafische Darstellung
6 DF <- data.frame(n, maxFehler)
7 ggplot(DF, aes(x = n, y = maxFehler)) + geom_line() + ylab("Prozent [%]") +
8   xlab("Stichprobengröße") + gtitle("Maximaler Schätzfehler")

```



Üblicherweise werden bei der Sonntagsfrage 1000 Personen befragt. Der maximale Schätzfehler liegt also im ungünstigsten Fall bei ca. 3.1%. Wir könnten nun eine maximal notwendige Fallzahl berechnen, indem wir den gewünschten maximalen Schätzfehler vorgeben, diesen gleich dem Ausdruck (5.34) setzen und die sich ergebende Gleichung nach n auflösen

$$\sqrt{n} = \frac{0.975}{\text{max. Schätzfehler}} \quad \Leftrightarrow \quad n = \left(\frac{0.975}{\text{max. Schätzfehler}} \right)^2 \quad (5.35)$$

Das Ergebnis runden wir dann noch zur nächsten ganzen Zahl auf und haben damit unsere Fallzahl. Wir können die Fallzahl aber auch direkt mit Hilfe der Funktion `ssize.propCI` aus dem Paket "MKpower" (Kohl (2020c)) berechnen. Es kann dabei auch die Stetigkeitskorrektur berücksichtigt werden sowie die Fallzahl für verschiedene exakte Intervalle berechnet werden. Anstelle des maximalen Schätzfehlers muss bei dieser Funktion die gewünschte Länge des Konfidenzintervalls angegeben werden.

```
1 ## ohne Stetigkeitskorrektur
2 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
```

```
1 ## mit Stetigkeitskorrektur
2 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
```

```
1 ## Clopper-Pearson
2 ssize.propCI(0.5, width = 0.062, method = "clopper-pearsom")
```

```
Sample size calculation by method of clopper-pearsom

n = 1032
prop = 0.5
width = 0.062
conf.level = 0.95

NOTE: Two-sided confidence interval
```

```
1 ## Agresti-Coull
2 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
```

Vor wichtigen Wahlen befragen die Meinungsforscher bis zu 50000 Personen, was in jedem Fall auf einen maximalen Schätzfehler von weniger als 0.5% führt.

```
1 ## ohne Stetigkeitskorrektur
2 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

```
1 ## mit Stetigkeitskorrektur
2 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

Diese Überlegungen und Berechnungen sind nicht nur für Meinungsforscher wichtig, sondern spielen auch in anderen Bereichen eine wichtige Rolle wie etwa in der Epidemiologie oder der medizinischen Statistik. Hier geht es zum Beispiel um die Schätzung der Prävalenzen, Inzidenzen oder Letalitäten von Erkrankungen oder auch der Erfolgsquoten von Behandlungen.

5.4 Übungsaufgaben

Beschreiben und interpretieren jeweils detailliert die Ergebnisse. Verwenden Sie für die Aufgaben 5–18 den ITS-Datensatz und wählen Sie jeweils geeignete Funktionen für die Berechnungen aus.

1. Konstruieren Sie einen Datensatz aus genau fünf positiven Zahlen, bei dem der Median gleich 5 und das arithmetische Mittel gleich 8 ist. Ändern Sie den Datensatz anschließend so ab, dass der Median unverändert bleibt, aber das arithmetische Mittel größer als das dritte Quartil ist.
2. Wie muss ein Datensatz aussehen, damit die Standardabweichung gleich 0 ist? Überlegen Sie sich den Zusammenhang an einfachen Datensätzen.
3. Man kann den Knochenabbau mittels TRAP (tartate resistant acid phosphatase) untersuchen, welches man im Blut messen kann. In einer Studie mit 31 jungen Frauen war das arithmetische Mittel gleich 13.2 U/l (Units pro Liter). Gehen Sie von einem Normalverteilungsmodell für TRAP aus, wobei Sie die Standardabweichung mit $\sigma = 6.5$ U/l als bekannt voraussetzen dürfen. Geben Sie ein 95%-Konfidenzintervall für den Mittelwert μ der Frauen an, die von der Studie repräsentiert werden. Wie verändert sich das Intervall, wenn der Wert für die Standardabweichung nicht bekannt war, sondern mit 6.5 U/l mittels der Stichprobenstandardabweichung (Standardisierung $\frac{1}{n-1}$) geschätzt wurde? D.h., vergleichen Sie die Ergebnisse der Formeln (5.24) und (5.26).
4. Gehen Sie von 6 Erfolgen bei 20 Versuchen aus. Können Sie in diesem Fall das approximative Konfidenzintervall für die Erfolgswahrscheinlichkeit p verwenden? D.h., überprüfen Sie die Faustformel (5.29). Vergleichen Sie anschließend das Clopper-Pearson-Intervall und das asymptotische Konfidenzintervall inklusive Stetigkeitskorrektur.

5. Schätzen Sie die Wahrscheinlichkeit, dass ein ITS-Patient männlich ist. Fügen Sie hierzu eine binäre Variable zum Datensatz hinzu, welche den Wert 1 besitzt, wenn der Patient ein Mann ist; z.B. durch

```
1 ITSDaten$Mann ← as.integer(ITSDaten$Geschlecht == "männlich")
```

Verwenden Sie das Bernoulli-Modell $\mathcal{P} = \{\text{Bernoulli}(p) \mid p \in (0, 1)\}$ und berechnen Sie den ML- und den CvM-MD-Schätzer für p . Bestimmen Sie die zugehörigen asymptotischen Konfidenzintervalle sowie auch Bootstrap-Konfidenzintervalle. Vergleichen Sie die asymptotischen und Bootstrap-Konfidenzintervalle mit dem Clopper-Pearson-Intervall.

6. Gehen Sie davon aus, dass sich die logarithmierten Bilirubin-Werte von ITS-Patienten durch eine Normalverteilung beschreiben lassen.

```
1 ITSDaten$logBili ← log10(ITSDaten$Bilirubin)
```

Berechnen Sie den ML-Schätzer und vergleichen Sie das Ergebnis mit Median und MAD sowie dem RMX-Schätzer (verwenden Sie Funktion `roblox` aus dem Paket "RobLox" (Kohl (2019)) oder die Funktion `rmx` aus dem Paket "rmx" (Kohl (2022c))). Für den RMX-Schätzer gehen Sie von 1 – 5% fehlerhaften Daten aus. Bestimmen Sie auch jeweils die zugehörigen Konfidenzintervalle verwenden Sie hierfür Bootstrap. Stellen Sie die Daten mittels eines Histogramms grafisch dar und ergänzen Sie die drei Normalverteilungsdichten für die geschätzten Parameter. Validieren Sie die drei Modelle zusätzlich mit pp- und qq-Plots.

7. Betrachten Sie nur Patienten mit einer Aufenthaltsdauer von mehr als einem Tag ($\text{LOS} > 1$).

```
1 ITSDaten.LOS2 ← ITSDaten[ITSDaten$LOS > 1, ]
```

Nehmen Sie an, dass man die maximale Körpertemperatur (d.h. Spalte Temperatur) von diesen ITS-Patienten durch eine Normalverteilung beschreiben kann. Berechnen Sie den ML-Schätzer und vergleichen Sie das Ergebnis mit Median und MAD sowie dem RMX-Schätzer (verwenden Sie Funktion `roblox` aus dem Paket "RobLox" (Kohl (2019)) oder die Funktion `rmx` aus dem Paket "rmx" (Kohl (2022c))). Für den RMX-Schätzer gehen Sie von 1 – 5% fehlerhaften Daten aus. Bestimmen Sie auch die entsprechenden Konfidenzintervalle mit Hilfe von Bootstrap. Stellen Sie die Daten graphisch in Form eines Histogramms dar und fügen Sie die drei Normalverteilungsdichten mit den geschätzten Parametern hinzu. Validieren Sie die drei Modelle zusätzlich mit pp- und qq-Plots.

8. Betrachten Sie nur Patienten mit einer Aufenthaltsdauer von genau einem Tag ($\text{LOS} = 1$) und entfernen Sie außerdem den Ausreißer mit der Körpertemperatur von 9.1°C .

```
1 ITSDaten.LOS1 ← ITSDaten[ITSDaten$LOS == 1, ]
2 ITSDaten.LOS1 ← ITSDaten.LOS1[ITSDaten.LOS1$Temperatur > 10, ]
```

Nehmen Sie an, dass man die maximale Körpertemperatur (d.h. Spalte Temperatur) von diesen ITS-Patienten durch eine Normalverteilung beschreiben kann. Berechnen Sie den ML-Schätzer und vergleichen Sie das Ergebnis mit Median und MAD sowie dem RMX-Schätzer (verwenden Sie Funktion `roblox` aus dem Paket "RobLox" (Kohl (2019)) oder die Funktion `rmx` aus dem Paket "rmx" (Kohl (2022c))). Für den RMX-Schätzer gehen Sie von 0 – 5% fehlerhaften Daten aus. Bestimmen Sie die entsprechenden Konfidenzintervalle mit Hilfe von Bootstrap. Stellen Sie die Daten graphisch in Form eines Histogramms dar und fügen Sie die drei Normalverteilungsdichten mit den geschätzten Parametern hinzu. Validieren Sie die drei Modelle zusätzlich mit pp- und qq-Plots.

9. Betrachten Sie nur Patienten mit einer Aufenthaltsdauer von mehr als einem Tag ($LOS > 1$). Nehmen Sie an, dass man die logarithmierten maximalen Herzfrequenzen (d.h. Spalte Herzfrequenz) dieser ITS-Patienten durch eine Normalverteilung beschreiben kann.

```
1 ITSDaten.LOS2 ← ITSDaten[ITSDaten$LOS > 1, ]
2 ITSDaten.LOS2$logHF ← log10(ITSDaten.LOS2$Herzfrequenz)
```

Berechnen Sie den ML-Schätzer und vergleichen Sie das Ergebnis mit Median und MAD sowie dem RMX-Schätzer (verwenden Sie Funktion `roblox` aus dem Paket "RobLox" (Kohl (2019)) oder die Funktion `rmx` aus dem Paket "rmx" (Kohl (2022c))). Für den RMX-Schätzer gehen Sie von 0 – 5% fehlerhaften Daten aus. Bestimmen Sie auch die entsprechenden Konfidenzintervalle mit Hilfe von Bootstrap. Stellen Sie die Daten graphisch in Form eines Histogramms dar und fügen Sie die drei Normalverteilungsdichten mit den geschätzten Parametern hinzu. Validieren Sie die drei Modelle zusätzlich mit pp- und qq-Plots.

10. Betrachten Sie nur Patienten mit einer Aufenthaltsdauer von genau einem Tag ($LOS = 1$). Nehmen Sie an, dass man die logarithmierten maximalen Herzfrequenzen (d.h. Spalte Herzfrequenz) dieser ITS-Patienten durch eine Normalverteilung beschreiben kann.

```
1 ITSDaten.LOS1 ← ITSDaten[ITSDaten$LOS == 1, ]
2 ITSDaten.LOS1$logHF ← log10(ITSDaten.LOS1$Herzfrequenz)
```

Berechnen Sie den ML-Schätzer und vergleichen Sie das Ergebnis mit Median und MAD sowie dem RMX-Schätzer (verwenden Sie Funktion `roblox` aus dem Paket "RobLox" (Kohl (2019)) oder die Funktion `rmx` aus dem Paket "rmx" (Kohl (2022c))). Für den RMX-Schätzer gehen Sie von 0 – 5% fehlerhaften Daten aus. Bestimmen Sie auch die entsprechenden Konfidenzintervalle mit Hilfe von Bootstrap. Stellen Sie die Daten graphisch in Form eines Histogramms dar und fügen Sie die drei Normalverteilungsdichten mit den geschätzten Parametern hinzu. Validieren Sie die drei Modelle zusätzlich mit pp- und qq-Plots.

11. Betrachten Sie nur die Patienten mit Leberversagen. Nehmen Sie an, dass man die logarithmierten Bilirubinwerte (d.h. Spalte Bilirubin) dieser ITS-Patienten durch eine Normalverteilung beschreiben kann.

```
1 ITSDaten.LV ← ITSDaten[ITSDaten$Leberversagen == 1, ]
2 ITSDaten.LV$logBili ← log10(ITSDaten.LV$Bilirubin)
```

Berechnen Sie den ML-Schätzer und vergleichen Sie das Ergebnis mit Median und MAD sowie dem RMX-Schätzer (verwenden Sie Funktion `roblox` aus dem Paket "RobLox" (Kohl (2019)) oder die Funktion `rmx` aus dem Paket "rmx" (Kohl (2022c))). Für den RMX-Schätzer gehen Sie von 0 – 10% fehlerhaften Daten aus. Bestimmen Sie auch die entsprechenden Konfidenzintervalle mit Hilfe von Bootstrap. Stellen Sie die Daten graphisch in Form eines Histogramms dar und fügen Sie die drei Normalverteilungsdichten mit den geschätzten Parametern hinzu. Validieren Sie die drei Modelle zusätzlich mit pp- und qq-Plots.

12. Nehmen Sie an, dass sich die Aufenthaltsdauer (LOS) von ITS-Patienten durch eine Gamma-Verteilung beschreiben lässt. Berechnen Sie den ML-, den CvM-MD- und den RMX-Schätzer (verwenden Sie Funktion `roptest` aus dem Paket "ROptEst" (Kohl and Ruckdeschel (2019)) sowie deren asymptotische Konfidenzintervalle. Für den RMX-Schätzer gehen Sie von 1 – 5% fehlerhaften Daten aus. Die Berechnung des RMX-Schätzers wird einige Zeit in Anspruch nehmen. Stellen Sie die Daten mittels eines Histogramms grafisch dar und ergänzen Sie die drei Gamma-Verteilungsdichten für die geschätzten Parameter. Validieren Sie die drei Modelle zusätzlich mit pp- und qq-Plots.
13. Betrachten Sie nur Patienten mit einer neurologischen Operation (d.h. `OP` = "Neuro").

```
1 ITSDaten.Neuro ← ITSDaten[ITSDaten$OP == "Neuro", ]
```

Betrachten Sie die Aufenthaltsdauer (d.h. Spalte LOS) dieser ITS-Patienten und nehmen Sie an, dass man diese durch eine Gammaverteilung beschreiben kann. Bestimmen Sie den ML-, den CvM-MD- und den RMX-Schätzer (verwenden Sie Funktion `roptest` aus dem Paket "ROptEst" (Kohl and Ruckdeschel (2019)) sowie deren asymptotische Konfidenzintervalle. Für den RMX-Schätzer gehen Sie von 1 – 5% fehlerhaften Daten aus. Die Berechnung des RMX-Schätzers wird einige Zeit in Anspruch nehmen. Stellen Sie die Daten graphisch in Form eines Histogramms dar und fügen Sie die drei Gammaverteilungsdichten mit den geschätzten Parametern hinzu. Validieren Sie die drei Modelle zusätzlich mit pp- und qq-Plots.

14. Wählen Sie nur Patienten mit `OP` = "Magen-Darm-Trakt" – d.h., einer Operation des Magen-Darm-Trakts – aus.

```
1 ITSDaten.MDT ← ITSDaten[ITSDaten$OP == "Magen-Darm-Trakt", ]
```

Betrachten Sie die Aufenthaltsdauer (d.h. Spalte LOS) dieser ITS-Patienten und nehmen Sie an, dass man diese durch eine Gammaverteilung beschreiben kann. Bestimmen Sie den ML-, den CvM-MD- und den RMX-Schätzer (verwenden Sie Funktion `roptest` aus dem Paket "ROptEst" (Kohl and Ruckdeschel (2019)) und deren asymptotische Konfidenzintervalle. Für den RMX-Schätzer gehen Sie von 1 – 5% fehlerhaften Daten aus. Die Berechnung des RMX-Schätzers wird einige

Zeit in Anspruch nehmen. Stellen Sie die Daten graphisch in Form eines Histogramms dar und fügen Sie die drei Gammaverteilungsdichten mit den geschätzten Parametern hinzu. Validieren Sie die drei Modelle zusätzlich mit pp- und qq-Plots.

15. Betrachten Sie nur Patienten mit einer Herz-Thorax-Operation (d.h. OP = "Herz-Thorax").

```
1 ITSDaten.HT <- ITSDaten[ITSDaten$OP == "Herz-Thorax", ]
```

Betrachten Sie die Aufenthaltsdauer (d.h. Spalte LOS) dieser ITS-Patienten und nehmen Sie an, dass man diese durch eine Gammaverteilung beschreiben kann. Bestimmen Sie den ML-, den CvM-MD- und den RMX-Schätzer und deren asymptotische Konfidenzintervalle. Für den RMX-Schätzer (verwenden Sie Funktion roptest aus dem Paket "ROptEst" (Kohl and Ruckdeschel (2019)) gehen Sie von 1 – 5% fehlerhaften Daten aus. Die Berechnung des RMX-Schätzers wird einige Zeit in Anspruch nehmen. Stellen Sie die Daten graphisch in Form eines Histogramms dar und fügen Sie die drei Gammaverteilungsdichten mit den geschätzten Parametern hinzu. Validieren Sie die drei Modelle zusätzlich mit pp- und qq-Plots.

16. Untersuchen Sie das Alter der ITS-Patienten genauer. Nehmen Sie an, Sie können das Alter durch eine Weibullverteilung beschreiben. Bestimmen Sie den ML-, CvM-MD- und den RMX-Schätzer (verwenden Sie Funktion roptest aus dem Paket "ROptEst" (Kohl and Ruckdeschel (2019)) und deren asymptotische Konfidenzintervalle. Sie benötigen hierfür das Paket "RobExtremes" (Ruckdeschel et al. (2019)). Für den RMX-Schätzer gehen Sie von 1 – 5% fehlerhaften Daten aus. Die Berechnung des RMX-Schätzers wird einige Zeit in Anspruch nehmen. Stellen Sie die Daten mittels eines Histogramms grafisch dar und ergänzen Sie die drei Weibull-Verteilungsdichten für die geschätzten Parameter. Validieren Sie die drei Modelle zusätzlich mit pp- und qq-Plots.

17. Betrachten Sie nur Patienten mit einer Aufenthaltsdauer von mehr als einen Tag (LOS > 1).

```
1 ITSDaten.LOS2 <- ITSDaten[ITSDaten$LOS > 1, ]
```

Untersuchen Sie den maximalen SAPS-II Score (d.h. Spalte SAPS.II) dieser ITS-Patienten und nehmen Sie an, dass man diesen durch eine Weibullverteilung beschreiben kann. Bestimmen Sie den ML-, den CvM-MD- und den RMX-Schätzer (verwenden Sie Funktion roptest aus dem Paket "ROptEst" (Kohl and Ruckdeschel (2019)) und deren asymptotische Konfidenzintervalle. Sie benötigen hierfür das Paket "RobExtremes" (Ruckdeschel et al. (2019)). Für den RMX-Schätzer gehen Sie von 1 – 5% fehlerhaften Daten aus. Die Berechnung des RMX-Schätzers wird einige Zeit in Anspruch nehmen. Stellen Sie die Daten graphisch in Form eines Histogramms dar und fügen Sie die drei Weibullverteilungsdichten mit den geschätzten Parametern hinzu. Validieren Sie die drei Modelle zusätzlich mit pp- und qq-Plots.

18. Betrachten Sie nur Patienten mit einer Aufenthaltsdauer von einem Tag (LOS = 1).

```
1 ITSDaten.LOS1 <- ITSDaten[ITSDaten$LOS == 1, ]
```

Untersuchen Sie den maximalen SAPS-II Score (d.h. Spalte SAPS.II) dieser ITS-Patienten und nehmen Sie an, dass man diesen durch eine Weibullverteilung beschreiben kann. Bestimmen Sie den ML-, den CvM-MD- und den RMX-Schätzer (verwenden Sie Funktion `roptest` aus dem Paket "ROptEst" (Kohl and Ruckdeschel (2019)) und deren asymptotische Konfidenzintervalle. Sie benötigen hierfür das Paket "RobExtremes" (Ruckdeschel et al. (2019)). Für den RMX-Schätzer gehen Sie von 1–5% fehlerhaften Daten aus. Die Berechnung des RMX-Schätzers wird einige Zeit in Anspruch nehmen. Stellen Sie die Daten graphisch in Form eines Histogramms dar und fügen Sie die drei Weibullverteilungsdichten mit den geschätzten Parametern hinzu. Validieren Sie die drei Modelle zusätzlich mit pp- und qq-Plots.

19. Laden Sie

<https://github.com/stamats/COVID-19/blob/master/COVID-19.Rmd>

und

<https://github.com/stamats/COVID-19/blob/master/Bevoelkerung2019.RData>

herunter und speichern Sie beide Dateien in einem gemeinsamen Verzeichnis. Erzeugen Sie mit Hilfe von RStudio die html-Datei zur R Markdown Datei und lesen Sie diese.

Achtung: Sie müssen hierfür zusätzliche R Pakete installieren!

Verwenden Sie aktuelle Zahlen für Deutschland und berechnen Sie damit die Dunkelziffer (inkl. 95%-Konfidenzintervall) neu.

20. In der Europäischen Union spricht man von einer seltenen Erkrankung, wenn die Inzidenz weniger als 1 von 2000 Personen beträgt. Die Inzidenzrate pro Jahr von Sarkoidose beträgt ca. 5-60 Fälle pro 100 Tausend Einwohner. Wir gehen davon aus, dass es sich hierbei um ein symmetrisches 95%-Konfidenzintervall für die Inzidenzrate handelt. Der Erwartungswert (Mittelpunkt des Intervalls) beträgt demnach 32.5 Fälle pro 100 Tausend Einwohner. Die Länge des Konfidenzintervalls beträgt 55 Fälle pro 100 Tausend Einwohner. Führen Sie eine Fallzahlplanung durch, mit der Sie bestimmen, wie viele Personen in eine Studie eingeschlossen werden müssten, um dieses asymptotische Konfidenzintervall zu bestätigen. Verwenden Sie die gleichen Werte auch für die Berechnung der Fallzahl mit Hilfe des Clopper-Pearson und des Agresti-Coull Intervalls.

21. Betrachten Sie <https://clinicaltrials.gov/ct2/show/study/NCT03976804>, eine aktuelle Studie zur Prävalenz von Lungenkrebs. Es sollen dort 500 Risikopatienten hinsichtlich der Prävalenz von Lungenkrebs untersucht werden. Bei verschiedenen anderen Prävalenzstudien in Europa wurde eine Prävalenz von ca. 2% ermittelt. Die Verantwortlichen der Studie erwarten, dass die Prävalenz in ihrer Studie mehr als 2% betragen wird. Wie groß muss die Prävalenz in dieser Studie mindestens sein, so dass die untere Grenze des zugehörigen (zweiseitigen) 95% Konfidenzintervalls größer als 2% ist; d.h., man von einem signifikanten Ergebnis und damit einem Erfolg der Studie sprechen kann? Beantworten Sie diese Frage mit Hilfe der Funktion `ssize.propCI` aus dem Paket "MKpower" (Kohl (2020c)). Setzen Sie hierzu Kombinationen aus Prävalenzen und Längen des Konfidenzintervalls ein, so dass die untere Grenze des entsprechenden 95% Konfidenzintervalls größer als 2% ist und sich zugleich eine Fallzahl von grob 500 ergibt.

22. Untersuchen Sie den Zusammenhang zwischen Überleben und dem SAPS-II Score, indem Sie

nach einem optimalen Cut-off für das Überleben suchen. Definieren Sie hierfür zuerst eine neue Variable.

```
1 ITSDaten$Verstorben ← as.integer(ITSDaten$Ergebnis == "Verstorben")
```

Wir gehen davon aus, dass die Wahrscheinlichkeit zu versterben, mit steigendem SAPS-II Score monoton wachsend ist. Verwenden Sie Youdens J Statistik und sichern Sie Ihre Ergebnisse mittels eines stratifizierten Bootstrap ab. Berechnen Sie Sensitivität und Spezifität für den berechneten optimalen Cut-off. Bestimmen Sie außerdem die AUC und den Brier Score.

23. Untersuchen Sie den Zusammenhang zwischen Überleben und der maximalen Herzfrequenz, indem Sie nach einem optimalen Cut-off für das Überleben suchen. Definieren Sie hierfür zuerst eine neue Variable.

```
1 ITSDaten$Verstorben ← as.integer(ITSDaten$Ergebnis == "Verstorben")
```

Wir gehen davon aus, dass die Wahrscheinlichkeit zu versterben, mit steigender maximaler Herzfrequenz monoton wachsend ist. Verwenden Sie Youdens J Statistik und sichern Sie Ihre Ergebnisse mittels eines stratifizierten Bootstrap ab. Berechnen Sie Sensitivität und Spezifität für den berechneten optimalen Cut-off. Bestimmen Sie außerdem die AUC und den Brier Score.

24. Sie wollen die Ausschusswahrscheinlichkeit eines Produktionsprozesses untersuchen und wählen dazu eine repräsentative Stichprobe aus den produzierten Teilen aus. Sie schätzen die unbekannte Ausschusswahrscheinlichkeit und bestimmen das zugehörige 95%-Konfidenzintervall. Dieses Verfahren wiederholen Sie fünf Monate hintereinander, wobei Sie jedes Monat eine neue unabhängige Stichprobe ziehen. Dann ist die Wahrscheinlichkeit, dass alle fünf Intervalle den wahren unbekannten Parameter überdecken kleiner als 95%. Wie groß ist diese Wahrscheinlichkeit genau? Wie wahrscheinlich ist es, dass zumindest vier der fünf Konfidenzintervall den wahren unbekannten Parameter überdecken?

6 Statistische Hypothesentests

In diesem Kapitel werden statistische Hypothesentests besprochen. Im einzelnen geht es um folgende Themen:

- Hypothesen
- Testentscheidungen, Power, Sensitivität, Fehler 1. und 2. Art
- Reihenfolge für die korrekte Durchführung eines Tests
- Fallzahlplanung und Poweranalyse
- 1-Stichproben Binomialtest (exakt und asymptotisch)
- 1-Stichproben Multinomialtest
- 2-Stichproben Binomialtest (asymptotisch)
- Exakter Test von Fisher, χ^2 -Test, Cramér's V Test
- McNemar χ^2 -Test
- Cochran-Mantel-Haenszel χ^2 -Test
- t-Test: 1-Stichproben, gepaart, 2-Stichproben, Welch, Hsu
- Wilcoxon Vorzeichen-Rangtest, Wilcoxon-Mann-Whitney U-Test
- F-Test, Ansari-Bradley Test
- 1-Weg ANOVA, Kruskal-Wallis Test
- 1-Weg ANOVA mit Messwiederholungen, Friedman Test, Quade Test
- Test auf Korrelation (Pearson, Spearman, Kendall)
- Normalverteilungstests: Shapiro-Wilk Test, Lilliefors (Kolmogorov-Smirnov) Test, Cramér-von Mises Test, Shapiro-Francia Test
- Bootstrap- und Permutationstests
- Post hoc Tests

Den R Code für dieses Kapitel befindet sich in der Datei `Testen.Rmd`, welche Sie von meinem GitHub-Account herunterladen können (Link: <https://github.com/stamats/ESDR/blob/master/Testen.Rmd>). Klicken Sie mit der rechten Maustaste auf `Raw`. Dann können Sie das Ziel speichern unter.... Am wenigsten Schwierigkeiten ergeben sich, wenn Sie meine R Markdown Dateien im selben Ordner wie die Daten, welche in den jeweiligen Kapiteln verwendet werden, speichern.

Wir installieren zunächst die in diesem Kapitel benötigten Pakete.

```
1 install.packages(c("coin", "exactRankTests", "ggpubr", "datarium"))
```

Achten Sie darauf, dass Sie die Pakete der vorhergehenden Kapitel 2–5 bereits installiert haben. Wir laden alle Pakete, die wir in diesem Kapitel verwenden werden.

```
1 library(DescTools)
2 library(ggplot2)
3 library(ggsci)
4 library(gridExtra)
5 library(MKinfer)
6 library(coin)
7 library(exactRankTests)
8 library(ggpubr)
9 library(datarium)
```

Wie bereits in Abschnitt 2.4 erklärt, ist ein wiederholtes Ausführen von `library` unproblematisch.

6.1 Einführung

Empirische Untersuchungen und Studien beginnen in der Regel mit einer neuen Idee, einer Vermutung zu einem bestimmten oftmals noch offenen Problem. Diese Vermutung wird in der Regel auf der Basis empirischer Beobachtungen und/oder fachlich-theoretischer Überlegungen aufgestellt. Es erleichtert die Überprüfung einer Vermutung, wenn diese präzise und quantifizierbar ausformuliert werden kann. Man spricht dann auch von einer **Hypothese**. Zunächst sollte man dafür alle vorhandenen Informationen zum Problem sammeln und den theoretischen Hintergrund erarbeiten, um die generelle Plausibilität der Hypothese zu überprüfen. Häufig stellt man dabei bereits fest, dass die Hypothese nicht zutreffen kann, was weitere Arbeit (und Geld) spart.

In vielen Bereichen sind direkte Beweise von Hypothesen nicht möglich und sie lassen sich auch nicht durch ein einmaliges Experiment direkt verifizieren. An dieser Stelle kommt die Statistik ins Spiel. Man sammelt repräsentative und relevante Daten für das Problem und unterzieht diese einer statistischen Analyse, wobei die Ergebnisse durch sogenannte **statistische (Hypothesen-)Tests** abgesichert werden können. Die generelle Vorgehensweise ist im folgenden Beispiel mit Hilfe eines Würfelspiels beschrieben.

Beispiel 6.1. Wir betrachten ein Würfelspiel, bei dem es wichtig ist, “6” zu würfeln. Nach einer gewissen Zeit des Spielens stellen wir fest, dass wir mit unserem Würfel recht selten eine “6” würfeln. Wir vermuten daher, dass

die Häufigkeit von “6” zu klein ist

oder allgemeiner, dass

die Häufigkeit von “6” nicht stimmt.

Insbesondere bedeutet dies, dass nicht alle Seiten des Würfels mit gleicher Wahrscheinlichkeit auftreten; d.h., der Würfel ist nicht fair. Die präzise und quantifizierbare Ausformulierung unserer Vermutung führt uns zur Hypothese:

Die Wahrscheinlichkeit p von “6” ist nicht gleich $\frac{1}{6}$; kurz: $p \neq \frac{1}{6}$

Eine Antwort mit statistischen Tests ist generell nur möglich, wenn man es mit sich gegenseitig ausschließenden Fällen zu tun. Für den Würfel gilt entweder

unsere Hypothese ist richtig; d.h., $p \neq \frac{1}{6}$

oder

unsere Hypothese ist falsch; d.h., $p = \frac{1}{6}$

Im vorliegenden Fall sammelt man Informationen (Evidenz) für die Hypothese durch n -faches Würfeln und Zählen von “6”. Die offenen Fragen, die wir mit Hilfe statistischer Tests beantworten können, sind:

1. Wie oft sollte man würfeln?
2. Bei wie vielen “6” entscheidet man sich für bzw. gegen die Hypothese?

Wie wir im Beispiel gesehen haben, bilden den Ausgangspunkt für statistische Tests zwei sich gegenseitig ausschließende Hypothesen. Diese werden üblicherweise folgendermaßen bezeichnet:

Nullhypothese H_0 : Hypothese, die *widerlegt* werden soll.

Alternative (Hypothese) H_1 : Hypothese, die *bestätigt* werden soll (Forschungshypothese).

Wir übertragen diese Sprechweise auf unser Würfelbeispiel.

Beispiel 6.2. Wir betrachten wieder das Würfelspiel, bei dem es wichtig ist, “6” zu würfeln. In diesem Fall erhalten wir:

Nullhypothese $H_0: p = \frac{1}{6}$ versus Alternative $H_1: p \neq \frac{1}{6}$

Die Alternative enthält die Fälle $p < \frac{1}{6}$ und $p > \frac{1}{6}$, weshalb man auch von einer **zweiseitigen** Hypothese spricht. Denkbar wären auch die **einseitigen** Fälle:

- $H_0: p = \frac{1}{6}$ versus $H_1: p < \frac{1}{6}$
- $H_0: p \geq \frac{1}{6}$ versus $H_1: p < \frac{1}{6}$

Anmerkung:

Die Entscheidung, ob man eine einseitige oder zweiseitige Alternative betrachtet, muss immer vor der Testdurchführung festgelegt werden. In der Medizin ergeben sich zum Beispiel nur sehr selten einseitige Alternativen. Zwar interessiert oft nur eine Verbesserung, jedoch hätte eine Verschlechterung weitreichende Konsequenzen, weshalb aus ethischen Gründen und zur Sicherheit der Patienten zweiseitige Alternativen gewählt werden müssen.

Im Rahmen der induktiven Statistik gehen wir von (repräsentativen) Stichproben aus einer größeren Population aus. Alle Größen, die wir berechnen, hängen daher von der konkreten Stichprobe ab und unterliegen unkontrollierbaren zufälligen Schwankungen. Im Hinblick auf Entscheidungen auf der Basis statistischer Ergebnisse führt dies dazu, dass wir Fehlentscheidungen nie völlig ausschließen können. Es ist unvermeidbar, dass wir uns mit einer (hoffentlich kleinen) positiven Wahrscheinlichkeit falsch entscheiden. Überträgt man dies auf das statistische Testen, ergibt sich die in Tabelle 6.1 dargestellte Situation.

	H_0 ist richtig	H_1 ist richtig
Entscheidung für H_0	Richtige Entscheidung $1 - \alpha$ (Sensitivität)	Fehler 2. Art β
Entscheidung für H_1	Fehler 1. Art α (Signifikanzniveau)	Richtige Entscheidung $1 - \beta$ (Power, Spezifität)

Tabelle 6.1: Entscheidungssituation beim statistischen Testen.

Die möglichen Fehlentscheidungen sind folglich:

Fehler 1. Art: Die Wahrscheinlichkeit, dass H_0 abgelehnt wird, obwohl sie zutrifft.

Fehler 2. Art: Die Wahrscheinlichkeit, dass H_0 nicht abgelehnt wird, obwohl sie falsch ist.

In praktischen Anwendungen wird üblicherweise der Fehler 1. Art als der schwerwiegende Fehler angesehen. Wir werden noch sehen, dass dieser Fehler beim Testen entsprechend strenger kontrolliert wird. Wir verdeutlichen die Fehler mit Hilfe eines Beispiels aus der Medizin.

Beispiel 6.3. Wir betrachten die folgende Situation: Es gibt eine wirksame und sichere Therapie, die seit vielen Jahren im Einsatz ist – ein sogenannter **Goldstandard**. Nun ist man davon überzeugt, dass man einen neuen Therapieansatz gefunden hat, der sogar noch besser funktioniert.

Ein Fehler 1. Art wäre in diesem Fall, dass man sich auf der Basis von vorliegenden Daten gegen den Goldstandard und für den neuen Ansatz entscheidet, obwohl der neue Ansatz gar nicht besser bzw. vielleicht sogar schlechter ist. Das würde bedeuten, dass man den Patienten eine wirksamere Therapie vorenthält und zusätzlich den Patienten, bei denen die neue Therapie unerwünschte Nebenwirkungen zeigt, sogar einen Schaden zufügt.

Im Unterschied dazu, wäre ein Fehler 2. Art, dass man den Goldstandard behält, obwohl der neue Ansatz tatsächlich besser wäre. Damit hat man natürlich, eine Chance für eine Verbesserung verpasst. Die Patienten erhalten aber weiterhin eine wirksame und sichere Therapie.

In diesem Fall wäre also der Fehler 1. Art die schwerwiegendere Fehlentscheidung.

Wir fassen kurz die wesentlichen Punkte zu den beiden Fehlern zusammen, wobei wir mit dem Fehler 1. Art beginnen.

Fehler 1. Art:

- Ist nicht vermeidbar, aber kontrollierbar.
- Die Fehlerwahrscheinlichkeit α muss *vor* der Testdurchführung festgelegt werden!
- α bildet die Basis für die Berechnung des Annahme- bzw. des Ablehnungsbereiches für H_0 .
- Im Prinzip kann α beliebig gewählt werden. Der Standard ist $\alpha = 0.05$, manchmal wird auch $\alpha = 0.01$ oder kleiner verwendet, sehr (sehr) selten $\alpha > 0.05$.
- In Abhängigkeit von α wird im Zusammenhang mit der Annahme von H_1 auch von statistisch signifikant ($\alpha = 0.05$), statistisch hoch signifikant ($\alpha = 0.01$) oder statistisch höchst signifikant ($\alpha = 0.001$) gesprochen.

Fehler 2. Art:

- Ist nur schwer zu bestimmen/schätzen.
- Generell gilt: Je größer α ist, umso kleiner ist β ; d.h., ein kleines α und ein kleines β sind zwei konkurrierende Ziele.
- Außerdem gilt: Je größer die Fallzahl n ist, umso kleiner ist β . Dies ist in der Praxis der einzige Weg, um β zu kontrollieren und führt zur Notwendigkeit einer detaillierten Fallzahlplanung und Poweranalyse.
- Für Fallzahlplanungen benötigt man jedoch ein gewisses Vorwissen über die Größe des Effekts, die Variabilität der verwendeten Schätzer, den Fehler 1. Art, das Testverfahren und die angestrebte Power.
- Standardannahmen für Fallzahlplanungen sind $\alpha = 0.05, 0.01$ und $1 - \beta = 0.8, 0.9$.

Die folgende Aufstellung gibt die Schritte wieder, die für die Durchführung eines statistischen Tests notwendig sind. In dieser strikten Reihenfolge muss etwa im Rahmen von klinischen Studien vorgegangen werden, denn damit ist sichergestellt, dass niemand nach Beginn der Studie noch Einfluss auf das Ergebnis des Tests nehmen kann.

1. Formulierung der Hypothesen H_0 und H_1 (ein-/zweiseitig?)
2. Festlegung des α -Fehlers (Signifikanzniveau)
3. Auswahl eines geeigneten Tests T , üblicherweise der Test mit der größten Power
4. Fallzahlplanung und Poweranalyse (Festlegung von β -Fehler bzw. Power $1 - \beta$, des erwarteten Effekts, der erwarteten Varianz, etc.)

5. Ermittlung von Ablehnungs- (K_α) und Annahmebereich (\bar{K}_α) von H_0
6. Durchführung des Experimentes und Erzeugung relevanter Daten x_1, \dots, x_n
7. Berechnung der Teststatistik $t = T(x_1, \dots, x_n)$
8. Entscheidung für H_1 ($t \in K_\alpha$) oder H_0 ($t \in \bar{K}_\alpha$)

In der Praxis fällt der Schritt 5 üblicherweise weg und die Testentscheidung wird auf Basis des sogenannten **p-Wertes** getroffen. Darunter versteht man die folgende (bedingte) Wahrscheinlichkeit

1-seitiger Test: $p = P(T \geq t | H_0)$ bzw. $p = P(T \leq t | H_0)$

2-seitiger Test: $p = 2 \min\{P(T \geq t | H_0), P(T \leq t | H_0)\}$, bei Symmetrie zu 0: $p = P(|T| \geq |t| | H_0)$

Es wird demnach die Wahrscheinlichkeit berechnet, dass der Wert der Teststatistik T extremere Werte annimmt als die beobachtete Teststatistik t unter der Annahme, dass H_0 richtig ist. Ist p klein, so ist es demnach unwahrscheinlich, dass die vorliegenden Daten unter der Nullhypothese zustande gekommen sind und man entscheidet sich entsprechend für die Alternative. Genauer entscheidet man folgendermaßen:

Falls $p \leq \alpha$: Ablehnen von H_0

Falls $p > \alpha$: Beibehalten von H_0 ; d.h. Ablehnen von H_1

Bemerkung 6.4. (a) Der p-Wert ist nicht die Wahrscheinlichkeit von H_0 . Diese Wahrscheinlichkeit gibt es nicht, denn H_0 ist entweder richtig oder falsch. Außerdem ist p auch nicht der Fehler 1. Art; d.h., die Wahrscheinlichkeit die Nullhypothese H_0 abzulehnen, obwohl diese korrekt ist.

(b) Es ist es falsch, in Abhängigkeit des p-Wertes von statistisch hoch oder höchst signifikant zu sprechen. Die Stärke der Signifikanz hängt nicht direkt von der Größe des p-Wertes ab, sondern davon, ob der p-Wert eine gewisse vorgegebene Signifikanzschranke α unterschreitet. Wurde zum Beispiel für den Test ein $\alpha = 5\%$ festgelegt, so sollte man auch bei einem $p < 0.01$ oder $p < 0.001$ "nur" von einem signifikanten Ergebnis und nicht etwa einem hoch oder höchst signifikanten Ergebnis sprechen.

(c) Von ganz entscheidender Bedeutung ist es auch, zu erkennen, dass statistische Signifikanz nicht gleichbedeutend mit Relevanz ist. Insbesondere bei sehr großen Stichproben können auch kleinste Unterschiede signifikant sein, ohne dass sich daraus irgendwelche Konsequenzen ableiten lassen. Es ist folglich wichtig, neben der Signifikanz auch immer die Größe des beobachteten Effekts und dessen Varianz im Auge zu behalten. Hierfür eignen sich Konfidenzintervalle sehr gut.

Im folgenden Beispiel zeigen wir anhand des **2-Stichproben t-Tests**, der nach dem Pseudonym seines Erfinders auch **Student t-Test** genannt wird, wie ein statistischer Test in der Praxis durchzuführen ist.

Beispiel 6.5. Gegeben seien eine (wohl definierte) Population innerhalb der wir zwei (wohl charakterisierte und disjunkte) Gruppen vergleichen wollen. Wir erwarten einen Unterschied zwischen den beiden Gruppen, wobei uns als Effekt die Differenz der beiden Erwartungswerte (Lageparameter) für ein bestimmtes Merkmal X interessiert. Wir nehmen an, dass dieses Merkmal in beiden Gruppen normalverteilt ist (zumindest näherungsweise) und für die beiden Gruppen die selbe unbekannte Varianz vorliegt; d.h., es gilt:

Gruppe 1: $X_1 \sim \text{Norm}(\mu_1, \sigma^2)$ versus Gruppe 2: $X_2 \sim \text{Norm}(\mu_2, \sigma^2)$

Für die Fallzahlplanung und Poweranalyse ist dies aber nicht ausreichend, sondern wir müssen die beiden Verteilungen genauer spezifizieren. Wir erwarten

Gruppe 1: $X_1 \sim \text{Norm}(0.5, 1)$ versus Gruppe 2: $X_2 \sim \text{Norm}(1.5, 1)$

Um die Erwartungswerte μ_1 und μ_2 der beiden Gruppen mittels eines statistischen Tests miteinander zu vergleichen bzw. das Vorliegen eines Unterschiedes $\mu_1 - \mu_2$ zu ermitteln, führen wir die oben beschriebenen Schritte 1-8 durch:

1. Wir betrachten die folgenden Hypothesen

$$H_0: \mu_1 = \mu_2 \quad \text{versus} \quad H_1: \mu_1 \neq \mu_2$$

was äquivalent zu

$$H_0: \mu_1 - \mu_2 = 0 \quad \text{versus} \quad H_1: \mu_1 - \mu_2 \neq 0$$

ist. Die Alternative ist in diesem Fall also zweiseitig.

2. Standardmäßig wählen wir als Fehler 1. Art (Signifikanzniveau): $\alpha = 0.05$
3. Da wir von einer Normalverteilung in beiden Gruppen ausgehen und die Differenz der Mittelwerte schätzen wollen, wobei auch die unbekannte Varianz mitgeschätzt werden muss, führt uns dies auf eine t-Verteilung. Entsprechend wählen wir als Test den 2-Stichproben t-Test, der für diese Situation die größtmögliche Power hat. Angenommen x_1, \dots, x_{n_1} seien die Beobachtungen für Gruppe 1 und y_1, \dots, y_{n_2} seien die Beobachtungen für Gruppe 2, dann erhalten wir als Teststatistik

$$T(x_1, \dots, x_{n_1}; y_1, \dots, y_{n_2}) = \sqrt{\frac{n_1 n_2}{n_1 + n_2} \frac{\text{AM}(x_1, \dots, x_{n_1}) - \text{AM}(y_1, \dots, y_{n_2})}{\text{SD}(x_1, \dots, x_{n_1}; y_1, \dots, y_{n_2})}} \quad (6.1)$$

wobei

$$\text{SD}(x_1, \dots, x_{n_1}; y_1, \dots, y_{n_2}) = \sqrt{\frac{(n_1 - 1)\tilde{S}(x_1, \dots, x_{n_1}) + (n_2 - 1)\tilde{S}(y_1, \dots, y_{n_2})}{n_1 + n_2 - 2}} \quad (6.2)$$

und \tilde{S} die Stichprobenvarianz mit Standardisierung $\frac{1}{n-1}$ ist.

4. Für die Fallzahlplanung und die Poweranalyse benötigen wir zusätzlich die (erwartete) Effektstärke $\delta = |\mu_1 - \mu_2|$, die (erwartete) Standardabweichung σ und die gewünschte Power $1 - \beta$. Wir erwarten $\delta = 1$ und $\sigma = 1$ und wählen $1 - \beta = 0.8$.

Die Auswirkung der Effektstärke auf die Fallzahl ist in Abbildung 6.1 dargestellt, wobei wir den Standardfall $\beta = 0.2$ betrachten und ohne Einschränkung $\sigma = 1$ angenommen haben. Für die Berechnungen wurde die Funktion `power.t.test` verwendet. Wir sehen, dass die Fallzahl mit zunehmender Effektstärke deutlich abnimmt; d.h., je größer der Effekt ist, desto kleiner ist die benötigte Fallzahl bzw. umgekehrt kann man mit sehr großen Stichproben auch kleinste (irrelevante)

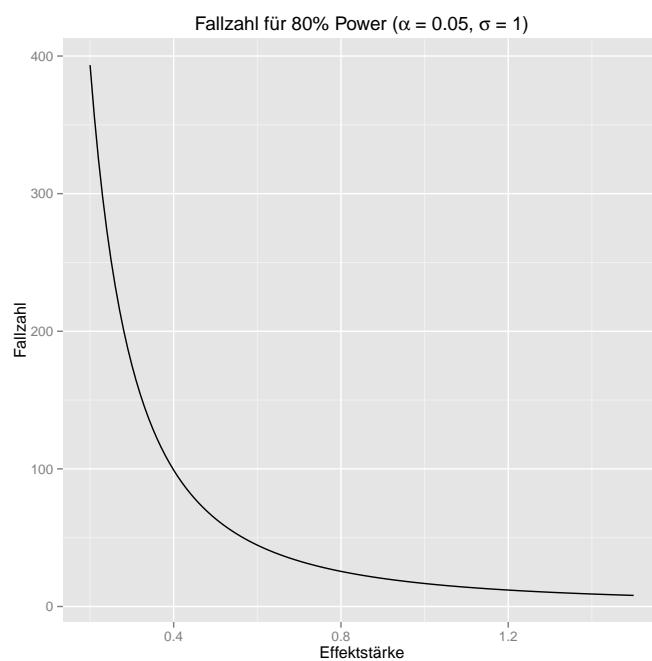


Abbildung 6.1: Fallzahl in Abhängigkeit von der Effektstärke.

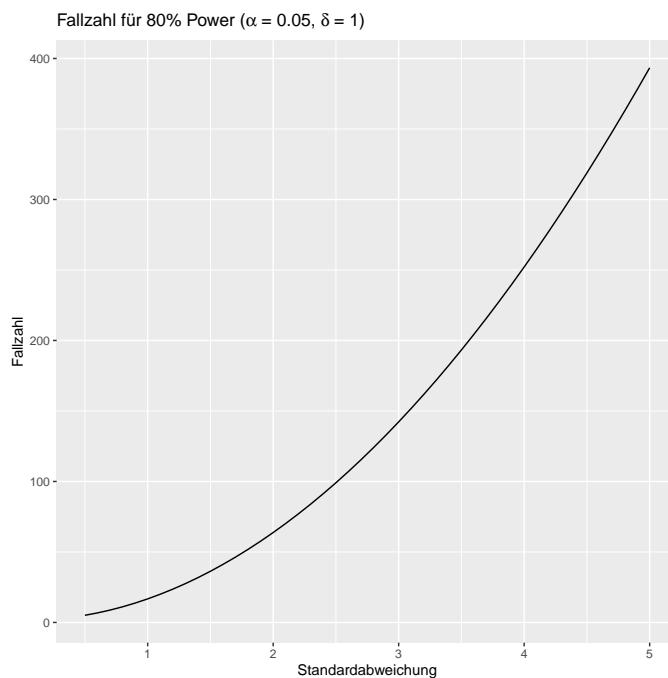


Abbildung 6.2: Fallzahl in Abhängigkeit von der Standardabweichung.

Effekte nachweisen.

Die Abbildung 6.2 zeigt die Abhängigkeit der Fallzahl von der Standardabweichung, wobei wir hier ohne Einschränkung eine Effektstärke von 1 angenommen haben. Die Berechnungen wurden wieder mit Hilfe der Funktion `power.t.test` durchgeführt. Je größer demnach die Varianz ist,

umso größer muss auch die Fallzahl gewählt werden. Insbesondere kann man sagen, dass für die Fallzahlplanung das (erwartete) Verhältnis $\frac{\delta}{\sigma}$ ausschlaggebend ist. Man nennt dies auch den (erwarteten) **standardisierten Effekt**. Andere oft verwendete Bezeichnungen dafür sind **Cohens d** oder auch **SMD** (standardized mean difference).

Wir führen für die vorliegende Situation die Fallzahlberechnung mit $\delta = 1$, $\sigma = 1$, $\alpha = 0.05$ und $1 - \beta = 0.8$ durch

```
1 power.t.test(delta = 1, sd = 1, sig.level = 0.05, power = 0.8)
```

```
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
```

Unter diesen Annahmen ist eine Fallzahl von 17 pro Gruppe ausreichend, um den erwarteten Unterschied mit einer Power von 80% und einem Fehler 1. Art von 5% nachzuweisen. In Studien wird die Fallzahl häufig noch etwas nach oben korrigiert, um eventuelle Ausfälle kompensieren zu können. Wir verwenden im Folgenden eine Fallzahl von $n = 20$. Bei vollständigen Daten führt dies auf eine Power von knapp 87%, wie die folgende Berechnung mit `power.t.test` zeigt.

```
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. Unter der Annahme, dass die Nullhypothese H_0 zutrifft, besitzt die Teststatistik T eine t-Verteilung mit $n_1 + n_2 - 2$ Freiheitsgraden. Dies können wir verwenden, um den Annahmebereich \bar{K}_α von H_0 zu bestimmen. Aufgrund der Symmetrie der Situation erhalten wir $\bar{K}_\alpha = [-c, c]$, wobei gelten muss

$$P(-c \leq T \leq c | H_0) = 1 - \alpha \quad (6.3)$$

d.h., c ist das $(1 - \alpha/2)$ -Quantil der $t_{n_1+n_2-2}$ -Verteilung. Der Wert c wird auch der **kritische Wert** des Tests genannt. Unter der Annahme, dass $n_1 = n_2 = 20$ erhalten wir im vorliegenden Fall als kritischen Wert

```
1 qt(0.975, df = 38)
```

```
[1] 2.024394
```

6. Wir führen das Experiment durch, indem wir mit Hilfe der Funktion `rnorm` Zufallszahlen erzeugen. Genauer verwenden wir $X_1 \sim \text{Norm}(0.5, 1)$ für die Gruppe 1 und $X_2 \sim \text{Norm}(1.5, 1)$ für die Gruppe 2.

```
1 ## Zufallszahlen zur Demonstration
2 X1 <- rnorm(n = 20, mean = 0.5, sd = 1)
3 X2 <- rnorm(n = 20, mean = 1.5, sd = 1)
```

7. Wir berechnen die Teststatistik mit Hilfe der Funktion `t.test`.

```
1 t.test(X1, X2, var.equal = TRUE)$statistic
```

```
t
-2.822995
```

Beim zweiseitigen Testen ist aufgrund der Symmetrie der t-Verteilung der Absolutbetrag dieses Wertes mit dem kritischen Wert c zu vergleichen. Ist er kleiner, so entscheidet man sich für H_0 ansonsten für H_1 . Im vorliegenden Fall entscheiden wir uns demnach für H_1 und wir konnten somit unsere Hypothese H_1 bestätigen.

8. Wir können alternativ auch den p-Wert berechnen; d.h., die Wahrscheinlichkeit, dass der oben berechnete oder ein extremerer Wert der Teststatistik auftritt unter der Annahme, dass H_0 zutrifft. Dieser extreme Wert kann beim zweiseitigen Test entweder noch unten oder nach auftreten. Aufgrund der Symmetrie der t-Verteilung genügt es aber, eine der beiden Wahrscheinlichkeiten zu berechnen und diese zu verdoppeln.

```
1 2*pt(abs(t.test(X1, X2, var.equal = TRUE)$statistic), df = 38,
2     lower.tail = FALSE)
```

```
t
0.00753125
```

Mit Hilfe der Funktion `t.test` erhalten wir etwas einfacher das gleiche Ergebnis

```
1 t.test(X1, X2, var.equal = TRUE)$p.value
```

```
[1] 0.00753125
```

Der p-Wert ist folglich kleiner als das vorgegebene Signifikanzniveau 0.05 und wir können von einem signifikanten Ergebnis sprechen. Der p-Wert ist sogar kleiner als 0.01. Es wäre nicht korrekt, in diesem Fall von einem hoch signifikanten Ergebnis zu sprechen, da wir mit $\alpha = 0.05$ gearbeitet haben (vgl. auch Bemerkung 6.4 (b)). Leider geschieht dies jedoch in vielen wissenschaftlichen Publikationen.

Die vollständige Ausgabe der Funktion `t.test` lautet

```
1 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
```

Das angezeigte 95%-Konfidenzintervall ist ein Intervall für die Differenz $\mu_1 - \mu_2$ und repräsentiert somit den zu erwartenden Effekt. Es kann auch für die Testentscheidung herangezogen werden. Im vorliegenden Fall wird die Nullhypothese durch $\mu_1 - \mu_2 = 0$ repräsentiert. Da der Referenzwert 0 nicht im angegebenen 95%-Konfidenzintervall liegt, ist die Nullhypothese zum Signifikanzniveau von 5% abzulehnen. Allgemein gesprochen gilt: Liegt der entsprechende Referenzwert der Nullhypothese nicht im $(1 - \alpha)$ -Konfidenzintervall, kann die Nullhypothese zum Signifikanzniveau α abgelehnt werden.

Da das Intervall den zu erwartenden Effekt und dessen Varianz widerspiegelt, leistet es aber noch mehr als der p-Wert und gibt uns auch einen Eindruck von der Relevanz und Variabilität der Ergebnisse. Im besten Fall wäre auf Basis der vorliegenden Daten sogar ein Effekt von bis zu ca. -1.38 möglich. Aber auch ein Effekt von nur ca. -0.23 wäre auf Basis der vorliegenden Daten nicht auszuschließen. Dieser Aspekt wird im Fall, dass sich kein signifikanter Unterschied ergibt, in der Übungsaufgabe 1 in Abschnitt 6.4 nochmals aufgegriffen.

Anmerkung:

Der sogenannte Publikationsbias, der besagt, dass positive Ergebnisse deutlich häufiger publiziert werden als negative (Dickersin et al. (1987), Song et al. (2010)), hat deutlich zu einem “Jagen” nach Signifikanz (“p-hacking”) beigetragen. Dies hat dazu geführt, dass man heute leider davon ausgehen muss, dass viele, wenn nicht sogar die meisten publizierten Ergebnisse falsch positiv sind (Ioannidis (2005), Colquhoun (2014)). Seit einigen Jahren wird dieses “Jagen” nach Signifikanz in der Forschung

aber richtiger Weise als sehr problematisch angesehen, weshalb man sich vermehrt mit den Stärken und Schwächen des statistischen Testens auseinandersetzt (Cohen (1994), Sterne and Davey Smith (2001), Head et al. (2015), Wasserstein and Lazar (2016), Amrhein et al. (2017)). Außerdem haben dadurch Konfidenzintervalle berechtigterweise an Bedeutung gewonnen (Rigby (1999), du Prel et al. (2009), Ranstam (2012), Sedgwick (2013)).

Für die Fallzahlplanung und Poweranalyse gibt es in R eine Reihe von Funktionen, die meist mit `power.` beginnen. Neben `power.t.test` sind es im Grundpaket "stats" (R Core Team (2022a)) die Funktionen `power.prop.test` und `power.anova.test`. Darauf hinaus gibt es eine Reihe von Erweiterungspaketen, die Funktionen zur Fallzahlplanung und Poweranalyse für verschiedenste Tests und Modelle zur Verfügung stellen.

Anmerkung:

Es ist übliche Praxis, die Voraussetzungen für statistische Tests in Vortests zu untersuchen. Dazu zählt die Überprüfung von Verteilungsannahmen, insbesondere die Normalverteilungsannahme, oder auch der Voraussetzung gleicher Varianzen (Varianzhomogenität). Neben den methodischen Problemen, dass damit mehrere voneinander abhängige Fragestellungen am gleichen Datensatz überprüft werden und man dabei die Reihenfolge der Schritte nicht einhält, die notwendig ist, um den Fehler 1. Art unter Kontrolle zu behalten, haben solche Vortests oft auch eine geringe Power als der Haupttest, der die wissenschaftliche Fragestellung widerspiegelt und die Basis für die Fallzahlplanung war. Damit werden bei kleinen Fallzahlen Abweichungen nur mit geringer Wahrscheinlichkeit erkannt (viele falsch negative Ergebnisse). Andererseits werden durch die Vortests bei großen Fallzahlen kleine, für den Haupttest oftmals irrelevante Abweichungen, aufgedeckt. Rasch et al. (2011) zeigen am Beispiel des t-Tests, dass sich die Praxis der Vortests nicht auszahlt. Zum einen führt dies zu unbekannten Fehlern 1. und 2. Art. Zum anderen ließ sich in ihrer Simulationsstudie weder ein Zugewinn an (empirischer) Power noch eine Reduktion des (empirischen) Fehlers 1. Art feststellen. Sie empfehlen daher, keinerlei Vortests durchzuführen und anstelle des Student t-Tests uneingeschränkt den Welch t-Test zu verwenden. Auch sollte aus ihrer Sicht der Wilcoxon-Mann-Whitney Test nur bei ordinalen Daten zum Einsatz kommen. Diese Tests werden in Abschnitt 6.3 genauer behandelt.

In den folgenden beiden Abschnitten werde ich eine Reihe wichtiger statistischer Tests kurz vorstellen.

6.2 Nominale Merkmale

In diesem Abschnitt werden in erster Linie statistische Tests für nominale Merkmale behandelt. Diese lassen sich natürlich auch für ordinale Merkmale verwenden, indem man die Information der Anordnung ignoriert oder im Fall von stetigen Merkmalen, indem man diese diskretisiert/kategorisiert. Es werden in diesem Abschnitt die in der Abbildung 6.3 dargestellten Tests genauer besprochen.

Im Folgenden werden wir uns Schritt für Schritt durch das Diagramm arbeiten. Wir beginnen mit dem einfachsten Modell nämlich dem Bernoulli-Modell, welches im Fall binärer Variablen Anwendung findet, und wollen statistische Tests für die Untersuchung der Erfolgswahrscheinlichkeit durchführen. Wir gehen zunächst davon aus, dass nur Daten von einer einzigen Stichprobe (Gruppe) vorliegen und wir die

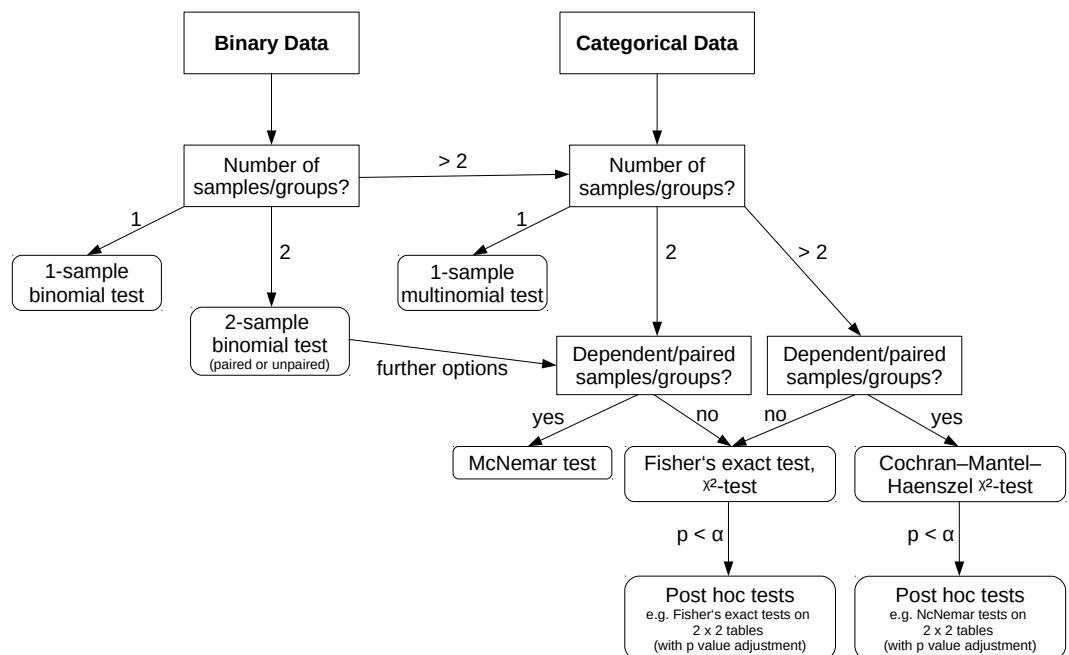


Abbildung 6.3: Baumdiagramm zur Auswahl des geeigneten Tests im Fall nominaler Merkmale.

Wahrscheinlichkeit für das Vorliegen von 1 (Erfolg) untersuchen wollen. Für den Vergleich der relativen Häufigkeit mit einem vorgegeben Wert, können wir den **1-Stichproben Binomialtest** verwenden, der in der Funktion `binom.test` implementiert ist. Wie im Fall der Konfidenzintervalle (vgl. Beispiel 5.13) können wir auch eine Approximation mit Hilfe der Normalverteilung (mit und ohne Stetigkeitskorrektur) verwenden. Diese steht in Form der Funktion `prop.test` zur Verfügung. Wir verwenden wieder den Datensatz der ITS-Patienten und untersuchen die Prävalenz für Leberversagen (prev). Wir möchten untersuchen, ob wir von einer Prävalenz von weniger als 5% ausgehen können; d.h.,

$$H_0 : \text{prev} \geq 0.05 \quad \text{versus} \quad H_1 : \text{prev} < 0.05$$

Wir gehen wir immer in diesem Abschnitt von einem Signifikanzniveau von $\alpha = 0.05$ aus und verwenden sowohl den exakten als auch den asymptotischen Test, also die Funktionen `binom.test` und `prop.test`. Die Alternative spezifizieren wir mittels `alternative = "less"`.

```

1 ITSDaten <- read.csv2(file = "ITSDaten.csv", fileEncoding = "utf8",
2                         stringsAsFactors = TRUE)
3 table(ITSDaten$Leberversagen)
  
```

0	1
480	20

```

1 ## exakter Test
2 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

```

```

1 ## asymptotischer Test mit Stetigkeitskorrektur
2 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

```

```

1 ## asymptotischer Test ohne Stetigkeitskorrektur
2 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

```

Achtung, die Angabe $p = 0.05$ im obigen Code bezieht sich auf die Prävalenz und nicht auf das Signifikanzniveau bzw. den p-Wert.

In allen drei Fällen bekommen wir sehr ähnliche Ergebnisse, was primär an der hohen Fallzahl liegt, wobei die Übereinstimmung zwischen dem exakten und dem asymptotischen Test durch die Stetigkeitskorrektur deutlich besser ist. Wir sehen außerdem, dass wir durch das einseitige Testen auch entsprechende einseitige Konfidenzintervalle erhalten. In allen drei Fällen ist der p-Wert größer als 0.05, weshalb wir

demnach die Nullhypothese beibehalten müssen; d.h., die Prävalenz von Leberversagen (prev) könnte auch größer oder gleich 5% sein. Wie wir anhand der Konfidenzintervalle sehen, wäre auf Basis der vorliegenden Daten auch eine Prävalenz von bis zu annähernd 5.8% möglich.

Da wir die Testentscheidung auch aus dem entsprechenden Konfidenzintervall ableiten können, können wir als Alternative auch die Funktion `binomCI` aus dem Paket "MKinfer" (Kohl (2022b)) verwenden und damit exakte, asymptotische oder Bootstrap-Konfidenzintervalle berechnen; zum Beispiel

```
1 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
```

Alle berechneten Bootstrap-Konfidenzintervalle schneiden den durch die Nullhypothese vorgegebenen Wahrscheinlichkeitsbereich von [0.05, 1]. Damit kann die Nullhypothese zum Signifikanzniveau von 5% nicht abgelehnt werden.

An diesem Beispiel lässt sich auch sehr schön das “p-hacking” demonstrieren. Hätte man anstelle von 5% eine Prävalenz von 6% angenommen, so wäre das Ergebnis signifikant (6% befindet sich außerhalb der Konfidenzintervalle); zum Beispiel

```
1 ## exakter Test
2 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:
0.0000000 0.05759556
sample estimates:
probability of success
0.04

```

In klinischen Studien begegnet man dem Problem, dass Annahmen oder sogar die gesamte statistische Analyse nach dem Vorliegen der Daten abgeändert werden mit der Forderung, dass jede Studie vor der Durchführung in einem entsprechenden Register eingetragen werden sollte. Alle wichtigen wissenschaftlichen Zeitschriften setzen dies voraus, wenn die Ergebnisse der entsprechenden Studie zur Veröffentlichung eingereicht werden. Zumindest im Fall der sogenannten primären Hypothesen, an denen man den Erfolg oder Misserfolg einer Studie festmacht, muss im Studienprotokoll auch genau angegeben werden, welche Ergebnisse man erwartet und welche statistischen Verfahren angewendet werden sollen. In vielen anderen Forschungsbereichen gibt es diese zusätzliche Absicherung nicht, weshalb auch nie wirklich sicher ist, ob nicht die statistische Analyse an die vorliegenden Daten angepasst wurde, um letztendlich signifikante Ergebnisse zu bekommen, was dann mit einem Erfolg der Studie gleichgesetzt wird.

Liegen bei einer Stichprobe mehr als zwei Merkmalsausprägungen vor, so kann man dies mit Hilfe eines Multinomialtests genauer untersuchen. Hierfür kann etwa die Funktion `multinomial.test` aus dem Paket "EMT" (Menzel (2021)) verwendet werden. Im Fall des 1-Stichproben Tests ist es hierbei nötig, für jede mögliche Kategorie deren erwartete Wahrscheinlichkeit anzugeben. Da die Berechnung des exakten Multinomialtests sehr aufwendig ist, verzichten wir auf ein Beispiel dazu.

Sollen zwei oder mehr Gruppen bezüglich eines binären Merkmals miteinander verglichen werden, so kommt es zusätzlich darauf an, ob die Gruppen voneinander abhängig sind oder nicht. Wir gehen zunächst einmal davon aus, dass zwei unabhängige Gruppen vorliegen und fassen die Daten in Form einer 2×2 -Kontingenztafel (vgl. Tabelle 6.2) zusammen. Ein erster möglicher Test in dieser Situation ist der

	0	1	Summe
Gruppe 1	a	b	$a + b$
Gruppe 2	c	d	$c + d$
Summe	$a + c$	$b + d$	$a + b + c + d$

Tabelle 6.2: Beispiel einer 2×2 -Kontingenztafel.

asymptotische **2-Stichproben Binomialtest**, der ebenfalls in der Funktion `prop.test` implementiert ist. Wir vergleichen die Prävalenz von Leberversagen von Frauen und Männern.

```
1 ## 2x2 Kontingenztafel
```

```
2 kont.tafel ← table(ITSDaten$Geschlecht, ITSDaten$Leberversagen)
3 kont.tafel
```

0	1
männlich	312 13
weiblich	168 7

```
1 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
```

Wir erhalten demnach keine signifikant unterschiedlichen Prävalenzen für Frauen und Männer. Für die asymptotische Berechnung kann auch die Funktion `binomDiffCI` aus dem Paket "MKinfer" (Kohl (2022b)) verwendet werden.

```
1 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
```

Da das Konfidenzintervall der Differenz die Null enthält, folgt auch hieraus wieder, dass es keine signifikant unterschiedlichen Prävalenzen für Frauen und Männer gibt. Es können mit dieser Funktion auch Bootstrap-Konfidenzintervalle berechnet werden.

```

1 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

```

Auch hier ergibt sich wieder kein signifikanter Unterschied. Eine alternative Betrachtungsweise von Kontingenztafeln führt auf hypergeometrische Verteilungen und den **exakten Test von Fisher**. Dieser ist in der Funktion `fisher.test` implementiert. Eine weitere asymptotische Möglichkeit der Analyse besteht durch einen χ^2 -Test, der mittels der Funktion `chisq.test` durchgeführt werden kann. In Fall des χ^2 -Tests sollten die Zählhäufigkeiten in den Zellen nicht zu klein sein. Je nach Quelle schwankt die Mindestanzahl zwischen 1 und 5 pro Zelle der Kontingenztafel. Wir wenden die beiden Tests auf die obige Situation an.

```

1 ## exakter Test von Fisher
2 fisher.test(kont.tafel)

Fisher's Exact Test for Count Data

data: kont.tafel
p-value = 1
alternative hypothesis: true odds ratio is not equal to 1
95 percent confidence interval:
0.3311862 2.7584805
sample estimates:
odds ratio
1

```

```
1 ## chi^2-Test
2 chisq.test(kont.tafel)
```

```
Pearson's Chi-squared test

data: kont.tafel
X-squared = 0, df = 1, p-value = 1
```

Auch hier ergibt sich in beiden Fällen keine Signifikanz. Im Fall des exakten Tests von Fisher bedeutet dies, dass die Odds-Ratio für das Leberversagen nicht signifikant von 1 verschieden ist; d.h., weder für Frauen noch für Männer kann von einem signifikant erhöhten Risiko ausgegangen werden. Im Fall des χ^2 -Tests können wir schließen, dass es keine signifikante (stochastische) Abhängigkeit zwischen dem Geschlecht und der Prävalenz von Leberversagen gibt.

Ein sehr interessantes Paket mit einer Vielzahl an statistischen Tests ist das Paket "coin" (Hothorn et al. (2006)). Es enthält exakte, asymptotische und approximative (auf Monte-Carlo Simulationen basierende) Tests. Einen approximativen χ^2 -Test können wir mit der Funktion `chisq_test` berechnen.

```
1 chisq_test(kont.tafel, distribution = "approximate")
```

```
Approximative Pearson Chi-Squared Test

data: Var2 by Var1 (männlich, weiblich)
chi-squared = 0, p-value = 1
```

Auch mit diesem auf Simulationen basierenden approximativen Testverfahren erhalten wir keine signifikante (stochastische) Abhängigkeit. Alle angewendeten Verfahren sind sich somit einig, dass die Nullhypothese beizubehalten ist.

Die verschiedenen Möglichkeiten, die wir für die Untersuchung von 2×2 -Kontingenztafeln kennengelernt haben, basieren auf unterschiedlichen Hypothesen, die nicht äquivalent sind. Bei den ersten asymptotischen Ansätzen werden direkt die Wahrscheinlichkeiten verglichen,

$$H_0 : p_1 - p_2 = 0 \quad \text{versus} \quad H_1 : p_1 - p_2 \neq 0$$

wobei p_1 und p_2 die wahren Wahrscheinlichkeiten der beiden Gruppen darstellen. Dies ist am einfachsten zu interpretieren und zeigt auch klar, welche Gruppe eine niedrigere oder höhere Wahrscheinlichkeit hat.

Im Fall des exakten Tests von Fisher ergibt sich eine Aussagen zu den **Chancen (Odds)** bzw. dem **Chancenverhältnis (OR = Odds-Ratio)**

$$H_0 : OR = 1 \quad \text{versus} \quad H_1 : OR \neq 1$$

d.h., falls $p_1 = p_2$ bzw. $p_1 - p_2 = 0$, so ist auch $\frac{p_1}{1-p_1} = \frac{p_2}{1-p_2}$ und

$$\text{OR} = \frac{\frac{p_1}{1-p_1}}{\frac{p_2}{1-p_2}} = 1$$

Leider ist es so, dass man ohne die Kenntnis von p_1 oder p_2 nicht vom Chancenverhältnis auf das **relative Risiko** $\frac{p_1}{p_2}$ oder p_2 bzw. p_1 zurückrechnen kann.

Im Fall das χ^2 -Tests basieren die Hypothesen auf der χ^2 -Statistik, welche die beobachteten Anzahlen mit den erwarteten Anzahlen vergleicht (siehe Gleichung (2.6))

$$H_0: \chi^2 = 0 \quad \text{versus} \quad H_1: \chi^2 > 0$$

Es besteht demnach ein Zusammenhang zu den Kontingenzkoeffizienten in Definition 2.9. Mit Hilfe der Funktion `CramerV` aus dem Paket "DescTools" (Andri et al. (2022)) kann auch ein Konfidenzintervall berechnet werden. Wir wählen in diesem Fall `method = "fisheradj"`, da die Berechnung mit der Standardmethode in diesem Beispiel Probleme macht.

```
1 CramerV(x = kont.tafel, conf.level = 0.95, method = "fisheradj")
```

Cramer V	lwr.ci	upr.ci
0.00000000	0.00000000	0.08769059

Das Konfidenzintervall enthält die Null, woraus wir schließen können, dass der Kontingenzkoeffizient nicht signifikant von Null verschieden ist. Wir erhalten demnach im Unterschied zu den beiden anderen Ansätzen keine Aussage zur Richtung des Unterschiedes.

Der exakte Test von Fisher und der χ^2 -Test können auch auf $r \times s$ -Kontingenztafeln ($r \geq 2, s \geq 2$) angewendet werden. Für wachsende Werte von r und s wird der exakte Test von Fisher jedoch immer rechenaufwendiger und es ist unter Umständen nötig, das Argument `workspace` auf einen höheren Wert zu setzen oder aber p-Werte heranzuziehen, die per Simulation gewonnen werden. Wir untersuchen den Zusammenhang zwischen der Operationsart und dem Ergebnis. In diesem Fall wäre es nötig, den `workspace` des exakten Tests von Fisher deutlich zu vergrößern. Wir verwenden daher stattdessen simulierte p-Werte.

```
1 kont.tafel <- table(ITSData$OP, ITSData$Ergebnis)
2 kont.tafel
```

	anderes	KH	nach Hause	Pflege/REHA	Verstorben
Andere	26	52	11	32	
Herz-Thorax	13	42	158	10	
Magen-Darm-Trakt	7	51	3	18	
Neuro	22	6	11	7	
Trauma	4	19	6	2	

```
1 ## exakter Test von Fisher
2 fisher.test(kont.tafel, simulate.p.value = TRUE, B = 1e5)
```

```
Fisher's Exact Test for Count Data with simulated p-value (based
on 1e+05 replicates)

data: kont.tafel
p-value = 1e-05
alternative hypothesis: two.sided

1 ## chi^2-Test
2 chisq.test(kont.tafel)

Pearson's Chi-squared test

data: kont.tafel
X-squared = 259.48, df = 12, p-value < 2.2e-16

1 chisq_test(kont.tafel, distribution = "approximate")

Approximative Pearson Chi-Squared Test

data: Var2 by
Var1 (Andere, Herz-Thorax, Magen-Darm-Trakt, Neuro, Trauma)
chi-squared = 259.48, p-value < 1e-04

1 CramerV(x = kont.tafel, conf.level = 0.95)

Cramer V      lwr.ci      upr.ci
0.4159155  0.3557842  0.4581223
```

Wir erhalten einen signifikanten Zusammenhang von moderater Stärke zwischen der Operationsart und dem Ergebnis. Durch eine Dichotomisierung könnte man aus der Kontingenztafel entsprechende 2×2 -Kontingenztafeln herauslösen und mit nachgelagerten Tests (post hoc Tests) die Zusammenhänge weiter untersuchen.

Es gibt auch noch eine Reihe alternativer Tests von Kontingenztafeln, auf die wir hier aber nicht näher eingehen werden; siehe etwa Campbell (2007).

Im Fall, dass zwei abhängige Gruppen vorliegen, kann der **Test von McNemar** angewendet werden. Es handelt sich dabei um einen χ^2 -Test, der mit Hilfe der Funktion `mcnemar.test` berechnet werden kann. Da im ITS-Datensatz keine abhängigen Messungen (z.B. vorher – nachher) enthalten sind, verwenden wir zur Demonstration die Daten aus dem Beispiel zur Funktion. Es handelt sich dabei um Daten über die Zustimmung zur Leistung des US-Präsidenten in zwei aufeinanderfolgenden Umfragen, wobei jeweils 1600 US-Amerikaner im Wahlalter befragt wurden.

```

1 Performance ← matrix(c(794, 86, 150, 570), nrow = 2,
2                               dimnames = list("First" = c("Approve", "Disapprove"),
3                                                 "Second" = c("Approve", "Disapprove")))
4 Performance

```

		Second	
		Approve	Disapprove
First	Approve	794	150
	Disapprove	86	570

Wir führen den McNemar Test durch.

```

1 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

```

Das signifikante Ergebnis zeigt, dass sich die Zustimmung zur Leistung des Präsidenten zwischen den beiden Umfragen signifikant verändert hat. Ein exakter, asymptotischer und approximativer McNemar Test kann auch mit Hilfe der Funktion `mh_test` aus dem Paket "coin" (Hothorn et al. (2006)) durchgeführt werden.

```
1 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

```

```
1 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

```

```
1 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
```

Alle drei Tests sind sich einig und zeigen ebenfalls eine signifikante Veränderung der Zustimmung an. Diese Situation kann auch mit der Funktion `binomDiffCI` aus dem Paket "MKinference" (Kohl (2022b)) analysiert werden.

```
1 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
```

Der Vorteil hierbei ist, dass man auch eine Aussage zur Richtung und Größe der Veränderung vornehmen kann. Es ergibt sich demnach, eine signifikante Veränderung (Abnahme) der Zustimmung um mehr als 2.1%. Es können anstelle der asymptotischen Konfidenzintervalle auch wieder Bootstrap-Intervalle berechnet werden.

```
1 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

```

Alle berechneten Bootstrap-Konfidenzintervalle liefern demnach sehr ähnliche Ergebnisse und bestätigen eine signifikante Veränderung (Abnahme) der Zustimmung um mindestens 2.1%.

Bei mehr als zwei abhängigen Gruppen kann der **Cochran–Mantel–Haenszel χ^2 -Test** angewendet werden, der in der Funktion `mantelhaen.test` implementiert ist. Wir verwenden zur Demonstration wieder Daten aus den Beispielen zur Funktion. Es handelt sich um die Daten von Hasen, die entweder sofort oder mit einer Verzögerung von 1.5 h nach einer Injektion von Streptokokken mit Penicillin behandelt wurden.

```

1 Rabbits <- array(c(0, 0, 6, 5,
2                               3, 0, 3, 6,
3                               6, 2, 0, 4,
4                               5, 6, 1, 0,
5                               2, 5, 0, 0), dim = c(2, 2, 5),
6 dimnames = list(Delay = c("None", "1.5h"),
7                   Response = c("Cured", "Died"),
8                   Penicillin.Level = c("1/8", "1/4", "1/2",
9                                         "1", "4")))
10 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

```

Neben dem asymptotischen Test gibt es auch einen exakten bedingten 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

```

```
1 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

```

Wir erhalten in beiden Fällen eine Odds-Ratio, die signifikant von Eins verschieden ist. Es folgt hieraus, dass das Risiko zu sterben bei einer um 1.5h verzögerten Behandlung mit Penicillin signifikant erhöht ist. Wir können auch die Funktion `cmh_test` aus dem Paket "coin" (Hothorn et al. (2006)) verwenden und einen asymptotischen, exakten und approximativen Cochran–Mantel–Haenszel Test berechnen.

```
1 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
```

```
1 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
```

```
1 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
```

Die Ergebnisse der Tests sprechen erneut dafür, dass eine sofortige Behandlung mit Penicillin zu einem besseren Behandlungserfolg führt. Hierbei liegt sowohl ein kleiner ($OR = 1.03$ bzw. $OR = 1.08$) und vermutlich klinisch wenig relevanter als auch ein enormer ($OR = 47.7$ bzw. $OR = 529.8$) Unterschied im Bereich des Möglichen. Ähnlich wie im Fall der unabhängigen Gruppen könnten auch hier wieder nachgelagerte Tests (post hoc Tests) zur weiteren Untersuchung der Zusammenhänge herangezogen werden, indem man etwa die entsprechenden Tests auf ausgewählte 2×2 -Kontingenztafeln anwendet. Es könnte hiermit zum Beispiel auch die Abhängigkeit von der Penicillindosis noch genauer untersucht werden.

6.3 Ordinale und quantitative Merkmale

In diesem Abschnitt werde ich zunächst die in der Abbildung 6.4 dargestellten Tests genauer besprechen. Das Baumdiagramm soll auch als Orientierung dienen, um den geeignetsten Test auszuwählen.

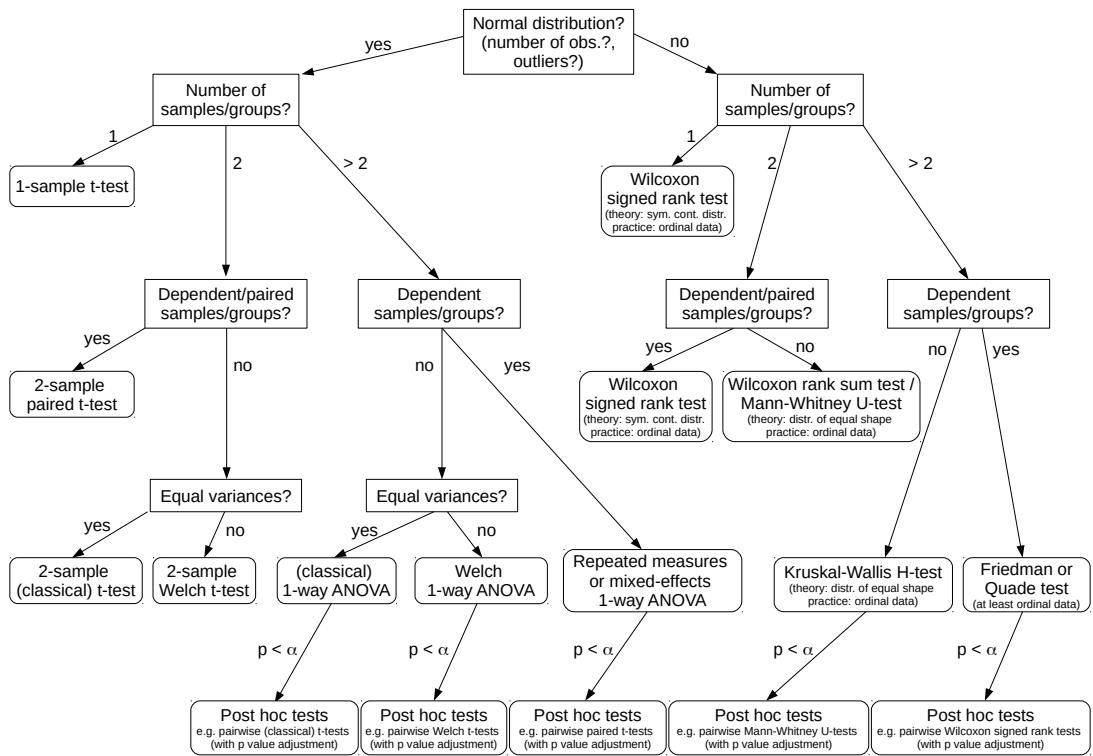


Abbildung 6.4: Baumdiagramm zur Auswahl des geeigneten Tests im Fall ordinaler oder quantitativer Merkmale.

Im Folgenden werden wir uns Schritt für Schritt durch das Diagramm arbeiten. Die erste Frage, die man sich bei der Auswahl eines geeigneten Tests stellen muss, ist, ob das zu untersuchende Merkmal einer Normalverteilung oder zumindest näherungsweise einer Normalverteilung folgt. Falls ja, spricht dies für die Auswahl eines t-Tests (linke Baumhälfte), aber nur wenn keine Ausreißer erwartet werden. Falls Ausreißer zu erwarten sind, entscheidet man sich besser für einen Rangtest (rechte Baumhälfte), der durch den Übergang zu Rängen eine gewisse Robustheit gegenüber Ausreißern aufweist. Ist die geplante Fallzahl pro Gruppe sehr klein (deutlich kleiner als 10), so muss man gewissermaßen an eine Normalverteilung „glauben“, da in diesem Fall die nichtparametrischen Rangtests unter Umständen gar keine Signifikanz liefern können, nicht einmal bei einer perfekten Separation der Gruppen. Je größer die Fallzahl ist, desto weniger spielen (aufgrund des zentralen Grenzwertsatzes) Abweichungen von der Normalverteilung eine Rolle. Bei einer geplanten Fallzahl von 50 oder mehr ist in der Regel die Normalapproximation des arithmetischen Mittels so gut, dass auch größere Abweichungen von der Normalverteilung (keine Ausreißer!) kein Problem für die Anwendung von t-Tests darstellen.

Wir beginnen mit der linken Seite des Baumes und dem wohl am häufigsten verwendeten statistischen Test, dem t-Test und seinen Varianten. Im einfachsten Fall liegt eine einzelne Stichprobe vor, deren Werte Realisationen von unabhängigen und Norm (μ, σ^2) -verteilten Zufallsvariablen sind. Untersucht wird der unbekannte Lageparameter μ , wobei die Varianz σ^2 unbekannt ist und somit ebenfalls geschätzt werden muss. Mögliche Nullhypotesen sind z.B. $\mu = \mu_0$ oder $\mu \leq \mu_0$, wobei $\mu_0 \in \mathbb{R}$ bekannt ist und für den

Test vorgegeben wird. Der entsprechende Test wird **1-Stichproben t-Test** genannt und kann mit Hilfe der Funktion `t.test` berechnet werden.

Wir verwenden wieder den ITS-Datensatz, der in Abschnitt 2.3 genauer beschrieben ist. Wie wir im Kapitel 5 gesehen haben, lässt sich die maximale Körpertemperatur der ITS-Patienten gut durch eine Normalverteilung beschreiben. Wir vermuten, dass wir bei den ITS-Patienten von einer erhöhten Temperatur ausgehen können; d.h.,

$$H_0: \mu \leq 37.5 \quad \text{versus} \quad H_1: \mu > 37.5$$

Wir verwenden hierzu den 1-Stichproben t-Test, wobei wir den Patient 398 außer Acht lassen. Die einseitige Alternative können wir mit Hilfe des Arguments `alternative = 'greater'` im Rahmen der Funktion `t.test` angeben. Mit dem Argument `mu = 37.5` können wir den Vergleichswert festlegen.

```
1 t.test(ITSDaten$Temperatur[-398], mu = 37.5, alternative = "greater")
```

```
One Sample t-test

data: ITSDaten$Temperatur[-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
```

Auf der Basis eines Signifikanzniveaus von 5% können wir also im Mittel von einer maximalen Körpertemperatur der ITS-Patienten von mehr als 37.5°C ausgehen. Der Wert 37.5 liegt entsprechend auch nicht im berechneten einseitigen 95% Konfidenzintervall. Der p-Wert steigt um mehr als den Faktor 1000 (!) an, wenn wir den deutlichen Ausreißer (Patient 398) hinzunehmen, wobei der Test weiterhin für die Alternative spricht.

```
1 t.test(ITSDaten$Temperatur, mu = 37.5, alternative = "greater")
```

```
One Sample t-test

data: ITSDaten$Temperatur
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
```

Der linke Endpunkt des Konfidenzintervalls hat sich um ca. 0.1 nach links verschoben und liegt jetzt nur unwesentlich über 37.5. Zwar erhalten wir in beiden Fällen ein signifikantes Ergebnis, dies sagt jedoch nichts über die klinische Relevanz aus. Da der Mittelwert unter Umständen nur minimal größer als 37.5°C ist, dürfte die klinische Relevanz nur gering sein.

Da in einer klinischen Studie das Weglassen von Patienten unter normalen Umständen nicht erlaubt ist und auch in anderen Kontexten schnell eine Datenmanipulation vermutet wird, wäre für diese Studie vielleicht auch die Wahl des entsprechenden 1-Stichproben Rangtests zu empfehlen gewesen. Es handelt sich im 1-Stichprobenfall um den **Wilcoxon Vorzeichen-Rangtest**, mit dem ganz allgemein gesprochen die Lokation der Daten untersucht wird. Im Fall von Symmetrie entspricht die Lokation dem Median, der in diesem Fall auch mit dem arithmetischen Mittel zusammenfällt.

```
1 wilcox.test(ITSData$Temperatur, mu = 37.5, alternative = "greater",
2               conf.int = TRUE)
```

```
Wilcoxon signed rank test with continuity correction

data: ITSData$Temperatur
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
```

Es handelt sich offenbar um die asymptotische Variante des Testes mit einer zusätzlichen Stetigkeitskorrektur. Hierbei wird standardmäßig kein Konfidenzintervall berechnet, sondern es muss hierfür zusätzlich `conf.int = TRUE` gesetzt werden. Wir entfernen den Ausreißer (Patient 398) und testen erneut.

```
1 wilcox.test(ITSData$Temperatur[-398], mu = 37.5, alternative = "greater",
2               conf.int = TRUE)
```

```
Wilcoxon signed rank test with continuity correction

data: ITSData$Temperatur[-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
```

Wir sehen, dass sich das Ergebnis dadurch nur unwesentlich verändert, wobei wir in beiden Fällen ein signifikantes Ergebnis bekommen; d.h., wir können aufgrund der (zumindest näherungsweisen) symmetrischen Situation sagen, dass der Median der maximalen Körpertemperatur signifikant größer als 37.5°C

ist. Die klinische Relevanz des Ergebnisses bleibt jedoch fragwürdig.

Es liegen zwei sogenannte verbundene oder gepaarte Stichproben vor, wenn wir zum Beispiel von einer Person zwei Werte des selben Merkmals zu unterschiedlichen Zeitpunkten erheben. Wir erhalten damit Wertepaare (x_i, y_i) , die wir als Realisationen von unabhängigen und identisch verteilten Zufallsvariablen (X_i, Y_i) ($i = 1, \dots, n, n \in \mathbb{N}$) auffassen. Weiter gilt: $D_i = X_i - Y_i \sim \mathcal{N}(\mu, \sigma^2)$, wobei der unbekannte Lageparameter μ interessiert und zusätzlich aber die Varianz σ^2 unbekannt ist. Mögliche Nullhypotesen sind etwa $\mu = \mu_0$ oder $\mu \leq \mu_0$ für ein vorgegebenes $\mu_0 \in \mathbb{R}$. Der Test heißt **gepaarter t-Test** und kann mit Hilfe der Funktion `t.test` unter Angabe des Arguments `paired = TRUE` berechnet werden. Der gepaarte t-Test entspricht gerade dem 1-Stichproben t-Test für die Differenz der Wertepaare. Analoges gilt für den 2-Stichproben Rangtest für gepaarte Stichproben. Der Test entspricht dem Wilcoxon Vorzeichen-Rangtest. Da im ITS-Datensatz keine wiederholten Messungen und damit keine gepaarten Daten vorliegen, verwenden wir zur Demonstration den `sleep` Datensatz aus dem Paket "datasets" (R Core Team (2022a)), welchen wir mit der Funktion `data` laden können. Die Probanden wurden mit zwei unterschiedlichen Schlafmitteln behandelt und es ist die Frage, ob sich die Veränderung der Schlafdauer für die beiden Medikamente unterscheidet.

```
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
```

Offenbar erhöht nur das zweite Medikament signifikant die Schlafzeit im Vergleich zur Kontrolle. Die Erhöhung beträgt mindestens 0.90 h. Wir vergleichen nun die beiden Medikamente (Datenpaare).

```
1 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
```

Das zweite Medikament ist signifikant besser als das erste Medikament und führt im Vergleich damit zu einer Schlafverlängerung von mindestens 0.70 h. Betrachten wir die Differenz zwischen den beiden Medikamenten und führen den 1-Stichproben t-Test durch, erhalten wir in der Tat das identische Ergebnis.

```
1 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
```

Wir können in dieser Situation auch alternativ den Wilcoxon Vorzeichen-Rangtest anwenden.

```
1 wilcox.test(sleep$extra[1:10], sleep$extra[11:20], paired = TRUE,
2           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
```

Wir erhalten zwar wieder ein signifikantes Ergebnis, aber auch eine Reihe von Warnungen, die hier im Buch unterdrückt sind. Der Auslöser für die Warnungen sind **Bindungen** im Datensatz; d.h., mindestens ein Wertepaar besteht aus zwei identischen Werten. Da man für den Wilcoxon Vorzeichen-Rangtest eine stetige Verteilung voraussetzt, dürfte dies nicht passieren und es ist unklar, ob der p-Wert und das Konfidenzintervall fehlerhaft sind. Eine exakte Variante des gepaarten Tests, welche auch bei Bindungen korrekte Ergebnisse liefert, ist in der Funktion `wilcoxonsign_test` des Paketes "coin" (Hothorn et al. (2006)) implementiert. Leider besitzt die Funktion keine Option, ein Konfidenzintervall mitberechnen zu lassen. Alternativ kann man auch auf die Funktion `wilcox.exact` des Paketes "exactRankTests" (Hothorn and Hornik (2022)) ausweichen, welches zwar nicht mehr aktiv weiterentwickelt wird, aber trotzdem verlässliche Ergebnisse liefert.

```
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
```

Die Werte haben sich leicht verändert. Es bleibt bei einer signifikanten Veränderung und einem Lokationsshift (bei Symmetrie: Differenz der Mediane) von mindestens 1.05 h.

Im nächsten Schritt betrachten wir die Situation, dass zwei unverbundene (unabhängige) Stichproben der Längen n_1 und n_2 aus $\mathcal{N}(\mu_1, \sigma^2)$ und $\mathcal{N}(\mu_2, \sigma^2)$ vorliegen, wobei die unbekannten Lageparameter interessieren und zugleich aber die Varianz σ^2 (bei beiden Stichproben gleich!) unbekannt ist. Diesen so genannten (klassischen) **2-Stichproben t-Test** oder **Student t-Test** haben wir ausführlich in Beispiel 6.5 besprochen. Er kann mit Hilfe der Funktion `t.test` und dem Argument `var.equal = TRUE` berechnet werden.

Dürfen zusätzlich die Varianzen der beiden Gruppen unterschiedlich sein, führt uns dies auf das sogenannte **Behrens-Fisher Problem**, welches näherungsweise mit dem **Welch t-Test** oder auch mit dem **Hsu t-Test** gelöst werden kann. Weitere Approximationen können mit Hilfe von Permutationen oder Bootstrap gefunden werden. Der Welch t-Test kann ebenfalls mit Hilfe der Funktion `t.test` berechnet werden kann, der Hsu t-Test etwa mit Hilfe der Funktion `hsu.t.test` aus dem Paket "MKinfer" (Kohl (2022b)).

Der entsprechende Permutations- und Bootstrap-Test sind zum Beispiel in den Funktionen `perm.t.test` und `boot.t.test` des Paketes "MKinfer" (Kohl (2022b)) implementiert. Es können mit diesen Funktionen per Permutationen bzw. Bootstrap auch die entsprechenden Varianten des 1-Stichproben und des Student t-Tests berechnet werden.

Wir wollen im Folgenden die Hypothese untersuchen, ob sich die mittlere maximale Körpertemperatur von Frauen (μ_1) und Männern (μ_2) signifikant unterscheidet. Es handelt sich hierbei um zwei unabhängige Gruppen und da wir für beide Gruppen von (näherungsweisen) Normalverteilungen ausgehen können, können wir dies mit Hilfe des 2-Stichproben t-Tests untersuchen. Offen ist, ob wir von gleichen Varianzen ausgehen können oder nicht. Wir berechnen die Varianzen für Frauen und Männer.

```
1 ## Frauen
2 sd(ITSDaten$Temperatur[ITSDaten$Geschlecht == "weiblich"])
```

```
[1] 1.258335
```

```
1 ## Männer
2 sd(ITSDaten$Temperatur[ITSDaten$Geschlecht == "männlich"])
```

```
[1] 1.946092
```

Die Werte beider Gruppen unterscheiden sich doch recht deutlich. Hierbei müssen wir jedoch beachten, dass es sich bei Patient 398 um einen Mann handelt. Wir berechnen nochmal die Standardabweichung der Männer, wobei wir Patient 398 weglassen.

```
1 ## Männer ohne Patient 398
2 sd(ITSDaten$Temperatur[-398][ITSDaten$Geschlecht[-398] == "männlich"])
```

```
[1] 1.123373
```

Erneut hat dieser eine Patient deutlich spürbare Auswirkungen auf das Ergebnis. Wir werden daher im Folgenden einmal mit und einmal ohne Patient 398 rechnen. In beiden Fällen gehen wir auf Nummer sicher und verwenden den Welch-t-Test, wobei wir die Aufteilung in die Geschlechter mittels der Formel `Temperatur ~ Geschlecht` vornehmen.

```
1 ## mit Patient 398
2 t.test(Temperatur ~ Geschlecht, data = ITSDaten)
```

```
Welch Two Sample t-test

data: Temperatur by Geschlecht
t = 0.37638, df = 481.71, p-value = 0.7068
alternative hypothesis: true difference in means between group männlich
and group weiblich is not equal to 0
95 percent confidence interval:
-0.2285543 0.3368620
```

```
sample estimates:
mean in group männlich mean in group weiblich
            37.68215          37.62800
```

```
1 ## ohne Patient 398
2 t.test(Temperatur ~ Geschlecht, data = ITSDaten[-398,])
```

```
Welch Two Sample t-test

data: Temperatur by Geschlecht
t = 1.2514, df = 323.73, p-value = 0.2117
alternative hypothesis: true difference in means between group männlich
and group weiblich is not equal to 0
95 percent confidence interval:
-0.08144621  0.36618695
sample estimates:
mean in group männlich mean in group weiblich
            37.77037          37.62800
```

Die Körpertemperatur der Frauen ist im Mittel etwas niedriger. Das Ergebnis des Tests (mit und ohne Patient 398) unterstützt aber die Hypothese, dass es sich hierbei lediglich um zufällige Schwankungen handelt. Wir können/müssen folglich davon ausgehen, dass sich die maximalen Körpertemperaturen von Männern und Frauen im Mittel nicht unterscheiden (Nullhypothese). Wir berechnen zum Vergleich den Hsu t-Test sowie den Permutations- und Bootstrap-Test.

```
1 hsu.t.test(Temperatur ~ Geschlecht, data = ITSDaten[-398,])
```

```
Hsu Two Sample t-test

data: Temperatur by Geschlecht
t = 1.2514, df = 174, p-value = 0.2125
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
-0.0821712  0.3669119
sample estimates:
mean of x mean of y   SD of x   SD of y
37.770370 37.628000  1.123373  1.258335
```

```
1 perm.t.test(Temperatur ~ Geschlecht, data = ITSDaten[-398,])
```

```
Permutation Welch Two Sample t-test

data: Temperatur by Geschlecht
(Monte-Carlo) permutation p-value = 0.2175
permutation difference of means (SE) = 0.1404711 (0.1100632)
95 percent (Monte-Carlo) permutation percentile confidence interval:
```

```

-0.07960847  0.35866667

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:
-0.08144621  0.36618695
sample estimates:
mean in group männlich mean in group weiblich
            37.77037             37.62800

```

```
1 boot.t.test(Temperatur ~ Geschlecht, data = ITSDaten[-398,])
```

```

Bootstrap Welch Two Sample t-test

data: Temperatur by Geschlecht
bootstrap p-value = 0.211
bootstrap difference of means (SE) = 0.1429764 (0.1134117)
95 percent bootstrap percentile confidence interval:
-0.08148986  0.36472760

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:
-0.08144621  0.36618695
sample estimates:
mean in group männlich mean in group weiblich
            37.77037             37.62800

```

Wir erhalten demnach in allen Fällen sehr ähnliche Ergebnisse. Wir stellen das Ergebnis des Welch t-Tests mit Hilfe des Paketes "ggplot2" (Wickham (2009)) grafisch dar. Man sollte immer darauf achten, dass die Grafik auch wirklich den Test widerspiegelt. Wir stellen daher Mittelwerte in Kombination mit den entsprechenden 95% Konfidenzintervallen dar. Wir bereiten hierfür zunächst einen `data.frame` mit den Mittelwerten und den oberen und unteren Grenzen der Konfidenzintervalle vor. Wir verwenden die Funktion `tapply`, um die Daten nach dem Geschlecht aufzuteilen und die entsprechende Funktion darauf anzuwenden.

```

1 AM <- tapply(ITSDaten$Temperatur[-398], ITSDaten$Geschlecht[-398], mean)
2 KI <- tapply(ITSDaten$Temperatur[-398], ITSDaten$Geschlecht[-398], meanCI)
3 n <- tapply(ITSDaten$Temperatur[-398], ITSDaten$Geschlecht[-398], length)
4 DF <- data.frame(AM = AM,
5                   KI.unten = c(KI$männlich$conf.int[1],
6                               KI$weiblich$conf.int[1]),
7                   KI.oben = c(KI$männlich$conf.int[2],
8                               KI$weiblich$conf.int[2]),
9                   Geschlecht = c("männlich", "weiblich"),
10                  n = n)

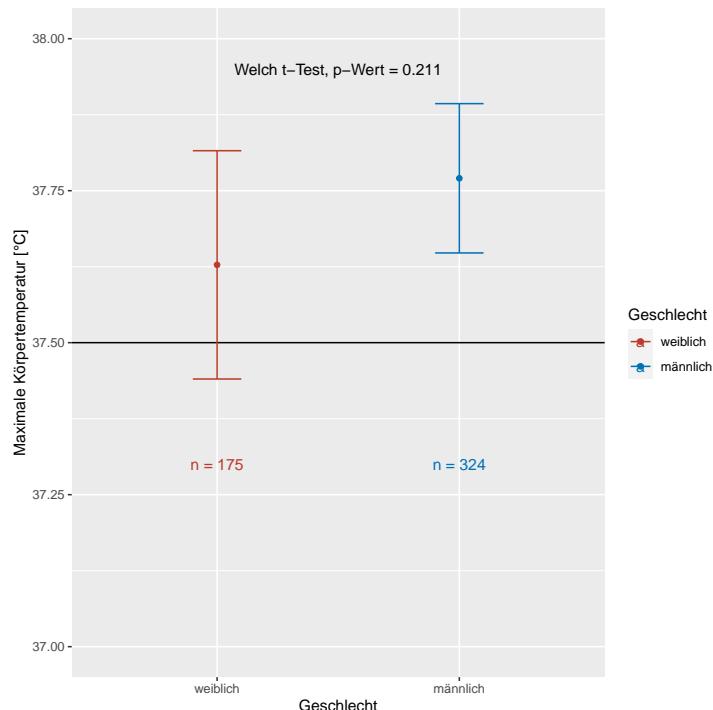
```

```
11 DF$Geschlecht ← factor(DF$Geschlecht, levels = c("weiblich", "männlich"))
12 DF
```

	AM	KI.unten	KI.oben	Geschlecht	n
männlich	37.77037	37.64759	37.89315	männlich	324
weiblich	37.62800	37.44026	37.81574	weiblich	175

Wir plotten das Ergebnis, wobei wir für die Farben die NEJM-Palette aus dem Paket "ggsci" (Xiao (2018)) wählen und mit den Funktionen `geom_text` und `annotate` zusätzlichen Text in der Graphik ergänzen.

```
1 ggplot(DF, aes(x = Geschlecht, y = AM, color = Geschlecht)) +
2   scale_color_nejm() + geom_point() + geom_hline(yintercept = 37.5) +
3   geom_errorbar(aes(ymin = KI.unten, ymax = KI.oben), width = 0.2) +
4   geom_text(aes(y = c(37.3, 37.3)), label = paste("n =", n))) +
5   annotate(geom = "text", x = 1.5, y = 37.95,
6           label = "Welch t-Test, p-Wert = 0.211") +
7   ylab("Maximale Körpertemperatur [°C]") + ylim(37.0, 38.0)
```



Anmerkung:

In der Praxis sollte man auf das methodisch fragwürdige Prüfen des Vorliegens gleicher Varianzen besser verzichten und in den allermeisten – wenn nicht allen – Fällen (bereits in der Planung!) den Welch t-Test wählen. Die Power des Welch t-Tests ist nur unwesentlich geringer als die Power des Student t-Tests und schützt aber umgekehrt vor einem Anstieg des Fehlers 1. Art, falls ungleiche Varianzen vorliegen (Ruxton (2006), Rasch et al. (2011)).

Falls Ausreißer vorliegen oder wir nicht von einer Normalverteilung ausgehen können und zugleich die Fallzahl klein (nicht zu klein!) bis moderat ist, so sollten wir auf den **Wilcoxon Rangsummentest** ausweichen, der auch als **Mann-Whitney U-Test** bekannt ist. Wir werden diesen Test im Folgenden durch **WMW-Test** abkürzen. Streng genommen ist dieser Test für zwei stetige Verteilungen von gleicher Form anwendbar. Dies impliziert insbesondere, dass die Varianzen der beiden Verteilungen gleich sein sollten wie beim Student t-Test. Empirische Resultate jedoch zeigen, dass eine leichte Verletzung der Varianzhomogenität den WMW-Test nicht beeinflusst. Dadurch, dass die Durchführung des Tests den Übergang zu Rängen beinhaltet, wird er trotz der oben angesprochenen Voraussetzungen auch als adäquat für ordinale Daten angesehen. Der Test ist ebenfalls in der Funktion `wilcox.test` implementiert. Darüber hinaus kann der Test mit der Funktion `wilcox_test` aus dem Paket "coin" (Hothorn et al. (2006)) sowie der Funktion `wilcox.exact` aus dem Paket "exactRankTests" (Hothorn and Hornik (2022)) berechnet werden.

Wir vergleichen weiter die maximale Körpertemperatur von Männern und Frauen.

```
1 ## ohne Patient 398
2 wilcox.test(Temperatur ~ Geschlecht, data = ITSDaten[-398,],
3             conf.int = TRUE)
```

```
Wilcoxon rank sum test with continuity correction

data: Temperatur by Geschlecht
W = 30488, p-value = 0.1643
alternative hypothesis: true location shift is not equal to 0
95 percent confidence interval:
-0.09997909 0.39995318
sample estimates:
difference in location
0.199972
```

```
1 ## mit Patient 398
2 wilcox.test(Temperatur ~ Geschlecht, data = ITSDaten, conf.int = TRUE)
```

```
Wilcoxon rank sum test with continuity correction

data: Temperatur by Geschlecht
W = 30488, p-value = 0.1833
alternative hypothesis: true location shift is not equal to 0
95 percent confidence interval:
-0.09993863 0.39999065
sample estimates:
difference in location
0.1000398
```

```
1 wilcox_test(Temperatur ~ Geschlecht, data = ITSDaten, conf.int = TRUE,
2             distribution = "approximate")
```

```
Approximative Wilcoxon-Mann-Whitney Test

data: Temperatur by Geschlecht (männlich, weiblich)
Z = 1.3309, p-value = 0.1879
alternative hypothesis: true mu is not equal to 0
95 percent confidence interval:
-0.1 0.4
sample estimates:
difference in location
0.1
```

```
1 wilcox.exact(Temperatur ~ Geschlecht, data = ITSDaten, conf.int = TRUE)
```

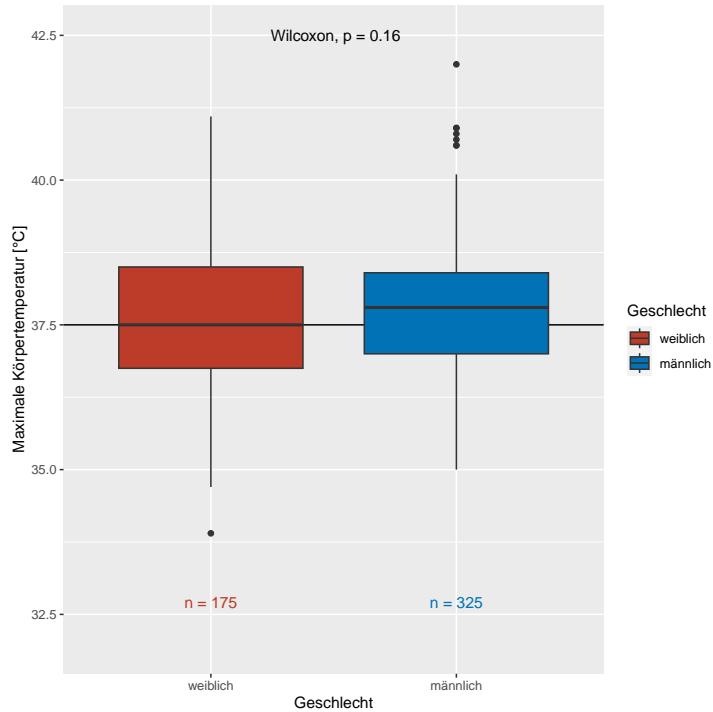
```
Asymptotic Wilcoxon rank sum test

data: Temperatur by Geschlecht
W = 30488, p-value = 0.1832
alternative hypothesis: true mu is not equal to 0
95 percent confidence interval:
-0.09998935 0.39994677
sample estimates:
difference in location
0.1000399
```

Die Ergebnisse sind in Übereinstimmung mit dem Welch t-Test, wobei die Auswirkung von Patient 398 wie bereits im 1-Stichproben Fall deutlich geringer ausfällt. Wir wollen nun die Ergebnisse in möglichst guter Übereinstimmung mit dem Testergebnis darstellen. Dem WMW-Test liegt der **Hodges-Lehmann Schätzer** (HL-Schätzer) zugrunde. Es handelt sich dabei um den Median aller paarweisen Differenzen. Im Fall des Wilcoxon Vorzeichen-Rangtests (1-Stichprobenfall) spricht man vom **Pseudomedian**, der sich berechnen lässt als der Median des Mittelwertes aller paarweisen Summen. Der Pseudomedian wiederum ist unter Symmetrie identisch zum Median. Da wir die beiden Gruppen nebeneinander darstellen wollen und die Verteilung der Daten nahezu symmetrisch ist, wählen wir für die Darstellung den Median bzw. Box- und Whisker Plots. Für die Ergänzung des Testergebnisses verwenden wir die Funktion `stat_compare_means` aus dem Paket "ggpubr" (Kassambara (2020)) in Kombinationen mit Funktionen des Paketes "ggplot2" (Wickham (2009)).

```
1 ggplot(data = ITSDaten, aes(x = factor(Geschlecht, levels = c("weiblich",
2                                         "männlich")),
3                     y = Temperatur,
4                     fill = factor(Geschlecht, levels = c("weiblich",
5                                         "männlich")))) +
6 geom_hline(yintercept = 37.5) + scale_fill_nejm(name = "Geschlecht") +
7 geom_boxplot() + stat_compare_means(label.y = 42.4, label.x = 1.35) +
8 ylab("Maximale Körpertemperatur [°C]") + ylim(32, 42.5) +
9 annotate(geom = "text", x = c(1, 2), y = c(32.7, 32.7),
```

```
10      label = c("n = 175", "n = 325"), color = pal_nejm()(2)) +
11      xlab("Geschlecht")
```



Wie in den meisten Fällen bedarf das Erstellen einer aussagekräftigen Grafik mit den Ergebnissen etwas Arbeit. Zum Beispiel wurde durch die Neudefinition des Faktors **Geschlecht** die alphabetische Reihenfolge aufgebrochen, womit nun zuerst "weiblich" und dann "männlich" dargestellt wird.

Natürlich kann es auch vorkommen, dass wir in einer Studie mehr als zwei Gruppen miteinander vergleichen möchten. Wir gehen zunächst von ***k* unabhängigen Gruppen** aus, die wir hinsichtlich eines normalverteilten Merkmals untersuchen wollen. Das Ziel ist es, herauszufinden, ob es Unterschiede im Mittelwert gibt. Man nennt dies eine **1-Weg ANOVA**, wobei ANOVA für "ANalysis Of VArience"(Varianzanalyse) steht. Wie im Fall des 2-Stichproben-t-Tests spricht man im Fall gleicher Varianzen von der klassischen 1-Weg ANOVA und nennt dies eine **Welch-1-Weg ANOVA**, falls die Varianzen unterschiedlich sein dürfen. Beide Varianten sind in der Funktion `oneway.test` implementiert. Das auf Rängen basierende Gegenstück zur 1-Weg ANOVA ist der **Kruskal-Wallis Test**, der manchmal auch als nichtparametrische 1-Weg ANOVA bezeichnet wird. Der Zusatz nicht-parametrisch macht deutlich, dass keine spezielle Verteilungsannahme, also insbesondere keine Normalverteilung, von Nöten ist. Auch bietet diese Analyse durch den Übergang zu Rängen wieder eine gewisse Robustheit gegenüber Ausreißern. Der Test kann in R mit Hilfe der Funktion `kruskal.test` durchgeführt werden.

Wir untersuchen weiter die maximale Körpertemperatur der ITS-Patienten, wobei wir dieses Mal die Patienten nach dem Ergebnis in Gruppen (d.h. $k = 4$) aufteilen. Im Fall der 1-Weg ANOVA vergleichen wir die Ergebnisse mit und ohne Patient 398 sowie mit und ohne der Annahme gleicher Varianzen. Wir müssen demnach die Funktion `oneway.test` viermal ausführen.

```
1 ## mit Patient 398, klassisch
2 oneway.test(Temperatur ~ Ergebnis, data = ITSDaten, var.equal = TRUE)
```

```
One-way analysis of means

data: Temperatur and Ergebnis
F = 1.9572, num df = 3, denom df = 496, p-value = 0.1195
```

```
1 ## mit Patient 398, Welch
2 oneway.test(Temperatur ~ Ergebnis, data = ITSDaten)
```

```
One-way analysis of means (not assuming equal variances)

data: Temperatur and Ergebnis
F = 4.5435, num df = 3.00, denom df = 182.28, p-value = 0.004262
```

```
1 ## ohne Patient 398, klassisch
2 oneway.test(Temperatur ~ Ergebnis, data = ITSDaten[-398,], var.equal = TRUE)
```

```
One-way analysis of means

data: Temperatur and Ergebnis
F = 5.2203, num df = 3, denom df = 495, p-value = 0.001481
```

```
1 ## ohne Patient 398, Welch
2 oneway.test(Temperatur ~ Ergebnis, data = ITSDaten[-398,])
```

```
One-way analysis of means (not assuming equal variances)

data: Temperatur and Ergebnis
F = 5.3574, num df = 3.00, denom df = 189.68, p-value = 0.001458
```

Die Annahme gleicher Varianz bei gleichzeitiger Berücksichtigung von Patient 398 führt dazu, dass sich bei der klassischen 1-Weg ANOVA kein signifikanter Unterschied zwischen den Gruppen feststellen lässt. In allen anderen Fällen ist der Unterschied zwischen den Gruppen signifikant. Insbesondere bei der klassischen 1-Weg ANOVA können demnach einzelne Ausreißer einen starken Einfluss auf die Ergebnisse haben.

Wir berechnen den Kruskal-Wallis Test, wobei wir die Funktion `kruskal.test` verwenden und einmal mit und einmal ohne Patient 398 rechnen.

```
1 ## mit Patient 398
2 kruskal.test(Temperatur ~ Ergebnis, data = ITSDaten)
```

```
Kruskal-Wallis rank sum test

data: Temperatur by Ergebnis
Kruskal-Wallis chi-squared = 15.259, df = 3, p-value = 0.001608

1 ## ohne Patient 398
2 kruskal.test(Temperatur ~ Ergebnis, data = ITSData[-398,])
```

```
Kruskal-Wallis rank sum test

data: Temperatur by Ergebnis
Kruskal-Wallis chi-squared = 15.869, df = 3, p-value = 0.001206
```

Wie bei den anderen Rangtests bestätigt sich einmal mehr, dass diese Verfahren nur wenig auf einzelne Ausreißer reagieren. Das Paket "coin" (Hothorn et al. (2006)) enthält die Funktionen `kruskal_test`, mit der unter anderem eine approximative Version des Kruskal-Wallis Tests berechnet werden kann.

```
1 kruskal_test(Temperatur ~ Ergebnis, data = ITSData,
2               distribution = "approximate")
```

```
Approximative Kruskal-Wallis Test

data: Temperatur by
      Ergebnis (anderes KH, nach Hause, Pflege/REHA, Verstorben)
chi-squared = 15.259, p-value = 0.0015
```

Insgesamt können wir von einem signifikanten Unterschied zwischen den Ergebnis-Gruppen ausgehen. Da wir es mit mehr als zwei Gruppen zu tun haben, bleiben die Fragen, welche Gruppen unterscheiden sich denn nun voneinander, wie genau sieht dieser Unterschied aus und wie groß sind überhaupt die Unterschiede (Effektstärken).

Wir nehmen an, dass wir die Mittelwerte oder allgemeiner die Lageparameter von mehr als zwei Gruppen mittels einer Variante der 1-Weg ANOVA miteinander verglichen haben und der Test ein signifikantes Ergebnis lieferte; d.h., der berechnete p-Wert ist kleiner als das vorgegebene Signifikanzniveau. In dieser Situation werden üblicherweise in einem zweiten Schritt sogenannte **post hoc Tests** in Kombination mit Berechnungen der Effekte und grafischen Darstellungen angewendet, um die Unterschiede zwischen den Gruppen herauszuarbeiten. Im Fall der 1-Weg ANOVA gibt es eine Reihe von Möglichkeiten von post hoc Tests. Die meiner Ansicht nach naheliegendste und einfachste Möglichkeit ist die Berechnung von paarweisen t-Tests. Dies ist mit Hilfe der Funktion `pairwise.t.t.test` möglich. Entsprechend ist es naheliegend im Fall des Kruskal-Wallis Tests paarweise WMW-Tests zu berechnen, was man mit Hilfe der Funktion `pairwise.wilcox.test` ausführen kann. Da bei dieser Analyse mehrere Tests gleichzeitig

durchgeführt werden, sollte man zusätzlich Korrekturen für das Signifikanzniveau bzw. die p-Werte verwenden, um den Fehler 1. Art unter Kontrolle zu behalten. Dies läuft unter dem Begriff **multiples Testen**, welches ich in Kapitel 7 genauer behandeln werde. In R wird hierbei standardmäßig die Korrektur von (Bonferroni-)Holm (Holm (1979)) angewendet.

Wir vergleichen die Ergebnisgruppen paarweise hinsichtlich der maximalen Körpertemperatur und verwenden dafür sowohl die Funktion `pairwise.t.test` als auch die Funktion `pairwise.wilcox.test`. Im Fall der t-Tests gehen wir von ungleichen Varianzen aus (Welch-t-Test) und schließen Patient 398 von den Berechnungen aus.

```
1 ## paarweise Welch t-Tests ohne Patient 398
2 pairwise.t.test(ITSDataen$Temperatur[-398], ITSDataen$Ergebnis[-398],
3                  pool.sd = FALSE)
```

```
Pairwise comparisons using t tests with non-pooled SD

data: ITSDataen$Temperatur [-398] and ITSDataen$Ergebnis [-398]

      anderes KH nach Hause Pflege/REHA
nach Hause 0.0459   -     -
Pflege/REHA 1.0000  0.0036   -
Verstorben  1.0000  0.0410  1.0000

P value adjustment method: holm
```

```
1 ## paarweise Wilcoxon-Mann-Whitney U-Tests
2 pairwise.wilcox.test(ITSDataen$Temperatur, ITSDataen$Ergebnis)
```

```
Pairwise comparisons using Wilcoxon rank sum test
with continuity correction

data: ITSDataen$Temperatur and ITSDataen$Ergebnis

      anderes KH nach Hause Pflege/REHA
nach Hause 0.0534   -     -
Pflege/REHA 1.0000  0.0024   -
Verstorben  1.0000  0.0534  1.0000

P value adjustment method: holm
```

Die Ausgabe entspricht den paarweisen (FWER-)adjustierten p-Werten, die in dieser Form mit dem ursprünglich gewählten Signifikanzniveau α verglichen werden können. Ein Vergleich mit $\alpha = 0.05$ zeigt, dass sich im Fall der paarweisen Welch t-Tests die “nach Hause”-Gruppe von anderen Gruppen signifikant unterscheidet. Im Fall der paarweisen WMW-Tests sind zwei der drei Vergleiche für die “nach Hause”-Gruppe knapp nicht signifikant und lediglich der Vergleich mit der “Pflege/REHA”-Gruppe ist signifikant. Die Art des Unterschiedes (signifikant kleiner oder größer) können wir durch Berechnung

der Mittelwerte bzw. der Hodges-Lehmann Schätzer untersuchen. Wir verwenden für die Berechnungen die Funktion `tapply` aus dem Paket "base" (R Core Team (2022a)) und die Funktion `pairwise.fun` aus dem Paket "MKinfer" (Kohl (2022b)).

```
1 ## Mittelwerte und SDs
2 tapply(ITSDaten$Temperatur[-398], ITSDaten$Ergebnis[-398], mean)

  anderes KH  nach Hause Pflege/REHA  Verstorben
  37.80694    37.44118    37.85185    37.96176
```



```
1 tapply(ITSDaten$Temperatur[-398], ITSDaten$Ergebnis[-398], sd)

  anderes KH  nach Hause Pflege/REHA  Verstorben
  0.9932627   1.0674633   1.1822532   1.4404632
```



```
1 ## Hodges-Lehmann Schätzer
2 pairwise.fun(ITSDaten$Temperatur[-398], ITSDaten$Ergebnis[-398],
3               function(x, y) wilcox.test(x, y, conf.int = TRUE)$estimate)

  anderes KH vs nach Hause anderes KH vs Pflege/REHA
  -0.39992021          0.09995271
  anderes KH vs Verstorben nach Hause vs Pflege/REHA
  0.10005373          0.40003222
  nach Hause vs Verstorben Pflege/REHA vs Verstorben
  0.49999961          0.09994576
```

Demnach weist die "nach Hause"-Gruppe den geringesten Mittelwert auf und auch die Ergebnisse beim HL-Schätzer bedeuten, dass jeweils die "nach Hause"-Gruppe die niedrigeren Werte besitzt. Wir können also im Fall einer Signifikanz sagen, dass die Werte der "nach Hause"-Gruppe signifikant niedriger sind. Alternativ können auch die Funktionen `pairwise.ext.t.test` und `pairwise.wilcox.exact` aus dem Paket "MKinfer" (Kohl (2022b)) verwendet werden, die eine umfassendere Ausgabe erzeugen.

```
1 ## paarweise Welch t-Tests ohne Patient 398
2 pairwise.ext.t.test(ITSDaten$Temperatur[-398], ITSDaten$Ergebnis[-398])

Pairwise Welch Two Sample t-tests

data: ITSDaten$Temperatur[-398] and ITSDaten$Ergebnis[-398]
alternative hypothesis: true difference in means is not equal to 0

  groups      t      df diff. in means       SE
1  anderes KH vs nach Hause -2.5605628 143.06918    -0.36576797 0.1428467
2  anderes KH vs Pflege/REHA  0.3091720 151.63933     0.04490741 0.1452506
3  anderes KH vs Verstorben  0.7362707 118.19309     0.15482026 0.2102763
4  nach Hause vs Pflege/REHA  3.4587352 356.99399     0.41067538 0.1187357
5  nach Hause vs Verstorben  2.6985241  97.79726     0.52058824 0.1929159
```

6 Pflege/REHA vs Verstorben	0.5645167	101.29157	0.10991285	0.1947026
2.5%	97.5%	p-value	adj. p-value	
1 -0.6481308	-0.08340515	0.0114868200	0.045947280	
2 -0.2420688	0.33188360	0.7576151000	1.0000000000	
3 -0.2615770	0.57121750	0.4630248000	1.0000000000	
4 0.1771660	0.64418470	0.0006083158	0.003649895	
5 0.1377430	0.90343350	0.0082054930	0.041027460	
6 -0.2763112	0.49613690	0.5736505000	1.0000000000	

```
1 ## paarweise Wilcoxon-Mann-Whitney U-Tests
2 pairwise.wilcox.exact(ITSData$Temperatur, ITSData$Ergebnis)
```

Pairwise Asymptotic Wilcoxon rank sum tests				
data: ITSData\$Temperatur and ITSData\$Ergebnis				
alternative hypothesis: true location shift is not equal to 0				
groups	W	diff.	in location	2.5%
1 anderes KH vs nach Hause	4849.0	-3.999203e-01	-0.69998380	
2 anderes KH vs Pflege/REHA	7009.0	9.995275e-02	-0.20003780	
3 anderes KH vs Verstorben	2581.5	9.993362e-02	-0.30004100	
4 nach Hause vs Pflege/REHA	19543.0	4.000034e-01	0.19999590	
5 nach Hause vs Verstorben	7073.0	4.999928e-01	0.09994986	
6 Pflege/REHA vs Verstorben	6603.5	3.633953e-05	-0.30002430	
97.5%	p-value	adj. p-value		
1 -0.09997455	0.0106402800	0.053201420		
2 0.39999110	0.7067306000	1.0000000000		
3 0.49999240	0.6874633000	1.0000000000		
4 0.69993780	0.0003939684	0.002363811		
5 0.80009050	0.0125830500	0.053201420		
6 0.40003520	0.8756310000	1.0000000000		

Für die grafische Darstellung des Ergebnisses der Welch 1-Weg ANOVA wählen wir Mittelwerte und entsprechende 95% Konfidenzintervalle. Dies kommt den Berechnungen, die beim Welch t-Test durchgeführt werden recht nahe. Genauer müsste man die Differenz der Mittelwerte und die zugehörigen 95% Konfidenzintervalle darstellen, wobei das Konfidenzniveau ganz genau genommen auch noch adjustiert werden müsste. Hieraus lassen sich dann aber nicht die Werte der einzelnen Gruppen ablesen, die man gerne ebenfalls visualisieren möchte. Wir bereiten hierfür zunächst wieder einen `data.frame` mit den Mittelwerten und den oberen und unteren Grenzen der Konfidenzintervalle vor. Wir verwenden für die Berechnungen die Funktion `tapply`.

```
1 AM <- tapply(ITSData$Temperatur[-398], ITSData$Ergebnis[-398], mean)
2 KI <- tapply(ITSData$Temperatur[-398], ITSData$Ergebnis[-398], meanCI)
3 n <- tapply(ITSData$Temperatur[-398], ITSData$Ergebnis[-398], length)
4 DF <- data.frame(AM = AM,
5                   KI.unten = c(KI$`anderes KH`$conf.int[1],
6                               KI$`nach Hause`$conf.int[1],
```

```

7   KI$ `Pflege/REHA`$conf.int[1],
8   KI$Verstorben$conf.int[1]),
9   KI.oben = c(KI`anderes KH`$conf.int[2],
10  KI`nach Hause`$conf.int[2],
11  KI`Pflege/REHA`$conf.int[2],
12  KI$Verstorben$conf.int[2]),
13  Ergebnis = names(KI),
14  n = n)
15 DF$Ergebnis <- factor(DF$Ergebnis, levels = c("nach Hause", "Pflege/REHA",
16  "anderes KH", "Verstorben"))
17 DF

```

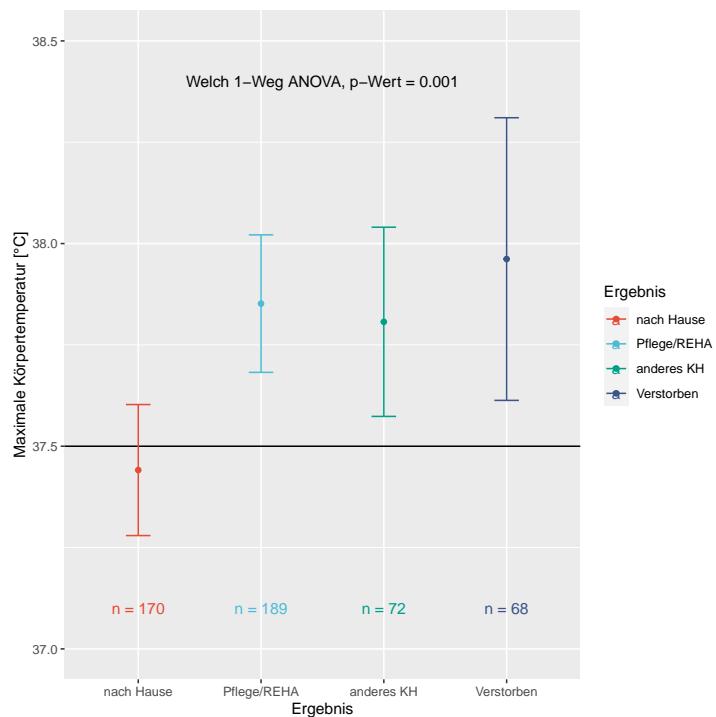
	AM	KI.unten	KI.oben	Ergebnis	n
anderes KH	37.80694	37.57354	38.04035	anderes KH	72
nach Hause	37.44118	37.27956	37.60280	nach Hause	170
Pflege/REHA	37.85185	37.68221	38.02149	Pflege/REHA	189
Verstorben	37.96176	37.61310	38.31043	Verstorben	68

Wir sehen anhand des Ergebnisses insbesondere auch, dass sich die Konfidenzintervalle der einzelnen Gruppen durchaus leicht überschneiden können und der Unterschied trotzdem signifikant ist. Die Darstellung spiegelt folglich nicht vollständig die durchgeführte Analyse wider. Auch kann man hieraus schließen, dass die gleichzeitige Analyse von zwei Gruppen eine etwas höhere Power hat, als wenn man die beiden Gruppen separat betrachten würde. Wir plotten das Ergebnis und wählen für die Farben die NEJM-Palette aus dem Paket "ggsci" (Xiao (2018)) und annotieren den Plot mit den Funktionen `geom_text` und `annotate`.

```

1 ggplot(DF, aes(x = Ergebnis, y = AM, color = Ergebnis)) +
2   scale_color_npg() + geom_point() + geom_hline(yintercept = 37.5) +
3   geom_errorbar(aes(ymin = KI.unten, ymax = KI.oben), width = 0.2) +
4   geom_text(aes(y = rep(37.1, 4), label = paste("n =", n))) +
5   annotate(geom = "text", x = 2.5, y = 38.4,
6           label = "Welch 1-Weg ANOVA, p-Wert = 0.001") +
7   ylab("Maximale Körpertemperatur [°C]") + ylim(37.0, 38.5)

```

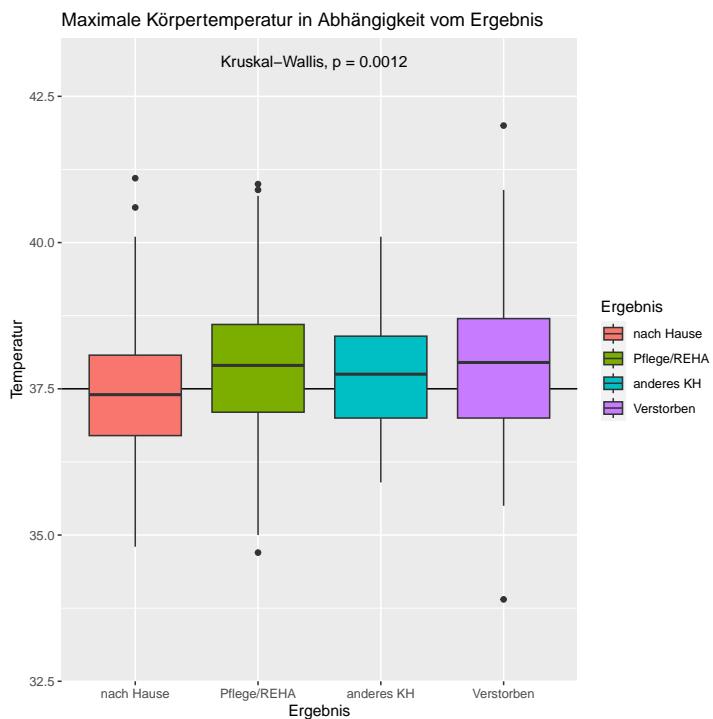


Ausgehend von einer näherungsweisen Symmetrie stellen wir das Ergebnis des Kruskal-Wallis Tests analog zum Fall des WMW-Tests mittels Box-und-Whisker Plots grafisch dar. Wir verwenden für die Darstellung die Funktion ggplot aus dem Paket "ggplot2" (Wickham (2009)). Zur besseren Übersichtlichkeit zeigen wir den Wert von Patient 398 nicht, der zur Gruppe der Verstorbenen gehört.

```

1 ITStmp <- ITSDaten[, c("Ergebnis", "Temperatur")]
2 ITStmp$Ergebnis <- factor(ITStmp$Ergebnis, levels = c("nach Hause",
3                                         "Pflege /REHA",
4                                         "anderes KH",
5                                         "Verstorben"))
6 ggplot(data=ITStmp, aes(x=Ergebnis, y=Temperatur, fill=Ergebnis)) +
7   geom_hline(yintercept = 37.5) + geom_boxplot() +
8   stat_compare_means(label.y = 43, label.x = 2.0) +
9   ggtitle("Maximale Körpertemperatur in Abhängigkeit vom Ergebnis") +
10  ylim(c(33, 43))

```



Die maximale Körpertemperatur der Gruppe, die nach Hause entlassen wurde, ist folglich im Mittel niedriger als bei den anderen Gruppen.

Anmerkung:

Beim Vergleich der Lage von zwei oder mehr unabhängigen Gruppen (Stichproben), die eine ähnliche Varianz aufweisen, wird die Power des Vergleichs im Wesentlichen durch die kleinste Gruppe vorgegeben. Daher sollte man bei Studien darauf achten, soweit dies möglich ist, dass man idealer Weise gleich große Gruppen miteinander vergleicht. Bei gleich großen Gruppen spricht man auch von einem **balancierten Design**.

Im Fall, dass es sich bei den Gruppen nicht um unabhängige, sondern abhängige (verbundene) Gruppen handelt (z.B. Messungen von denselben Personen an unterschiedlichen Zeitpunkten) spricht man von einer **1-Weg ANOVA mit Messwiederholungen** (repeated measures). Man spricht auch von Inner- (within-) und Zwischen- (between-) Subjektfaktoren. Der Innersubjektfaktor entspricht in vielen Fällen unterschiedlichen Zeitpunkten, während der Zwischensubjektfaktor die Behandlung beschreibt. Neben diesem klassischen Ansatz werden für die Analyse heute vermehrt sogenannte gemischte (mixed-effects) Modelle verwendet, bei denen das Subjekt (Innersubjektfaktor), von dem mehrere Messungen vorliegen als ein sogenannter Zufallseffekt modelliert wird. Darüber hinaus gibt es auch wieder rangbasierte nicht-parametrische Ansätze zur Analyse unter anderem den Quade oder den Friedman Test. Der **Quade Test** kann als die Erweiterung des Wilcoxon Vorzeichen-Rangtests auf mehr als zwei abhängige Gruppen angesehen werden. Der **Friedman Test** ist konservativer und entspricht der Erweiterung des Vorzeichen-/Mediantests (ein spezieller Binomialtest) auf mehr als zwei verbundene Gruppen.

Da im ITS-Datensatz keine Variablen mit Messwiederholungen enthalten sind, verwenden wir den Datensatz `selfesteem` aus dem Paket "datarium" (Kassambara (2019)), in dem Daten zum Selbstwertgefühl von 10 Personen enthalten sind. Bei diesen Personen wurde im Rahmen einer speziellen Diät an

drei Zeitpunkten ein Score für das Selbstwertgefühl ermittelt.

```
1 data(selfesteem)
2 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

Wir bringen die Daten vom vorgegebenen “breiten” auf ein entsprechendes “langes” Format und führen anschließend die oben vorgestellten 1-Weg ANOVAs durch. Mit den Funktionen `head` und `tail` geben wir die ersten und letzten sechs Zeilen des “langen” Datensatzes aus.

```
1 SE.lang <- data.frame(id = rep(selfesteem$id, 3),
2                         Score = c(selfesteem$t1, selfesteem$t2,
3                                     selfesteem$t3),
4                         Zeitpunkt = rep(c("t1", "t2", "t3"), each = 10))
5 head(SE.lang)
```

	id	Score	Zeitpunkt
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

```
1 tail(SE.lang)
```

	id	Score	Zeitpunkt
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

Wir können alle Berechnungen mit Hilfe der Funktion `rm.oneway.test` aus dem Paket "MKinfer" (Kohl (2022b)) durchführen.

```
1 ## Klassische repeated-measures 1-Weg ANOVA
2 rm.oneway.test(x = SE.lang$Score , g = SE.lang$Zeitpunkt , id = SE.lang$id)
```

```
Repeated measures 1-way ANOVA

data: SE.lang$Score , SE.lang$Zeitpunkt and SE.lang$id
F = 55.469, num df = 2, denom df = 18, p-value = 2.014e-08
```

```
1 ## Mixed-effects 1-Weg ANOVA
2 rm.oneway.test(x = SE.lang$Score , g = SE.lang$Zeitpunkt , id = SE.lang$id ,
3                 method = "lme")
```

```
Mixed-effects 1-way ANOVA

data: SE.lang$Score , SE.lang$Zeitpunkt and SE.lang$id
F = 65.261, num df = 2, denom df = 18, p-value = 5.641e-09
```

```
1 ## Quade-Test
2 rm.oneway.test(x = SE.lang$Score , g = SE.lang$Zeitpunkt , id = SE.lang$id ,
3                 method = "quade")
```

```
Quade test

data: x, g and id
Quade F = 24.673, num df = 2, denom df = 18, p-value = 6.96e-06
```

```
1 ## Friedman-Test
2 rm.oneway.test(x = SE.lang$Score , g = SE.lang$Zeitpunkt , id = SE.lang$id ,
3                 method = "friedman")
```

```
Friedman rank sum test

data: x, g and id
Friedman chi-squared = 18.2, df = 2, p-value = 0.0001117
```

Offenbar sind sich die verschiedenen Verfahren einig und es gibt eine signifikante Veränderung über die Zeit. Wir wollen diese Veränderung mit Hilfe von gepaarten t-Tests und Wilcoxon Vorzeichen-Rangtests genauer untersuchen.

```
1 ## Paarweise gepaarte t-Tests
2 pairwise.t.test(SE.lang$Score , SE.lang$Zeitpunkt , paired = TRUE)
```

```
Pairwise comparisons using paired t tests
```

```

data: SE.lang$Score and SE.lang$Zeitpunkt

  t1      t2
t2 0.0015 -
t3 1e-06  0.0015

P value adjustment method: holm

```

```

1 ## Paarweise Wilcoxon Vorzeichen-Rangtests
2 pairwise.wilcox.test(SE.lang$Score, SE.lang$Zeitpunkt, paired = TRUE)

```

```

Pairwise comparisons using Wilcoxon signed rank exact test

data: SE.lang$Score and SE.lang$Zeitpunkt

  t1      t2
t2 0.0059 -
t3 0.0059 0.0059

P value adjustment method: holm

```

Wir erhalten paarweise signifikante Unterschiede zwischen allen Zeitpunkten. Da wir es mit gepaarten Werten zu tun haben und wir wissen, dass die gepaarten Tests identisch sind zum 1-Stichprobentest der Differenzen, berechnen wir die Mittelwerte und SDs sowie Mediane für die paarweisen Differenzen der Zeitpunkte. Hierzu verwenden wir die Funktion `pairwise.fun` aus dem Paket "MKinfer" (Kohl (2022b)), mit der eine beliebige vorgegebene Funktion auf Paare von Gruppen angewendet werden kann.

```

1 ## Mittelwerte
2 pairwise.fun(SE.lang$Score, SE.lang$Zeitpunkt, function(x, y) mean(x-y))

```

```

t1 vs t2 t1 vs t3 t2 vs t3
1.79382 4.49622 2.70240

```

```

1 ## SDs
2 pairwise.fun(SE.lang$Score, SE.lang$Zeitpunkt, function(x, y) sd(x-y))

```

```

t1 vs t2 t1 vs t3 t2 vs t3
1.141907 1.074852 1.755559

```

```

1 ## Mediane
2 pairwise.fun(SE.lang$Score, SE.lang$Zeitpunkt, function(x, y) median(x-y))

```

```

t1 vs t2 t1 vs t3 t2 vs t3
1.245675 4.623277 2.998624

```

Wir erhalten demnach von Zeitpunkt t1 nach Zeitpunkt t3 ansteigende Scores. Alternativ können wir auch wieder die Funktionen `pairwise.t.test` und `pairwise.wilcox.exact` aus dem Paket "MKinfer" (Kohl (2022b)) verwenden.

```
1 ## Paarweise gepaarte t-Tests
2 pairwise.t.test(SE.lang$Score, SE.lang$Zeitpunkt, paired = TRUE)
```

```
Pairwise Paired t-tests

data: SE.lang$Score and SE.lang$Zeitpunkt
alternative hypothesis: true mean of differences is not equal to 0

  groups      t df mean of diffs       SE    2.5%   97.5%
1 t1 vs t2 4.967618 9     1.79382 0.3611027 2.610691
2 t1 vs t3 13.228148 9     4.49622 0.3398979 5.265122
3 t2 vs t3 4.867816 9     2.70240 0.5551565 3.958251

  p-value adj. p-value
1 7.724196e-04 1.544839e-03
2 3.343804e-07 1.003141e-06
3 8.861912e-04 1.544839e-03
```

```
1 ## Paarweise Wilcoxon Vorzeichen-Rangtests
2 pairwise.wilcox.exact(SE.lang$Score, SE.lang$Zeitpunkt, paired = TRUE)
```

```
Pairwise Exact Wilcoxon signed rank tests

data: SE.lang$Score and SE.lang$Zeitpunkt
alternative hypothesis: true location shift is not equal to 0

  groups   W (pseudo)median      2.5%   97.5%       p-value adj. p-value
1 t1 vs t2 55          1.445577 1.113997 2.766025 0.001953125 0.005859375
2 t1 vs t3 55          4.425733 3.668177 5.232444 0.001953125 0.005859375
3 t2 vs t3 54          2.692048 1.421954 3.941918 0.003906250 0.005859375
```

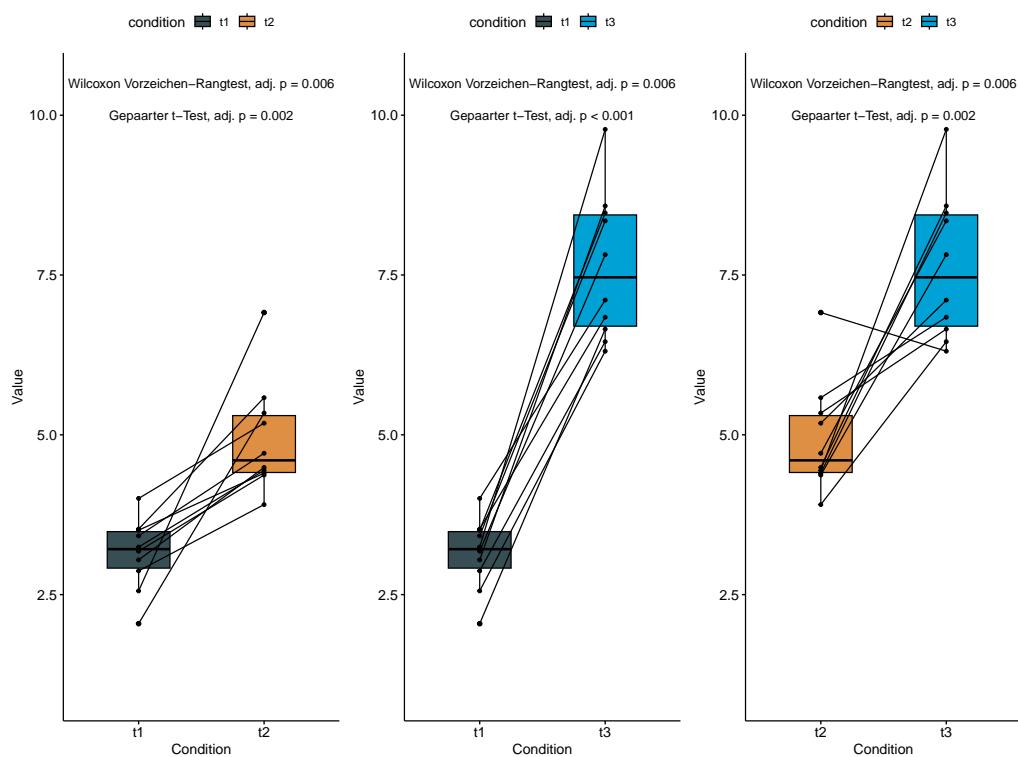
Wir stellen die paarweisen Differenzen mit Hilfe der Funktion `ggpairs` aus dem Paket `ggpubr` (Kassambara (2020)) grafisch dar. Wir verwenden die Funktion `grid.arrange` des Paketes "gridExtra" (Auguie (2017)), um die drei Plots nebeneinander darzustellen.

```
1 gg1 <- ggpairs(data = selfesteem, cond1 = "t1", cond2 = "t2",
2                   fill = "condition") + ylim(1, 10.5) +
3                   scale_fill_manual(values = pal_jama()(3)[1:2]) +
4                   annotate(geom = "text", x = 1.5, y = 10.5,
5                           label = "Wilcoxon Vorzeichen-Rangtest, adj. p = 0.006") +
6                   annotate(geom = "text", x = 1.5, y = 10,
7                           label = "Gepaarter t-Test, adj. p = 0.002")
8 gg2 <- ggpairs(data = selfesteem, cond1 = "t1", cond2 = "t3",
9                   fill = "condition") + ylim(1, 10.5) +
```

```

10  scale_fill_manual(values = pal_jama()(3)[c(1,3)]) +
11  annotate(geom = "text", x = 1.5, y = 10.5,
12          label = "Wilcoxon Vorzeichen-Rangtest, adj. p = 0.006") +
13  annotate(geom = "text", x = 1.5, y = 10,
14          label = "Gepaarter t-Test, adj. p < 0.001")
15 gg3 <- ggpaired(data = selfesteem, cond1 = "t2", cond2 = "t3",
16                   fill = "condition") + ylim(1, 10.5) +
17  scale_fill_manual(values = pal_jama()(3)[c(2,3)]) +
18  annotate(geom = "text", x = 1.5, y = 10.5,
19          label = "Wilcoxon Vorzeichen-Rangtest, adj. p = 0.006") +
20  annotate(geom = "text", x = 1.5, y = 10,
21          label = "Gepaarter t-Test, adj. p = 0.002")
22 grid.arrange(gg1, gg2, gg3, nrow = 1)

```



Auch grafisch sind deutliche Unterschiede zwischen den Zeitpunkten zu erkennen, wobei bei den allermeisten Probanden die Scores über die Zeit klar ansteigen.

Damit haben wir alle Tests der Abbildung 6.4 besprochen und wir wenden uns im Folgenden noch einigen weiteren wichtigen statistischen Tests zu.

In manchen Fällen interessieren uns nicht die Mittelwerte, sondern die Varianzen von zwei unabhängigen Gruppen. Liegt ein Merkmal vor, welches in beiden Gruppen normalverteilt ist, so basiert der entsprechende Test auf dem Quotienten der beiden Varianzen. Nach Bemerkung 4.28 führt uns dies auf eine F-Verteilung und man spricht entsprechend von einem **F-Test**. Wir können diesen Test mit Hilfe der Funktion `var.test` durchführen.

Das nichtparametrische Gegenstück zum F-Test, welches ohne Normalverteilungsannahme auskommt

und auf Rängen basiert ist der **Ansari-Bradley Test**. Er kann mit Hilfe der Funktion `ansari.test` angewendet werden.

Wie wir oben gesehen haben, sind die Varianzen der maximalen Körpertemperatur von Männern und Frauen verschieden. Wir untersuchen, ob es sich hierbei lediglich um eine zufällige Schwankung handelt oder nicht, wobei wir einmal mit und einmal ohne Patient 398 rechnen.

```
1 ## mit Patient 398
2 var.test(Temperatur ~ Geschlecht, data = ITSDaten)
```

```
F test to compare two variances

data: Temperatur by Geschlecht
F = 2.3919, num df = 324, denom df = 174, p-value = 6.066e-10
alternative hypothesis: true ratio of variances is not equal to 1
95 percent confidence interval:
1.832443 3.089714
sample estimates:
ratio of variances
2.391851
```

```
1 ## ohne Patient 398
2 var.test(Temperatur ~ Geschlecht, data = ITSDaten[-398,])
```

```
F test to compare two variances

data: Temperatur by Geschlecht
F = 0.797, num df = 323, denom df = 174, p-value = 0.08265
alternative hypothesis: true ratio of variances is not equal to 1
95 percent confidence interval:
0.6105184 1.0296970
sample estimates:
ratio of variances
0.7969952
```

Im Fall der Varianz ist der Einfluss des Ausreißers (Patient 398) noch stärker als im Fall des Mittelwertes. Je nachdem, ob wir mit oder ohne Patient 398 testen, erhalten wir einen signifikanten Unterschied oder eben nicht. Der p-Wert ändert sich um den Faktor 10^8 !

Wir wollen untersuchen, ob dies auch auf den Ansari-Bradley Test zutrifft und verwenden hierfür die Funktion `ansari.test`. Im Unterschied zum F-Test werden beim Ansari-Bradley Test die Skalenwerte selbst und nicht deren Quadrate (Varianz) miteinander verglichen; d.h., das Verhältnis der Skalenwerte muss quadriert werden, um es direkt mit dem Verhältnis der Varianzen vergleichen zu können. Für die Berechnung des Verhältnisses der Skalenwerte sowie des entsprechenden Konfidenzintervalls müssen die Lageparameter der beiden Gruppen gleich sein. Daher empfiehlt es sich vor der Berechnung die Daten am Median zu zentrieren. Neben der Funktion `ansari.test` verwenden wir die Funktion `tapply` für die Berechnungen.

```
1 ITStmp <- ITSDaten[,c("Temperatur", "Geschlecht")]
2 tapply(ITStmp$Temperatur, ITStmp$Geschlecht, median)
```

männlich	weiblich
37.8	37.5

```
1 Mann <- ITStmp$Geschlecht == "männlich"
2 ITStmp$Temperatur[Mann] <- ITStmp$Temperatur[Mann] - 37.8
3 ITStmp$Temperatur[!Mann] <- ITStmp$Temperatur[!Mann] - 37.5
4 ansari.test(Temperatur ~ Geschlecht, data = ITStmp, conf.int = TRUE)
```

```
Ansari-Bradley test

data: Temperatur by Geschlecht
AB = 41490, p-value = 0.3629
alternative hypothesis: true ratio of scales is not equal to 1
95 percent confidence interval:
0.7692467 1.0909439
sample estimates:
ratio of scales
0.9473912
```

```
1 ansari.test(Temperatur ~ Geschlecht, data = ITStmp[-398,], conf.int = TRUE)
```

```
Ansari-Bradley test

data: Temperatur by Geschlecht
AB = 41332, p-value = 0.3299
alternative hypothesis: true ratio of scales is not equal to 1
95 percent confidence interval:
0.7619207 1.0833571
sample estimates:
ratio of scales
0.937491
```

Die Ergebnisse zeigen, dass der Einfluss des Ausreißers (Patient 398) deutlich kleiner ist, wie wir es auch bei den anderen Rangtests bereits gesehen haben. Zusammendfassend müssen wir davon ausgehen, dass die Hypothese einer gleichen Varianz/Skala nicht abgelehnt werden kann.

Ein weiterer wichtiger Test, besteht darin, zu untersuchen, ob es eine signifikante Korrelation zwischen zwei Variablen gibt. Gehen wir davon aus, dass zwei normalverteilte Merkmale vorliegen, so folgt hieraus, dass der Zusammenhang dieser beiden Variablen linear ist. Folglich können wir in diesem Fall die Stärke des Zusammenhangs mit Hilfe der Pearson Korrelation ρ untersuchen. Die zugehörige Teststatistik folgt einer t-Verteilung mit $n - 2$ Freiheitsgraden, die Nullhypothese lautet $\rho = 0$. Können wir nicht von einer

Normalverteilung ausgehen und müssen wir allgemeiner von einem monotonen Zusammenhang ausgehen, so stellen die Korrelationen nach Spearman und Kendall gute Alternativen dar. Die zugehörigen drei Tests können mit Hilfe der Funktion `cor.test` durchgeführt werden.

Wir untersuchen, ob der Zusammenhang zwischen der maximalen Körpertemperatur und der maximalen Herzfrequenz signifikant von 0 verschieden ist. Wir verwenden hierfür die Funktion `cor.test` und untersuchen Pearson-, Spearman- und Kendall-Korrelation, wobei wir im Fall der Pearson-Korrelation Patient 398 von den Berechnungen ausschließen.

```
1 ## Pearson ohne Patient 398
2 cor.test(ITSDaten$Temperatur[-398], ITSDaten$Herzfrequenz[-398])
```

```
Pearson's product-moment correlation

data: ITSDaten$Temperatur[-398] and ITSDaten$Herzfrequenz[-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
```

```
1 ## Spearman
2 cor.test(ITSDaten$Temperatur, ITSDaten$Herzfrequenz, method = "spearman")
```

```
Spearman's rank correlation rho

data: ITSDaten$Temperatur and ITSDaten$Herzfrequenz
S = 15291695, p-value = 1.521e-09
alternative hypothesis: true rho is not equal to 0
sample estimates:
rho
0.2659957
```

```
1 ## Kendall
2 cor.test(ITSDaten$Temperatur, ITSDaten$Herzfrequenz, method = "kendall")
```

```
Kendall's rank correlation tau

data: ITSDaten$Temperatur and ITSDaten$Herzfrequenz
z = 6.0032, p-value = 1.935e-09
alternative hypothesis: true tau is not equal to 0
sample estimates:
tau
0.1826903
```

In allen drei Fällen bekommen wir eine signifikante Korrelation. Leider können wir in diesem Fall nur testen, ob die Korrelation signifikant von 0 verschieden bzw. positiv oder negativ ist und beispielsweise nicht, ob sich die Korrelation signifikant von einem vorgegebenen Wert unterscheidet. Dies kann im Fall der Pearson-Korrelation vom Konfidenzintervall abgeleitet werden. Im Fall der Spearman- und Kendall-Korrelation werden jedoch keine Konfidenzintervalle berechnet. Für das Konfidenzintervall der Spearman-Korrelation können wir auf die Funktion `SpearmanRho` des Paketes "DescTools" (Andri et al. (2022)) ausweichen, für das Konfidenzintervall der Kendall-Korrelation auf die Funktion `KendallTauB` des Paketes "DescTools" (Andri et al. (2022)).

```
1 ## Spearman
2 SpearmanRho(ITSData$Temperatur, ITSData$Herzfrequenz, conf.level = 0.95)

    rho      lwr.ci      upr.ci
0.2659957 0.1825635 0.3456245
```

```
1 ## Kendall
2 KendallTauB(ITSData$Temperatur, ITSData$Herzfrequenz, conf.level = 0.95)

    tau_b      lwr.ci      upr.ci
0.1826903 0.1254649 0.2399157
```

Wie wir bereits in Abschnitt 2.6.2 gesehen haben, entspricht die Spearman-Korrelation gerade der Pearson-Korrelation der Ränge. Auch dies können wir für die Berechnung eines entsprechenden Konfidenzintervalls nutzen und erhalten mit der Funktion `cor.test` das gleiche Ergebnis wie mit der Funktion `SpearmanRho`.

```
1 cor.test(rank(ITSData$Temperatur), rank(ITSData$Herzfrequenz))

Pearson's product-moment correlation

data: rank(ITSData$Temperatur) and rank(ITSData$Herzfrequenz)
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
```

Insgesamt liegt demnach zwar ein signifikanter, aber auch nur schwacher bis mäßiger Zusammenhang zwischen der maximalen Körpertemperatur und der maximalen Herzfrequenz vor, welcher kaum von praktischem Nutzen sein dürfte. Wir können mit Hilfe der Konfidenzintervalle zum Beispiel testen, ob die Korrelation signifikant kleiner als 0.5 ist, womit der Zusammenhang für viele praktische Anwendungen kaum von Nutzen ist. Wir wollen den Test zu einem Signifikanzniveau von $\alpha = 0.01$ durchführen. Entsprechend müssen wir dann 99% Konfidenzintervalle betrachten. Die die Funktionen `SpearmanRho`

und KendallTauB keine einseitigen Konfidenzintervalle berechnen, berechnen wir stattdessen zweiseitige $1 - 2\alpha$ Konfidenzintervall. Die Obergrenze dieser Intervalle entspricht dann gerade der Obergrenze des entsprechenden einseitigen Intervalls.

```
1 ## Pearson
2 cor.test(ITSData$Temperatur[-398], ITSData$Herzfrequenz[-398],
3         alternative = "less", conf.level = 0.99)$conf.int

[1] -1.0000000  0.3897994
attr(,"conf.level")
[1] 0.99
```



```
1 ## Spearman
2 SpearmanRho(ITSData$Temperatur, ITSData$Herzfrequenz, conf.level = 0.98)

rho      lwr.ci      upr.ci
0.2659957 0.1666303 0.3600128
```



```
1 cor.test(rank(ITSData$Temperatur), rank(ITSData$Herzfrequenz),
2         alternative = "less", conf.level = 0.99)$conf.int

[1] -1.0000000  0.3600128
attr(,"conf.level")
[1] 0.99
```



```
1 ## Kendall
2 KendallTauB(ITSData$Temperatur, ITSData$Herzfrequenz, conf.level = 0.98)

tau_b      lwr.ci      upr.ci
0.1826903 0.1147675 0.2506131
```

In allen drei Fällen könnten wir sagen, dass die Korrelation hoch signifikant ($\alpha = 0.01!$) kleiner als 0.5 ist.

Im letzten Teil dieses Abschnittes werden wir uns noch kurz mit Verteilungstests beschäftigen. Wir wollen herausfinden, ob wir davon ausgehen können, dass die vorliegenden Daten einer bestimmten vorgegebenen Verteilung P folgen oder nicht. Wir haben es demnach mit den folgenden Hypothesen zu tun

$$H_0: P \in \mathcal{P} \quad \text{versus} \quad H_1: P \notin \mathcal{P}$$

Vermutlich am häufigsten wird hierbei untersucht, ob Daten einer Normalverteilung folgen oder nicht. Auf diesen Fall werden wir uns hier beschränken; d.h., ob

$$P \in \mathcal{P} = \{\text{Norm}(\mu, \sigma^2) \mid \mu \in \mathbb{R}, \sigma \in (0, \infty)\} \quad (6.4)$$

Für diese Situation gibt es eine Vielzahl von möglichen Tests. Das liegt daran, dass die Alternative sehr groß (unendlichdimensional) ist und kein einzelner Test die Alternative vollständig abdecken kann. Entsprechend ist es auch unklar, welcher Test in dieser Situation die größte Power hat. In R implementierte

bekannte Tests sind etwa: Shapiro-Wilk Test, Kolmogorov-Smirnov Test bzw. Lilliefors Test, Anderson-Darling Test, Cramér-von Mises Test, Shapiro-Frankia Test, Jarque-Bera Test, D'Agostino Test, etc. Neben der Funktion `shapiro.test` wenden wir die Funktionen `LillieTest`, `CramerVonMisesTest` und `ShapiroFranciaTest` aus dem Paket "DescTools" (Andri et mult. al. (2022)) an, um einige bekannte Tests auf Normalverteilung für die maximale Körpertemperatur zu berechnen. Da diese Tests zum Teil auch deutlich auf Ausreißer reagieren, was durchaus berechtigt ist, wenn man diese Werte als korrekten Bestandteil des Datensatzes ansieht, lassen wir Patient 398 bei den Berechnungen unberücksichtigt.

```
1 ## Shapiro-Wilk Test
2 shapiro.test(ITSData$Temperatur[-398])
```

```
Shapiro-Wilk normality test

data: ITSData$Temperatur[-398]
W = 0.99189, p-value = 0.008013
```

```
1 ## Lilliefors (Kolmogorov-Smirnov) Test
2 LillieTest(ITSData$Temperatur[-398])
```

```
Lilliefors (Kolmogorov-Smirnov) normality test

data: ITSData$Temperatur[-398]
D = 0.048498, p-value = 0.007001
```

```
1 ## Cramer-von Mises Test
2 CramerVonMisesTest(ITSData$Temperatur[-398])
```

```
Cramer-von Mises normality test

data: ITSData$Temperatur[-398]
W = 0.16686, p-value = 0.01426
```

```
1 ## Shapiro-Francia Test
2 ShapiroFranciaTest(ITSData$Temperatur[-398])
```

```
Shapiro-Francia normality test

data: ITSData$Temperatur[-398]
W = 0.99131, p-value = 0.006149
```

In diesem Fall sind sich alle Tests einig und lehnen die Normalverteilungsannahme ab. Dies dürfte an der recht großen Stichprobe liegen und dass diese Tests bei entsprechend großen Stichproben auch kleinste Abweichungen von der vorgegebenen Verteilung mit großer Sicherheit erkennen können. Da die Abweichung eher klein ist, wie die Analyse in Abschnitt 5.2 zeigt, können diese Ergebnisse im Hinblick auf

den zentralen Grenzwertsatz sicher vernachlässigt werden und man kann ohne Einschränkung einen t-Test oder eine 1-Weg ANOVA durchführen, zumindest wenn man den offensichtlichen Ausreißer (Patient 398) weglässt. Dies wird auch dadurch bestätigt, dass t-Test, 1-Weg ANOVA, WMW-Test und Kruskal-Wallis Test bei Nichtberücksichtigung von Patient 398 durchweg sehr ähnliche Ergebnisse liefern.

Anmerkung:

Von der Verwendung von Verteilungstests ist in den meisten Fällen abzuraten, wie etwa als Vor-
tests (Rasch et al. (2011)). Insbesondere sollte auf die Anwendung des Kolmogorov-Smirnov Tests
verzichtet werden (Schoder et al. (2006); Ghasemi and Zahediasl (2012)). Diese Tests lösen in den
meisten Fällen das vorliegende Problem nicht bzw. unzureichend. Bei kleinen und mittleren Stich-
proben, bei denen man nur bedingt mit Grenzwertsätzen argumentieren kann, haben die Tests eine
geringe Power und erkennen zuverlässig nur deutliche Abweichungen von der vorgegebenen Vertei-
lung. Bei großen oder sehr großen Stichproben werden auch kleinste Abweichungen erkannt. Diese
spielen in den meisten Fällen aber keine wesentliche Rolle für die angewendeten statistischen Verfah-
ren, da entsprechende Grenzwertsätze greifen. Zusammenfassend ist meine Empfehlung daher, besser
die in Abschnitt 5.2 vorgestellten diagnostischen Plots für die Modellvalidierung und -diagnostik zu
verwenden anstelle verschiedenster statistischer Tests.

6.4 Übungsaufgaben

Erklären Sie jeweils die Schritte Ihrer Analyse und interpretieren Sie die Ergebnisse. Verwenden Sie für die Aufgaben 4–15, 17 und 18 den ITS-Datensatz und wählen Sie jeweils geeignete Funktionen für die Berechnungen aus.

1. In einer randomisierten und kontrollierten Studie wurde eine neue Behandlung zur Vermeidung der Übertragung von HIV untersucht. Es wurden keine signifikanten Unterschiede zwischen der neuen Behandlung und einer Kontrollgruppe gefunden. Das Verhältnis der neu aufgetretenen Erkrankungen zwischen den beiden Gruppen war 1.0. Das 95%-Konfidenzintervall für dieses Verhältnis war [0.63, 1.58]. Kann man auf der Basis dieser Ergebnisse sicher sein, dass die neue Behandlung keinen Effekt hat? Was könnte in diesem Kontext die Aussage: "Die Abwesenheit von Evidenz ist nicht die Evidenz für Abwesenheit" ("Absence of Evidence Is Not Evidence of Absence") bedeu-
ten?
2. Die Verantwortlichen der FranSO-Studie (Kemmler et al. (2017)) nahmen für die Fallzahlberech-
nung für den primären Endpunkt (Sarkopenie z-Score) eine Differenz der Mittelwerte von 1.0 an,
bei einer Standardabweichung von 1.4. Berechnen Sie die Fallzahl mit dem Student t-Test. Verwen-
den Sie ein Signifikanzniveau von 5% und eine Power von 80%. Wie stark erhöht sich die Fallzahl,
wenn Sie die Power auf 90% erhöhen? In die Studie wurden letztendlich 33 bzw. 34 Probanden
pro Gruppe eingeschlossen. Wie groß ist die Power der Studie, wenn Sie eine Fallzahl von 33 pro
Gruppe zugrundelegen?
3. Wir erwarten für eine neu zu planende Studie eine Differenz der Mittelwerte von 0.70 für den
gewählten primären Endpunkt, wobei wir annehmen, dass die Standardabweichung der Kontroll-

gruppe bei 0.75 und die Standardabweichung der Behandlungsgruppe bei 1.40 liegen wird. Verwenden Sie für die Fallzahlplanung den Welch t-Test mit einem Signifikanzniveau von 5% und einer Power von 90%. Wie verändert sich die Fallzahl, wenn Sie stattdessen den Hsu t-Test für den Fallzahlplanung heranziehen? Welche Fallzahl erhalten Sie für den WMW-Test? Untersuchen Sie außerdem, wie sich die Fallzahl verändert, wenn die Differenz der Mittelwerte tatsächlich etwas kleiner oder größer ist, als erwartet. Betrachten Sie hierfür das Interval [0.60, 0.80].

Verwenden Sie für die Berechnungen die Funktionen `power.welch.t.test`, `power.hsu.t.test` und `sim.size.wilcox.test` aus dem Paket "MKpower" (Kohl (2020c)). Im Fall des WMW-Tests verwenden Sie `sim.size.wilcox.test` in Kombination mit der Funktion `rnorm`. Weitere Einzelheiten zur Anwendung dieser Funktionen finden Sie auf den Hilfeseiten zu den Funktionen sowie in der Vignette des Paketes, welche Sie mit `vignette("MKpower")` aufrufen können.

4. Überprüfen Sie die Vermutung, dass auf einer ITS mehr Männer als Frauen behandelt werden. Formulieren Sie die Hypothesen und verwenden Sie sowohl einen exakten als auch einen asymptotischen Test, um dies zu überprüfen. Stellen Sie die Daten mit dem Testergebnis geeignet grafisch dar.
5. Untersuchen Sie, ob auf einer ITS Männer signifikant häufiger versterben als Frauen. Berechnen Sie sowohl einen exakten als auch einen asymptotischen Test, um dies zu überprüfen. Verwenden Sie als Ausgangspunkt die 2×2 Kontingenztafel, die Sie mittels

```
1 table(ITSDataen$Geschlecht, ITSDataen$Ergebnis == "Verstorben")
```

erhalten. Stellen Sie die Daten mit dem Testergebnis geeignet grafisch dar.

6. Untersuchen Sie, ob Patienten auf einer ITS im Mittel signifikant älter als 60 Jahre sind. Verwenden Sie den 1-Stichproben t-Test sowie den Wilcoxon Vorzeichen-Rangtest für die Analyse. Welcher Test erscheint Ihnen angebrachter? Stellen Sie die Daten mit dem Testergebnis geeignet grafisch dar.
7. Gehen Sie davon aus, dass sich die logarithmierten Bilirubinwerte von ITS-Patienten durch eine Normalverteilung beschreiben lassen.

```
1 ITSDataen$logBili <- log10(ITSDataen$Bilirubin)
```

Vergleichen Sie die mittlere log-Bilirubinkonzentration der ITS-Patienten mit und ohne Leberversagen mit Hilfe eines t-Tests. Erscheint Ihnen der klassische t-Test oder der Welch-t-Test für diese Situation angebrachter? Verwenden Sie in einem weiteren Schritt den WMW-Test und berechnen Sie den Test einmal mit und einmal ohne die Werte zu logarithmieren. Vergleichen Sie die beiden Ergebnisse. Was stellen Sie fest? Was ist der Grund dafür? Stellen Sie die Daten und die Testergebnisse geeignet grafisch dar.

8. Vergleichen Sie die mittlere Aufenthaltsdauer von Männern und Frauen auf der ITS. Verwenden Sie hierzu den WMW-Test. Stellen Sie die Daten und die Testergebnisse geeignet grafisch dar.

9. Gehen Sie davon aus, dass sich die maximale Körpertemperatur der ITS-Patienten durch eine Normalverteilung beschreiben lässt. Untersuchen Sie, ob sich die Mittelwerte der verschiedenen OP-Gruppen signifikant unterscheiden. Verwenden Sie für die Analyse die Welch 1-Weg ANOVA und lassen Sie Patient 398 unberücksichtigt. Falls Sie ein signifikantes Ergebnis erhalten, untersuchen Sie den Unterschied mit Hilfe geeigneter post hoc Tests genauer. Stellen Sie die Daten und die Testergebnisse geeignet grafisch dar.
10. Betrachten Sie nur Patienten mit einer Aufenthaltsdauer von mehr als einem Tag ($LOS > 1$).

```
1 ITSDaten.LOS2 ← ITSDaten[ITSDaten$LOS > 1, ]
```

Gehen Sie davon aus, dass sich die logarithmierten Bilirubinwerte von ITS-Patienten durch eine Normalverteilung beschreiben lassen. Verwenden Sie die Welch 1-Weg ANOVA um herauszufinden, ob die Mittelwerte der logarithmierten Bilirubinwerte für die OP-Gruppen unterschiedlich sind. Falls Sie ein signifikantes Ergebnis erhalten, untersuchen Sie den Unterschied mit Hilfe geeigneter post hoc Tests genauer. Stellen Sie die Daten und die Testergebnisse geeignet grafisch dar.

11. Gehen Sie davon aus, dass die maximal gemessene Herzfrequenz der ITS-Patienten näherungsweise durch eine Normalverteilung beschrieben werden kann. Untersuchen Sie mit Hilfe einer Welch-1-Weg ANOVA, ob sich die mittleren Herzfrequenzen der verschiedenen Ergebnis-Gruppen signifikant unterscheiden. Falls Sie, ein signifikantes Ergebnis erhalten, untersuchen Sie die Unterschiede mit Hilfe von post hoc Tests genauer. Stellen Sie die Daten und die Testergebnisse geeignet grafisch dar.
12. Vergleichen Sie die mittlere Aufenthaltsdauer der verschiedenen OP-Gruppen. Verwenden Sie hierzu den Kruskal-Wallis Test. Falls Sie, ein signifikantes Ergebnis erhalten, untersuchen Sie die Unterschiede mit Hilfe von post hoc Tests genauer. Stellen Sie die Daten und die Testergebnisse geeignet grafisch dar.
13. Betrachten Sie nur Patienten mit einer Aufenthaltsdauer von mehr als einem Tag ($LOS > 1$).

```
1 ITSDaten.LOS2 ← ITSDaten[ITSDaten$LOS > 1, ]
```

Wenden Sie den Kruskal-Wallis Test an, um herauszufinden, ob sich die Ergebnis-Gruppen hinsichtlich ihrer mittleren SAPS-II Scores signifikant unterscheiden. Falls Sie, ein signifikantes Ergebnis erhalten, untersuchen Sie die Unterschiede mit Hilfe von post hoc Tests genauer. Stellen Sie die Daten und die Testergebnisse geeignet grafisch dar.

14. Betrachten Sie die SAPS II Scores der ITS-Patienten und untersuchen Sie, ob sich die mittleren Werte für die verschiedenen OP-Gruppen unterscheiden. Verwenden Sie hierfür den Kruskal-Wallis Test. Sollten Sie ein signifikantes Ergebnis erhalten, so untersuchen Sie die Unterschiede mit Hilfe von post hoc Tests genauer. Stellen Sie die Daten und die Testergebnisse geeignet grafisch dar.

15. Betrachten Sie nur Patienten mit einer Aufenthaltsdauer von mehr als einem Tag ($\text{LOS} > 1$).

```
1 ITSDaten.LOS2 ← ITSDaten[ITSDaten$LOS > 1, ]
```

Wenden Sie den Kruskal-Wallis Test an, um herauszufinden, ob sich die Ergebnis-Gruppen hinsichtlich ihrer mittleren Alters signifikant unterscheiden. Falls Sie, ein signifikantes Ergebnis erhalten, untersuchen Sie die Unterschiede mit Hilfe von post hoc Tests genauer. Stellen Sie die Daten und die Testergebnisse geeignet grafisch dar.

16. Betrachten Sie den folgenden Datensatz:

```
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
```

Es wurden von 10 Subjekten Messungen an drei Zeitpunkten durchgeführt. Erzeugen Sie einen `data.frame` mit den Daten. Untersuchen Sie mit Hilfe einer repeated-measures 1-Weg ANOVA, ob es signifikante Unterschiede zwischen den Zeitpunkten gibt. Verwenden Sie außerdem ein mixed-effects 1-Weg ANOVA sowie den Quade- und den Friedman-Test für die Analyse. Falls sich signifikante Ergebnisse ergeben, führen Sie post hoc Tests mit gepaarten t-Tests und Wilcoxon Vorzeichen-Rangtests durch. Stellen Sie die Daten und die Testergebnisse geeignet grafisch dar.

17. Untersuchen Sie, ob sich die log-Bilirubinwerte von Männern und Frauen hinsichtlich ihrer Skala unterscheiden. Verwenden Sie dafür sowohl den F-Test als auch den Ansari-Bradley Test. Stellen Sie die Daten und die Testergebnisse geeignet grafisch dar.
18. Untersuchen Sie mit Hilfe eines geeigneten Tests, ob es eine signifikante Korrelation zwischen dem Alter und dem SAPS II Score gibt. Welcher Korrelationskoeffizient erscheint Ihnen für diese Situation als geeignet und warum? Stellen Sie die Daten und die Testergebnisse geeignet grafisch dar.
19. Verwenden Sie den `chem` Datensatz, der im Paket "MASS" (Venables and Ripley (2002)) enthalten ist und den Sie mit folgendem R Code laden können

```
1 library(MASS)
2 data(chem)
```

Verwenden Sie den Shapiro-Wilk Test sowie den Lilliefors (Kolmogorov-Smirnov), den Cramér-von Mises und den Shapiro-Francia Test aus dem Paket "DescTools" (Andri et mult. al. (2022)), um den Datensatz hinsichtlich der Normalverteilungsannahme zu untersuchen. Was stellen Sie fest? Überprüfen Sie die Normalverteilungsannahme zusätzlich mittels eines qq-Plots, den Sie zum Beispiel mit Hilfe der Funktionen `qqnorm` und `qqline` erzeugen können. Kommen Sie zum selben Ergebnis wie die Tests? Wiederholen Sie anschließend die Normalverteilungstests und den qq-Plot, wobei Sie die Beobachtung 17 von den Berechnungen ausschließen.

7 Multiples Testen

In diesem Kapitel wird das Problem des multiplen Testens besprochen. Im Einzelnen geht es um folgende Themen:

- Family-wise error rate (FWER)
- Bonferroni-Methode
- (Bonferroni-)Holm-Methode
- Simes-Hochberg-Methode
- Fallzahlplanung für multiple primäre Endpunkte
- False discovery rate (FDR)
- Benjamini-Hochberg-Methode
- Benjamini-Yekutieli-Methode
- Vulkanplot (volcano plot)

Der R Code für dieses Kapitel ist in der Datei `MultiplesTesten.Rmd` enthalten, welche Sie von meinem GitHub-Account herunterladen können (Link: <https://github.com/stamats/ESDR/blob/master/MultiplesTesten.Rmd>). Klicken Sie mit der rechten Maustaste auf Raw. Dann können Sie das Ziel speichern unter.... Am wenigsten Schwierigkeiten ergeben sich, wenn Sie meine R Markdown Dateien im selben Ordner wie die Daten, welche in den jeweiligen Kapiteln verwendet werden, speichern. Wir installieren zunächst die in diesem Kapitel benötigten Pakete.

```
1 BiocManager::install(c("multtest", "limma", "ComplexHeatmap"),  
2                      update = FALSE)  
3 install.packages("MKomics")
```

Achten Sie darauf, dass Sie die Pakete der vorhergehenden Kapitel 2–6 bereits installiert haben. Wir laden die Pakete, die wir in diesem Kapitel verwenden werden.

```
1 library(ggplot2)  
2 library(MKpower)  
3 library(multtest)  
4 library(MKomics)  
5 library(MKinfer)
```

Wie bereits in Abschnitt 2.4 erklärt, ist ein wiederholtes Ausführen von `library` unproblematisch.

7.1 Einführung

Im Kapitel 6 haben wir schon einige der Fallstricke beim statistischen Testen kennengelernt. Dort war jedoch die Annahme, dass man lediglich einen statistischen Test durchführt und das Ergebnis dieses Testes den Erfolg oder Misserfolg einer Studie beschreibt. Dies ist in der Praxis immer noch oft der Fall. Im Kontext von klinischen Studien spricht man etwa vom **primären Endpunkt**, der immer noch häufig durch einen einzigen Parameter beschrieben wird. Es häufen sich jedoch auch die Situationen, in denen dies nicht mehr als adäquat angesehen wird. Zum Beispiel sollten bei Studien zu Migränemedikamenten die folgenden vier Parameter herangezogen werden: Schmerz, Übelkeit, Lichtempfindlichkeit und Geräuschempfindlichkeit (Offen et al. (2007)). Entsprechend werden in immer mehr Studien nicht nur eine, sondern mehrere (primäre) Hypothesen gleichzeitig untersucht.

Außerdem wurden in den letzten Jahren und Jahrzehnten etwa in der Biologie und Medizin Hochdurchsatzverfahren entwickelt, mit denen es möglich ist, hunderte oder tausende von Parametern (z.B. Gene, Proteine oder Metaboliten) simultan zu messen. Auch im Internet oder von Smartphone-Apps werden eine Vielzahl von Variablen gleichzeitig ermittelt und deren Relevanz z.B. für die Platzierung von Werbeanzeigen oder das Einkaufsverhalten untersucht. Man spricht in diesem Zusammenhang heute auch von **“Big Data”**, die dann von einem Bioinformatiker oder **“Data Scientist”** ausgewertet und visualisiert werden.

In vielen Bereichen werden demnach heute nicht mehr nur eine, sondern $N > 1$ ($N \in \mathbb{N}$) Hypothesen gleichzeitig betrachtet und entsprechend müssen nicht nur ein, sondern N statistische Tests simultan durchgeführt werden. Wir wissen bereits aus Kapitel 6:

- für einen einzelnen Test haben wir einen Fehler 1. Art von $\alpha \in (0, 1)$
- Falls wir $N > 1$ Hypothesen testen und zwar jeweils zum gleichen Signifikanzniveau α , haben wir für jeden einzelnen der N Tests wieder einen Fehler 1. Art von α .

Wir wissen jedoch nicht, wie groß die Fehlerwahrscheinlichkeit für alle N Tests zusammen ist bzw. viel grundlegender, wie sich das Konzept des Fehlers 1. Art geeignet auf N Tests übertragen lässt. Im Folgenden wollen wir uns mit den aktuell zwei wichtigsten Ansätzen hierzu befassen.

7.2 Family-wise Error Rate (FWER)

Im Fall der **family-wise error rate (FWER)** betrachtet man die Wahrscheinlichkeit bei $N \in \mathbb{N}$ Tests mindestens einen Fehler 1. Art zu machen; d.h.,

$$\text{FWER} = P(\text{"mind. 1 Fehler 1. Art"}) \quad (7.1)$$

Im Fall $N = 1$ entspricht diese gerade dem Fehler 1. Art, also $\text{FWER} = \alpha$. Die exakte Berechnung dieser Wahrscheinlichkeit für $N > 1$ ist generell schwierig. Im Fall, dass die N Hypothesen (stochastisch) unabhängig voneinander sind und alle N statistischen Tests mit dem gleichen Signifikanzniveau

$\alpha \in (0, 1)$ durchgeführt werden, ist die exakte Berechnung aber möglich. Wir benutzen zunächst die Gegenwahrscheinlichkeit

$$\text{FWER} = P(\text{"mind. 1 Fehler 1. Art"}) = 1 - P(\text{"kein Fehler 1. Art"}) \quad (7.2)$$

wobei im Fall eines einzelnen Tests gilt

$$N = 1 : \quad P(\text{"kein Fehler 1. Art"}) = \text{Sensitivität} = 1 - \alpha \quad (7.3)$$

Im Fall von N (stochastisch) unabhängigen Tests folgt daher

$$P(\text{"kein Fehler 1. Art"}) = (1 - \alpha)^N \quad (7.4)$$

Damit ergibt sich

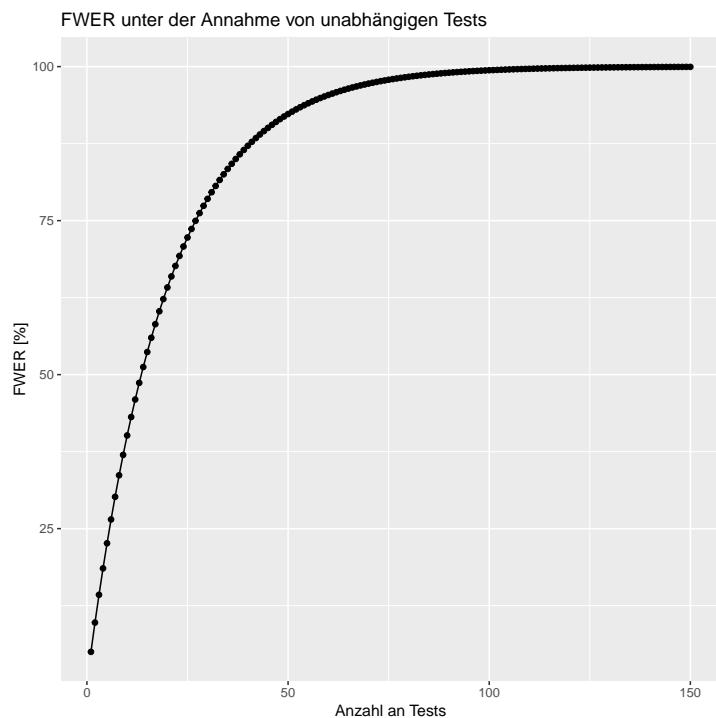
$$\text{FWER} = 1 - (1 - \alpha)^N \quad (7.5)$$

Wir stellen die Situation für $\alpha = 0.05$ grafisch dar.

```

1 N <- 1:150
2 FWER <- 100*(1 - (1-0.05)^N)
3 DF <- data.frame(N = N, FWER = FWER)
4 ggplot(DF, aes(x = N, y = FWER)) +
5   geom_point() + geom_line() + xlab("Anzahl an Tests") + ylab("FWER [%]") +
6   ggtitle("FWER unter der Annahme von unabhängigen Tests")

```



Die FWER steigt demnach recht schnell mit der Anzahl der Tests an und wir erhalten zum Beispiel

```
1 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

Leider ist die Annahme unabhängiger Hypothesen und Tests in der Praxis in den allermeisten Fällen nicht erfüllt. Hinzu kommt, dass die genaue Abhängigkeitstruktur zwischen den Hypothesen in der Praxis schwer zu beschreiben bzw. nicht bekannt ist. Damit ist die exakte Berechnung der FWER üblicher Weise nicht möglich und man kann sich nur Abschätzungen (durch obere Schranken) bedienen.

Die einfachste und auch allgemein gültigste Abschätzung folgt direkt aus der Eigenschaft der sogenannten Sub-Additivität von Wahrscheinlichkeitsmaßen

$$P\left(\bigcup_{i=1}^N A_i\right) \leq \sum_{i=1}^N P(A_i) \quad (7.6)$$

für beliebige (messbare) Ereignisse A_1, \dots, A_N . Hieraus ergibt sich mit

$$P(A_i) = P(\text{"Fehler 1. Art von Test } i\text{"}) = \alpha \quad \forall i = 1, \dots, N \quad (7.7)$$

die folgende Abschätzung für die FWER

$$\text{FWER} \leq \min\{N\alpha, 1\} \quad (7.8)$$

Da $N\alpha > 1$ für $N > \frac{1}{\alpha}$, wird das Minimum aus $N\alpha$ und 1 gebildet, um in jedem Fall eine gültige Wahrscheinlichkeit ($\in [0, 1]$) zu erhalten. Verwenden wir anstelle von α ein **adjustiertes Signifikanzniveau** $\tilde{\alpha} < \alpha$ für jeden Test, speziell $\tilde{\alpha} = \frac{\alpha}{N}$, so erhalten wir

$$\text{FWER} \leq \min\{N\tilde{\alpha}, 1\} = N\tilde{\alpha} = N \frac{\alpha}{N} = \alpha \quad (7.9)$$

Offensichtlich lässt sich diese Abschätzung auch auf die p-Werte p_1, \dots, p_n der Tests übertragen und es ist äquivalent, anstelle des Signifikanzniveaus die p-Werte zu adjustieren. Anstelle p_i mit $\tilde{\alpha}$ zu vergleichen, vergleicht man $\tilde{p}_i = \min\{N p_i, 1\}$ mit α .

Anmerkung:

In der Praxis werden beim multiplen Testen üblicherweise die p-Werte adjustiert. Die einzige wichtige Ausnahme bildet die Fallzahlplanung. Hier wird üblicherweise das Signifikanzniveau adjustiert.

Der oben skizzierte Ansatz die Abschätzung mittels der Sub-Additivität zu nutzen, ist als **Bonferroni-Methode** bekannt.

Algorithmus 7.1. Gegeben seien $N \in \mathbb{N}$ statistische Tests zum Signifikanzniveau $\alpha \in (0, 1)$ mit Nullhypotesen $H_{0,1}, \dots, H_{0,N}$ und entsprechenden Alternativen $H_{1,1}, \dots, H_{1,N}$, welche zu den p Werten p_1, \dots, p_N führen. Die **Bonferroni-Methode** ist das folgende ein-Schritt Verfahren:

Ersetze p_i durch $\tilde{p}_i := \min\{N p_i, 1\}$ ($\forall i = 1, \dots, N$).

Falls $\tilde{p}_i \leq \alpha$, wähle $H_{1,i}$, andernfalls behalte $H_{0,i}$ bei ($\forall i = 1, \dots, N$).

Anmerkung:

Die Methode von Bonferroni benötigt keine Kenntnis über die Abhängigkeitsstruktur der Hypothesen. Dies führt dazu, dass die Bonferroni-Methode sehr konservativ ist; d.h., man hat eine hohe Sicherheit, dass man keine falsch positiven Entscheidungen trifft. Jedoch werden dadurch umgekehrt auch viele korrekterweise positive Tests als nicht signifikant eingestuft (falsch negative Tests). Die Power dieser Methode ist demnach vergleichsweise klein bzw. der Fehler 2. Art vergleichsweise groß.

Es gibt neben der Bonferroni-Methode etwas weniger konservative sogenannte mehr-Schritt Verfahren, um die FWER zu kontrollieren. Für diese Verfahren müssen die p-Werte in einem ersten Schritt zunächst aufsteigend sortiert werden

$$p_{(1)} \leq p_{(2)} \leq \dots \leq p_{(N)}$$

Die wichtigste Alternative zur Bonferroni-Methode ist die Methode von Holm (1979), die auch als **(Bonferroni-)Holm-Methode** bekannt ist.

Algorithmus 7.2. Gegeben seien $N \in \mathbb{N}$ statistische Tests zum Signifikanzniveau $\alpha \in (0, 1)$ mit Nullhypotesen $H_{0,1}, \dots, H_{0,N}$ und zugehörigen Alternativen $H_{1,1}, \dots, H_{1,N}$, welche zu den p Werten p_1, \dots, p_N führen. Weiter seien $p_{(1)}, \dots, p_{(N)}$ die aufsteigend sortierten p Werte und $H_{0,(j)}$ und $H_{1,(j)}$ ($j = 1, \dots, N$) die zugehörigen Hypothesen. Die **(Bonferroni-)Holm-Methode** wird für $j = 1, \dots, N$ schrittweise beginnend mit dem kleinsten p-Wert durchgeführt. Der adjustierte p-Wert ergibt sich wie folgt:

Ersetze $p_{(j)}$ durch

$$\tilde{p}_{(j)} = \max_{k=1, \dots, j} \left\{ \min \left\{ (N - k + 1) p_{(k)}, 1 \right\} \right\} \quad (7.10)$$

Falls $\tilde{p}_{(j)} \leq \alpha$, wähle $H_{1,(j)}$, andernfalls behalte $H_{0,(j)}$ bei.

Es wäre auch möglich, einen frühzeitigen Stopp in den Algorithmus zu integrieren.

Bemerkung 7.3. Wird das erste Mal durch einen adjustierten p-Wert $\tilde{p}_{(j)}$ das Signifikanzniveau α überschritten, so gilt für alle folgenden adjustierten p-Werte $\tilde{p}_{(k)} > \alpha$ ($k > j$). Man könnte also auf die Berechnung weiterer adjustierter p-Werte verzichten. In der Praxis werden aber in der Regel davon unabhängig alle p-Werte adjustiert. Deshalb haben wir diesen möglichen frühzeitigen Stopp nicht in den obigen Algoithmus mit aufgenommen.

Anmerkung:

Die (Bonferroni-) Holm-Methode ist ein sogenanntes **step-down Verfahren**; d.h., ausgehend vom kleinsten p-Wert werden Schritt für Schritt größere p-Werte betrachtet. Man spricht von step-down, da die Teststatistik mit wachsendem p-Wert kleiner wird, man steigt also die Teststatistik hinab. Dieses Verfahren ist etwas weniger konservativ (d.h., hat größere Power) als die Bonferroni-Methode.

Wir betrachten ein einfaches Beispiel.

Beispiel 7.4. Gegeben seien die p-Werte von sechs Tests zu einem Signifikanzniveau von 5% mit Nullhypothesen $H_{0,1}, \dots, H_{0,6}$ und Alternativen $H_{1,1}, \dots, H_{1,6}$. Die zugehörigen p-Werte seien

$$p_1 = 0.004, p_2 = 0.011, p_3 = 0.039, p_4 = 0.012, p_5 = 0.001, p_6 = 0.480$$

Ohne eine Adjustierung der p-Werte würde man sich demnach bei einem Signifikanzniveau von 5% in fünf der sechs Fälle für die Alternative entscheiden; d.h., nur $H_{0,6}$ kann nicht abgelehnt werden.

(a) Die Bonferroni-Methode ergibt

$$\begin{aligned}\tilde{p}_1 &= 6p_1 = \mathbf{0.024} \\ \tilde{p}_2 &= 6p_2 = 0.066 \\ \tilde{p}_3 &= 6p_3 = 0.234 \\ \tilde{p}_4 &= 6p_4 = 0.072 \\ \tilde{p}_5 &= 6p_5 = \mathbf{0.006} \\ \tilde{p}_6 &= 6p_6 = 2.880 \Rightarrow 1.000\end{aligned}$$

Wir gleichen unsere Ergebnisse mit den Ergebnissen der Funktion `p.adjust` ab.

```
1 pval <- c(0.004, 0.011, 0.039, 0.012, 0.001, 0.480)
2 p.adjust(pval, method = "bonferroni")
```

```
[1] 0.024 0.066 0.234 0.072 0.006 1.000
```

d.h., „nur“ $H_{0,1}$ und $H_{0,5}$ können zum 5% Niveau abgelehnt werden.

(b) Wir stellen zunächst fest

$$p_5 \leq p_1 \leq p_2 \leq p_4 \leq p_3 \leq p_6$$

Wir berechnen basierend auf dieser Reihenfolge die adjustierten p-Werte mit Hilfe der (Bonferroni-)Holm-Methode

$$\begin{aligned}\tilde{p}_5 &= 6p_5 = \mathbf{0.006} \\ \tilde{p}_1 &= 5p_1 = \mathbf{0.020} \\ \tilde{p}_2 &= 4p_2 = \mathbf{0.044} \\ \tilde{p}_4 &= 3p_4 = 0.036 \Rightarrow \mathbf{0.044} \\ \tilde{p}_3 &= 2p_3 = 0.078 \\ \tilde{p}_6 &= 1p_6 = 0.480\end{aligned}$$

Im Fall von \tilde{p}_4 läge der berechnete Wert von 0.036 unter dem vorherigen Wert von \tilde{p}_2 . Dies ist nicht zulässig und wird durch die Maximumbildung in der Gleichung (7.10) verhindert. Wir kontrollieren die Ergebnisse wieder mit Hilfe der Funktion `p.adjust`, wobei eine Sortierung der p-Werte nicht nötig ist, sondern innerhalb der Funktion vorgenommen wird.

```

1 ## Sortierung nicht nötig
2 p.adjust(pval, method = "holm")

[1] 0.020 0.044 0.078 0.044 0.006 0.480

```

Damit können wir die Nullhypothesen $H_{0,5}$, $H_{0,1}$, $H_{0,2}$ und $H_{0,4}$ zum 5% Niveau ablehnen. Dies zeigt insbesondere, dass die (Bonferroni-) Holm-Methode weniger konservativ ist als die Bonferroni-Methode.

Eine weiteres bekanntes Adjustierungsverfahren und in einem gewissen Sinn ein Gegenstück zur (Bonferroni-)Holm-Methode ist die **Simes-Hochberg-Methode**.

Algorithmus 7.5. Gegeben seien $N \in \mathbb{N}$ **unabhängige** statistische Tests zum Signifikanzniveau $\alpha \in (0, 1)$ mit Nullhypothesen $H_{0,1}, \dots, H_{0,N}$ und zugehörigen Alternativen $H_{1,1}, \dots, H_{1,N}$, welche zu den p Werten p_1, \dots, p_N führen. Weiter seien $p_{(1)}, \dots, p_{(N)}$ die aufsteigend sortierten p Werte und $H_{0,(j)}$ und $H_{1,(j)}$ ($j = 1, \dots, N$) die zugehörigen Hypothesen. Die **Simes-Hochberg-Methode** besteht für $j = N, \dots, 1$ aus der folgenden Rechenvorschrift

Ersetze $p_{(j)}$ durch

$$\tilde{p}_{(j)} = \min_{k=j, \dots, N} \left\{ \min\{(N - k + 1)p_{(k)}, 1\} \right\} \quad (7.11)$$

Falls $\tilde{p}_{(j)} \leq \alpha$, wähle $H_{1,(j)}$, andernfalls behalte $H_{0,(j)}$ bei.

Es wäre auch wieder möglich, einen frühzeitigen Stopp in den Algorithmus zu integrieren.

Bemerkung 7.6. Erhält man beginnend mit dem größten p -Wert einen adjustierten p -Wert, der kleiner als das vorgegebene Signifikanzniveau ist, so könnte man das Verfahren auch stoppen, da alle folgenden adjustierten p -Werte auf jeden Fall höchstens noch kleiner sein werden. In der Praxis werden aber in der Regel davon unabhängig alle p -Werte adjustiert. Daher wurde dieser mögliche frühzeitige Stopp nicht in den obigen Algoithmus mit aufgenommen.

Anmerkung:

Die Simes-Hochberg-Methode ist ein sogenanntes **step-up Verfahren**. Man beginnt mit dem größten p -Wert und steigt damit die Teststatistik hinauf. Generell sind step-up Verfahren weniger konservativ als die entsprechenden step-down Verfahren. Dies trifft auch auf die Simes-Hochberg-Methode zu, die etwas weniger konservativ ist als ihr step-down Gegenstück die (Bonferroni-)Holm-Methode. Jedoch werden zusätzlich unabhängige Tests benötigt, was eine recht starke Voraussetzung darstellt, die in der Praxis meist nicht erfüllt ist.

Wir betrachten im folgenden Beispiel die Situation von Beispiel 7.4.

Beispiel 7.7. Wir müssen zusätzlich annehmen, dass die Tests unabhängig sind. Wir berechnen mit Hilfe des Simes-Hochberg Verfahrens die adjustierten p -Werte, wobei zur Erinnerung

$$p_5 \leq p_1 \leq p_2 \leq p_4 \leq p_3 \leq p_6$$

Wir erhalten

$$\begin{aligned}\tilde{p}_6 &= 1p_6 = 0.480 \\ \tilde{p}_3 &= 2p_3 = 0.078 \\ \tilde{p}_4 &= 3p_4 = \mathbf{0.036} \\ \tilde{p}_2 &= 4p_2 = 0.044 \Rightarrow \mathbf{0.036} \\ \tilde{p}_1 &= 5p_1 = \mathbf{0.020} \\ \tilde{p}_5 &= 6p_5 = \mathbf{0.006}\end{aligned}$$

Im Fall von \tilde{p}_2 läge der berechnete Wert von 0.044 über dem vorherigen Wert von \tilde{p}_4 . Dies ist nicht zulässig und wird durch die Minimumbildung in der Gleichung (7.11) verhindert. Wir kontrollieren unsere Berechnungen wieder mit Hilfe der Funktion `p.adjust`.

```
1 ## Sortieren nicht nötig
2 p.adjust(pval, method = "hochberg")
```

`[1] 0.020 0.036 0.078 0.036 0.006 0.480`

Damit können wir wie im Fall der (Bonferroni-)Holm-Methode die Nullhypotesen $H_{0,5}$, $H_{0,1}$, $H_{0,2}$ und $H_{0,4}$ zum 5% Niveau ablehnen, wobei aber \tilde{p}_2 und \tilde{p}_4 etwas kleiner sind als im Fall von (Bonferroni-)Holm. Dies zeigt insbesondere, dass das Simes-Hochberg-Verfahren etwas weniger konservativ ist als die (Bonferroni-)Holm-Methode. Jedoch wird dies durch die zusätzliche Voraussetzung von unabhängigen Tests teuer erkauft.

Anmerkung:

In der Praxis sollte in den meisten Fällen das (Bonferroni-)Holm-Verfahren zur Kontrolle der FWER herangezogen werden. Es ist weniger konservativ als das Verfahren von Bonferroni und kann im Unterschied zur Simes-Hochberg-Methode auch bei abhängigen Tests verwendet werden.

Abschließend betrachten wir noch ein Beispiel zur Fallzahlplanung in einer Situation mit multiplen primären Endpunkten.

Beispiel 7.8. Wir nehmen an, dass wir drei (primäre) Endpunkte simultan betrachten wollen, wobei zwei Gruppen miteinander verglichen werden sollen. Die Studie ist demnach nur dann erfolgreich, wenn wir für alle drei Endpunkte einen signifikanten Unterschied erhalten. Zur Vereinfachung nehmen wir weiter an, dass die drei Parameter, welche die primären Endpunkte widerspiegeln, einer Normalverteilung folgen. Die Gruppen können hierbei jeweils unterschiedliche Standardabweichungen aufweisen. Folglich entscheiden wir uns für den Welch t-Test für die Fallzahlplanung. Wir erwarten für die drei Endpunkte die folgenden Situationen:

Endpunkt 1: $|\mu_1 - \mu_2| = 0.5$, $\sigma_1 = 1$, $\sigma_2 = 1.2$

Endpunkt 2: $|\mu_1 - \mu_2| = 0.75$, $\sigma_1 = 1.5$, $\sigma_2 = 1.2$

Endpunkt 3: $|\mu_1 - \mu_2| = 1.0$, $\sigma_1 = 1.5$, $\sigma_2 = 1.75$

Wir wollen in der Studie eine Power von mindestens 90% erreichen. Außerdem soll die FWER maximal 5% betragen. Für die Fallzahlplanung ist es in solchen Fällen meist am einfachsten die Bonferroni-Methode heranzuziehen. Wir sollten demnach die Tests mit einem Fehler 1. Art von $\alpha = 0.05/3$ durchführen. Es ergibt sich

```
1 ## Endpunkt 1
2 power.welch.t.test(delta = 0.5, sd1 = 1.0, sd2 = 1.2,
3                     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 = 0.016666667
      power = 0.9
      alternative = two.sided

NOTE: n is number in *each* group
```

```
1 ## Endpunkt 2
2 power.welch.t.test(delta = 0.75, sd1 = 1.5, sd2 = 1.2,
3                      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 = 0.016666667
      power = 0.9
      alternative = two.sided

NOTE: n is number in *each* group
```

```
1 ## Endpunkt 3
2 power.welch.t.test(delta = 1.0, sd1 = 1.5, sd2 = 1.75,
3                      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

```

Wir erhalten für den ersten Endpunkt die höchste Fallzahl. Die Studie sollte folglich mit einer Fallzahl von 134 Probanden pro Gruppe durchgeführt werden, um die gewünschte Power von mindestens 90% zu erreichen und eine FWER von maximal 5% einzuhalten. Wir können mit der Funktion `power.welch.t.test` nicht nur die Fallzahl berechnen, sondern auch ausgehend von einer Fallzahl die zugehörige Power. Wir berechnen für eine Fallzahl von 134 die Power für den zweiten und dritten Endpunkt.

```

1 ## Endpunkt 2
2 power.welch.t.test(delta = 0.75, sd1 = 1.5, sd2 = 1.2,
3                     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

```

```

1 ## Endpunkt 3
2 power.welch.t.test(delta = 1.0, sd1 = 1.5, sd2 = 1.75,
3                     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

```

Im Fall von Endpunkt 2 und 3 führt eine Fallzahl von 134 Probanden pro Gruppe demnach auf eine Power von 98.2% und 99.5%. Diese Herangehensweise ignoriert einen möglichen Zusammenhang zwischen den

Variablen. Falls auch die Korrelationen zwischen den Variablen bekannt sind, könnte man eine Fallzahlplanung unter Verwendung von multivariaten Normalverteilungen durchführen, zum Beispiel mit Hilfe von Monte-Carlo Simulationen.

Anmerkung:

Es besteht das generelle Problem, dass alle bekannten Ansätze die FWER zu kontrollieren, recht konservativ sind, indem sie eine recht starke Adjustierung der p Werte enthalten. Daher führen diese Verfahren gerade in Situationen mit vielen Tests (> 10) dazu, dass viele wahre Unterschiede unentdeckt bleiben.

7.3 False Discovery Rate (FDR)

Die Konservativität der FWER bzw. der bekannten FWER-Verfahren hat dazu geführt, dass man für Situationen mit vielen oder sehr vielen simultanen Tests nach geeigneten Alternativen gesucht hat und immer noch sucht. Wir betrachten hierzu zunächst die möglichen **Testentscheidungen beim multiplen Testen**.

$i = 1, \dots, N$	Entscheidung für		Summe
	für H_{0i}	gegen H_{0i}	
H_{0i} richtig	U	V	$N_0 (= U + V)$
H_{0i} falsch	W	S	$N_1 (= W + S)$
Summe	$N - R (= U + W)$	$R (= V + S)$	$N (= U + V + W + S)$

Tabelle 7.1: Testentscheidungen beim multiplen Testen.

Offenbar gilt mit diesen Bezeichnungen: $\text{FWER} = P(V \geq 1)$. Es bietet sich an, anstelle der strikten Kontrolle der Wahrscheinlichkeit von falsch positiven Testergebnissen nach einer Kontrolle “im Mittel” (d.h. im Erwartungswert) zu suchen. Dies führte zum erwarteten Anteil falsch positiver Tests an allen positiven Tests, was als **false discovery rate (FDR)** bezeichnet wird

$$\text{FDR} = E\left(\frac{V}{R}\right) = E\left(\frac{V}{V + S}\right) \quad (7.12)$$

Allgemein gilt: $\text{FDR} \leq \text{FWER}$, was bedeutet, dass die FDR weniger konservativ als die FWER ist.

Das wohl am häufigsten verwendete Verfahren zur Berechnung der FDR adjustierten p-Werte ist die Methode von Benjamini and Hochberg (1995).

Algorithmus 7.9. Gegeben seien $N \in \mathbb{N}$ **unabhängige** bzw. **positiv abhängige** statistische Tests zum Signifikanzniveau $\alpha \in (0, 1)$ mit Nullhypotesen $H_{0,1}, \dots, H_{0,N}$ und Alternativen $H_{1,1}, \dots, H_{1,N}$, welche zu den p-Werten p_1, \dots, p_N führen. Weiter seien $p_{(1)}, \dots, p_{(N)}$ die aufsteigend sortierten p-Werte und $H_{0,(j)}$ und $H_{1,(j)}$ ($j = 1, \dots, N$) die zugehörigen Hypothesen. Die **Benjamini-Hochberg-Methode** besteht für $j = N, \dots, 1$ aus der folgenden Rechenvorschrift

Ersetze $p_{(j)}$ durch

$$\tilde{p}_{(j)} = \min_{k=j, \dots, N} \left\{ \min \left\{ \frac{N}{k} p_{(k)}, 1 \right\} \right\} \quad (7.13)$$

Falls $\tilde{p}_{(j)} \leq \alpha$, wähle $H_{1,(j)}$, andernfalls behalte $H_{0,(j)}$ bei.

Bemerkung 7.10. (a) Erhält man beginnend mit dem größten p -Wert einen adjustierten p -Wert, der kleiner als das vorgegebene Signifikanzniveau ist, so könnte man das Verfahren auch stoppen, da alle folgenden adjustierten p -Werte auf jeden Fall höchstens noch kleiner sein werden. In der Praxis werden aber in der Regel davon unabhängig alle p -Werte adjustiert. Daher wurde dieser mögliche frühzeitige Stop nicht in den obigen Algotithmus mit aufgenommen.

(b) Die genaue Definition der erlaubten positiven Abhängigkeit ist in Benjamini and Yekutieli (2001) beschrieben und erfasst, wie dort dargestellt wird, viele praktische Fragestellungen.

(c) Die Benjamini-Hochberg-Methode ist ein **step-up Verfahren** wie auch die Methode von Simes-Hochberg. Wir können diese beiden Verfahren auch direkt miteinander vergleichen, da sie sich nur im Korrekturfaktor unterscheiden. Im Folgenden wollen wir untersuchen, ob die Benjamini-Hochberg-Methode tatsächlich zu kleinern p -Werten führt

$$\begin{aligned} \frac{N}{k} p_{(k)} &\leq (N - k + 1)p_{(k)} \quad \text{für } k = 1, \dots, N \\ \frac{N}{k} &\leq (N - k + 1) \quad (p_{(k)} > 0) \\ N &\leq Nk - k^2 + k \\ 0 &\leq N(k - 1) - k^2 + k \\ 0 &\leq N(k - 1) - k(k - 1) \\ 0 &\leq (N - k)(k - 1) \end{aligned}$$

In der letzten Gleichung ist die rechte Seite mit Ausnahme der Fälle $k = 1$ und $k = N$ immer größer als 0; d.h., die adjustierten p -Werte sind im Fall von Benjamini-Hochberg in allen Schritten außer dem ersten und den letzten (dort sind sie gleich) kleiner als bei Simes-Hochberg.

Wir setzen Beispiel 7.4 fort.

Beispiel 7.11. Wir nehmen zusätzlich an, dass die Test unabhängig oder zumindest nur positiv abhängig sind. Zur Erinnerung

$$p_1 = 0.004, p_2 = 0.011, p_3 = 0.039, p_4 = 0.012, p_5 = 0.001, p_6 = 0.480$$

und

$$p_5 \leq p_1 \leq p_2 \leq p_4 \leq p_3 \leq p_6$$

Wir erhalten mit dem Benjamini-Hochberg-Verfahren

$$\begin{aligned}\tilde{p}_6 &= 1p_6 = 0.480 \\ \tilde{p}_3 &= \frac{6}{5}p_3 = \mathbf{0.0468} \\ \tilde{p}_4 &= \frac{6}{4}p_4 = \mathbf{0.018} \\ \tilde{p}_2 &= \frac{6}{3}p_2 = 0.022 \Rightarrow \mathbf{0.018} \\ \tilde{p}_1 &= \frac{6}{2}p_1 = \mathbf{0.012} \\ \tilde{p}_5 &= 6p_5 = \mathbf{0.006}\end{aligned}$$

Im Fall von \tilde{p}_2 läge der berechnete Wert von 0.022 über dem vorherigen Wert von \tilde{p}_4 . Dies ist nicht zulässig und wird durch die Minimumbildung in der Gleichung (7.13) verhindert. Auch diese Berechnungen können wir wieder mit Hilfe der Funktion `p.adjust` kontrollieren.

```
1 ## Sortieren nicht nötig
2 p.adjust(pval, method = "fdr")
```

```
[1] 0.0120 0.0180 0.0468 0.0180 0.0060 0.4800
```

Damit können wir die Nullhypotesen $H_{0,5}$, $H_{0,1}$, $H_{0,2}$, $H_{0,4}$ und auch $H_{0,3}$ zum 5% Niveau ablehnen. Die adjustierten p-Werte sind auffallend kleiner als im Fall der FWER Methoden.

Anmerkung:

In der Praxis kommt zur Kontrolle der FDR in den allermeisten Fällen das Verfahren von Benjamini und Hochberg (1995) zum Einsatz. Ist unklar, ob die Annahme von positiv abhängigen Tests erfüllt ist, kann man auf die konservativeren **Benjamini-Yekutieli-Methode** ausweichen, die in Benjamini und Yekutieli (2001) als Modifikation der Benjamini-Hochberg-Methode eingeführt wurde und die in jedem Fall die FDR kontrolliert.

Im folgenden Beispiel werden wir die vorgestellten Verfahren auf einen realen und sehr bekannten Genexpressionsdatensatz anwenden.

Beispiel 7.12. Wir verwenden die Genexpressionsdaten von Golub et al. (1999), welche im Bioconductor Paket "multtest" (Pollard et al. (2012)) enthalten sind. Es handelt sich dabei um eine der ersten Publikationen, in denen die Genexpression zur Vorhersage (Diagnose) einer Erkrankung verwendet wurde. Und zwar wurden Proben von Patienten mit akuter lymphatischer Leukämie (ALL) und von Patienten mit akuter myeloischer Leukämie (AML) mittels ihrer molekularen Gensignatur klassifiziert. Das Paket "multtest" enthält verschiedene Verfahren, die zur Adjustierung von p-Werten eingesetzt werden können. Wir laden den Datensatz `golub`.

```
1 data(golub)
2 str(golub)
```

```

num [1:3051, 1:38] -1.458 -0.752 0.457 3.135 2.766 ...
- attr(*, "dimnames")=List of 2
..$ : NULL
..$ : NULL

```

Zur Verdeutlichung wandeln wir die vorhandene Variable `golub.cl` in einen Faktor um.

```

1 golub.cl <- factor(golub.cl, labels = c("ALL", "AML"))
2 table(golub.cl)

```

```

golub.cl
ALL AML
27 11

```

Wir wenden den Welch t-Test an, um die beiden Leukämiearten zu vergleichen. Hierzu implementieren wir zunächst eine einfache Funktion `ttest`, die nur den p-Wert des Tests zurückgibt. Mit Hilfe der Funktion `apply` wenden wir diese Funktion dann auf die Zeilen (`MARGIN = 1`) des Datensatzes an.

```

1 ttest <- function(x, g) t.test(x~g)[[ "p.value" ]]
2 p.werte <- apply(X = golub, MARGIN = 1, FUN = ttest, g = golub.cl)

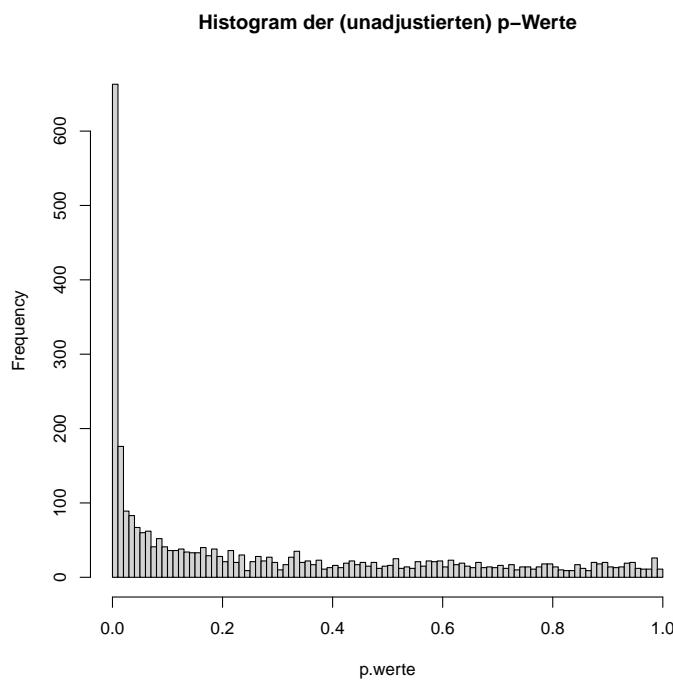
```

Wir stellen die (unadjustierten) p-Werte in Form eines Histogramms graphisch dar.

```

1 hist(p.werte, nclass = 101,
2      main = "Histogram der (unadjustierten) p-Werte")

```



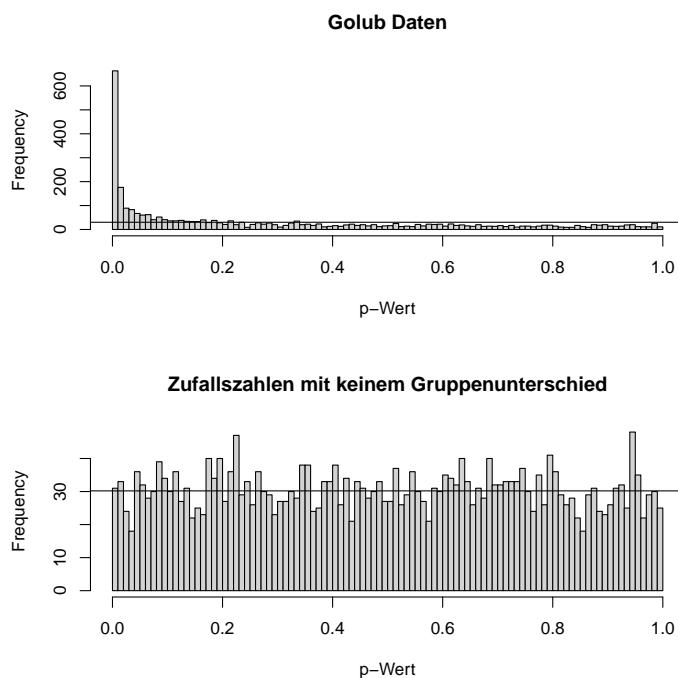
Das Histogramm weist mit einem deutlichen Peak bei kleinen p-Werten stark darauf hin, dass es viele signifikante Unterschiede zwischen den beiden Gruppen gibt. Um dies zu verdeutlichen, erzeugen wir

im nächsten Schritt (Pseudo-)Zufallszahlen von zwei Gruppen, zwischen denen es keine Unterschiede gibt und berechnen die zugehörigen p-Werte. Wir verwenden einen Zufallsdatensatz, der die gleiche Dimension besitzt, wie der Genexpressionsdatensatz.

```
1 M ← matrix(rnorm(nrow(golub)*ncol(golub)), nrow = nrow(golub))
2 p.werte.vgl ← apply(M, 1, ttest, g = golub.cl)
```

Wir vergleichen die Histogramme der p-Werte der beiden Analysen miteinander.

```
1 par(mfrow = c(2, 1))
2 hist(p.werte, nclass = 101, xlab = "p-Wert", main = "Golub Daten")
3 abline(h = 30.2)
4 hist(p.werte.vgl, nclass = 101, xlab = "p-Wert",
      main = "Zufallszahlen mit keinem Gruppenunterschied")
5 abline(h = 30.2)
```



Im Fall, dass keine signifikanten Unterschiede vorliegen, folgen die p-Werte des Welch t-Tests einer uniformen Verteilung (Gleichverteilung). Insbesondere ist damit per Zufall zu erwarten, dass wir in etwa 5% der Fälle einen p-Wert kleiner als 5% – d.h., falsch positive Testergebnisse – erhalten.

```
1 ## zu erwartende Anzahl
2 0.05*nrow(M)
```

```
[1] 152.55
```

```
1 ## tatsächliche Anzahl
2 sum(p.werte.vgl < 0.05)
```

```
[1] 142
```

Wir sehen, dass die erwartete Anzahl falsch positiver Tests im Fall des Datensatzes mit Zufallszahlen sehr gut mit der tatsächlichen Anzahl übereinstimmt. Auch im Fall des realen Datensatzes, der die gleiche Dimension besitzt, muss man von einer entsprechenden Anzahl an falsch positiven Testergebnissen ausgehen. Insgesamt erhält man die folgende Anzahl positiver Tests.

```
1 sum(p.werte < 0.05)
```

```
[1] 1078
```

Es verbleibt die wichtige Frage: Wie viele dieser 1078 positiven Testergebnisse sind echt positiv und wie viele falsch positiv?

Die exakte Antwort auf diese Frage ist in der Praxis nicht bekannt. Wir berechnen zur näherungsweisen Beantwortung dieser Frage die adjustierten p-Werte, wobei wir die verschiedenen Methoden, die in der Funktion `mt.rawp2adjp` im Paket "multtest" (Pollard et al. (2012)) implementiert sind, anwenden.

```
1 p.werte.adj ← mt.rawp2adjp(p.werte)
```

Wir berechnen für jedes der Verfahren die Anzahl der p-Werte, die kleiner als 5% sind.

```
1 colSums(p.werte.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		

Das erste Ergebnis stammt von den unadjustierten p-Werten. Bei den Verfahren Bonferroni bis SidakSD handelt es sich um Verfahren, welche die FWER kontrollieren. Wir sehen, dass wir lediglich wenig mehr als 100 der mehr als 1000 signifikanten Gene als echt positiv ansehen dürfen, wenn wir die FWER kontrollieren wollen. In dieser Gruppe von Genen beträgt dann aber auch die Wahrscheinlichkeit von einem oder mehr falsch positiven Tests nur maximal 5%.

Die verbleibenden vier Ergebnisse stammen von Verfahren, welche die FDR kontrollieren. Im Fall der Benjamini-Hochberg-Methode (BH) erhalten wir nahezu 700 Gene, die wir nach der Adjustierung weiterhin als signifikant ansehen können. Der erwartete Anteil von falsch positiven Testergebnissen unter diesen knapp 700 Genen beträgt demnach maximal 5%, was ungefähr $0.05 \times 695 \approx 35$ Genen entspricht. Bei den ursprünglichen 1078 Genen müssen wir davon ausgehen, dass dieser Anteil bei ca. $\frac{152.55}{1078} \approx 14\%$ liegt. Anstelle von ca. jedem siebten positiven Ergebnis in der unadjustierten Liste von signifikanten Genen ist in der reduzierten BH-Liste nur noch ca. jedes 20-ste positive Ergebnis falsch positiv.

Interessant ist auch das Ergebnis der Benjamini-Yekutieli-Methode (BY), für welche keine zusätzliche Annahme über die Form der Abhängigkeit nötig ist. Wir sehen, dass dieses Verfahren zwischen den FWER Methoden und dem Verfahren von Benjamini-Hochberg rangiert.

Für weitere Einzelheiten zu den nicht im Detail besprochenen Verfahren verweisen wir auf die Hilfeseite der Funktion `mt.rawp2adjp` und die dort angegebene Literatur.

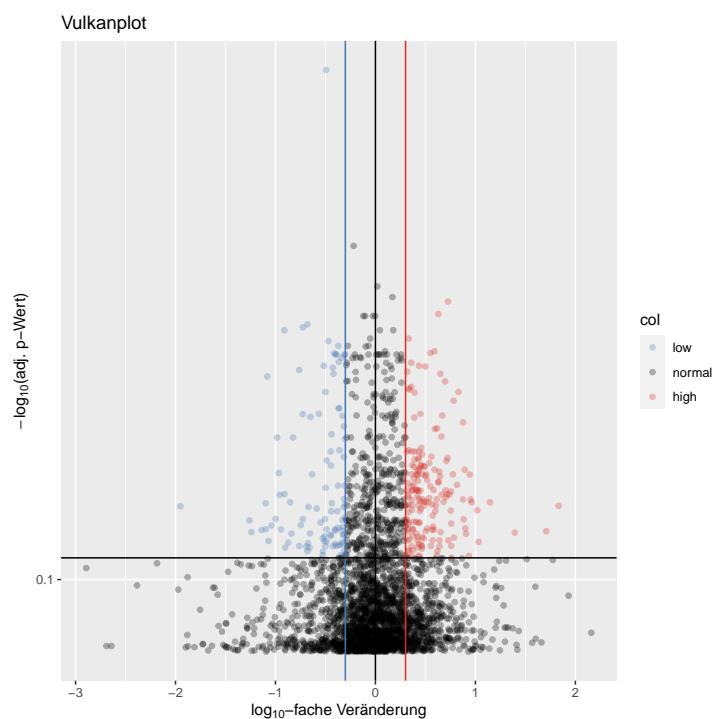
Zum Abschluss wollen wir noch auf das Hauptergebnis der Phase I des Microarray Quality Control (MAQC-I) Projektes hinweisen. Ein Projekt, das von der FDA (U.S. Food and Drug Administration) initiiert wurde. In diesem Projekt, in dem es um die Reproduzierbarkeit der Ergebnisse von Genexpressionsanalysen mit Hilfe sogenannter Mikroarrays ging, zeigte sich, dass die Reproduzierbarkeit der Resultate erhöht werden kann, wenn man adjustierte p-Werte mit der log-fachen Veränderung kombiniert; siehe MAQC-Consortium (2006). Wir setzen das obige Beispiel fort und stellen diese Kombination mit Hilfe eines sogenannten **Vulkanplot (volcano plot)** graphisch dar.

Beispiel 7.13. Für die Erzeugung des Vulkanplots müssen wir zunächst die log-fachen Veränderungen berechnen. Wir verwenden hierzu die Funktion `pairwise.logfc` aus dem Paket "MKomics" (Kohl (2020b)).

```
1 logFC ← apply(golub, 1, pairwise.logfc, g = golub.cl)
```

Den Vulkanplot können wir zum Beispiel mit der Funktion `volcano` aus dem Paket "MKinfer" (Kohl (2022b)) erzeugen. Neben der log-fachen Veränderung werden wir die mittels der Benjamini-Hochberg-Methode adjustierten p-Werte grafisch darstellen. Es ist bei Genexpressionsanalysen üblich, einen adjustierten p-Wert von weniger als 0.05 als signifikant und eine Veränderung um das 2-Fache oder mehr als relevant einzuschätzen. Die Werte des Datensatzes sind \log_{10} -transformiert. Folglich liegt die Grenze für die log-fache Veränderung bei $\pm \log_{10}(2) \approx \pm 0.3$.

```
1 volcano(x = logFC, pval = p.werte.adj[["adjp"]][, "BH"], effect.low = -log10(2),
2   effect.high = log10(2), alpha = 0.3,
3   xlab = expression(paste(log[10], "-fache Veränderung")),
4   ylab = expression(paste(-log[10], "(adj. p-Wert)")),
5   title = "Vulkanplot")
```



Wie im Plot zu sehen ist, werden die p-Werte für die graphische Darstellung zunächst \log_{10} -transformiert. Dies führt dazu, dass aus den sehr kleinen p-Werten große negative Zahlen werden. Durch die zusätzliche Änderung des Vorzeichens, werden aus diesen großen negativen Zahlen dann große positive Zahlen; d.h., die kleinen p-Werte befinden sich in der Abbildung oben. Die Grenze für die transformierten p-Werte liegt damit bei $-\log_{10}(0.05) \approx 1.3$. Die wichtigsten Gene befinden sich demnach oben links (blau) und oben rechts (rot) im Plot.

Mit Hilfe der Funktionen `expression` und `paste` lassen sich in der Beschriftung von Plots auch einfache mathematische Ausdrücke verwenden. Eine genauere Beschreibung findet man auf der Hilfeseite `plotmath`.

Anmerkung:

Das Ergebnis des MAQC-I Projekts ist aus statistischer Sicht nicht sehr überraschend, da die log-fache Veränderung in diesem Fall ein Maß für den Effekt ist und entsprechend auch die Relevanz der Ergebnisse repräsentiert. Unabhängig von Genexpressiondaten kann man davon ausgehen, dass im Fall einzelner wie auch multipler Tests eine Kombination aus Signifikanz und Relevanz die Wahrscheinlichkeit für echt positive Testergebnisse und damit auch deren Reproduzierbarkeit erhöht.

7.4 Übungsaufgaben

Erklären Sie jeweils die Schritte Ihrer Analyse und interpretieren Sie die Ergebnisse.

1. Adjustieren Sie die folgenden p-Werte per Hand.

0.001, 0.760, 0.550, 0.003, 0.002, 0.271, 0.005, 0.007, 0.210, 0.008

Verwenden Sie die Methoden von

- a) Bonferroni
- b) Bonferroni-Holm
- c) Simes-Hochberg
- d) Benjamini-Hochberg

Vergleichen Sie Ihre Ergebnisse mit den Ergebnissen, die Sie mit Hilfe der Funktion `p.adjust` erhalten. Interpretieren Sie die Ergebnisse der verschiedenen Methoden und vergleichen Sie diese miteinander. Bestätigt sich, dass die FWER-Methoden konservativer als die FDR-Methode sind?

2. Adjustieren Sie die folgenden p-Werte per Hand.

0.001, 0.820, 0.950, 0.001, 0.001, 0.011, 0.012, 0.001, 0.009, 0.001

Verwenden Sie die Methoden von

- a) Bonferroni
- b) Bonferroni-Holm

- c) Simes-Hochberg
- d) Benjamini-Hochberg

Vergleichen Sie Ihre Ergebnisse mit den Ergebnissen, die Sie mit Hilfe der Funktion `p.adjust` erhalten. Wie verhalten sich die verschiedenen Adjustierungsmethoden bei mehreren gleichen p-Werten? Vergleichen und interpretieren Sie die Ergebnisse der verschiedenen Methoden.

3. Führen Sie in Anlehnung an die Studie von Dodick et al. (2019) eine Fallzahlplanung durch. Es wurden darin zwei primäre Endpunkte betrachtet. Zum einen der Anteil der Patienten, der nach zwei Stunden schmerzfrei ist und zum zweiten der Anteil der Patienten, der nach zwei Stunden frei von dem darüber hinaus lästigsten Symptom (Geräuschempfindlichkeit, Geruchsempfindlichkeit, Übelkeit) ist. Laut dem Studienprotokoll, welches unter <https://clinicaltrials.gov/ct2/show/NCT02828020> zu finden ist, waren die Annahmen für die Fallzahlplanung

Endpunkt 1: $p_1 = 0.1, p_2 = 0.24$

Endpunkt 2: $p_1 = 0.267, p_2 = 0.377$

Führen Sie eine Fallzahlberechnung mit Hilfe der Funktion `power.prop.test` aus dem Paket "stats" (R Core Team (2022a)) durch. Die FWER soll maximal 5% betragen, die Power mindestens 90%. Sehen Sie damit die ursprünglich geplante Fallzahl von 450 pro Gruppe als gerechtfertigt an?

4. Verwenden Sie den ITS-Datensatz und vergleichen Sie die Patienten mit und ohne Leberversagen hinsichtlich Geschlecht, Alter, maximale Herzfrequenz, maximale Körpertemperatur, SAPS-II Score, Aufenthaltsdauer und Outcome (Ergebnis). Verwenden Sie hierfür jeweils geeignete Tests. Speichern Sie die p-Werte und adjustieren Sie diese mit der Methode von Holm. Verwenden Sie hierfür die Funktion `p.adjust`. Hinsichtlich welcher Variablen erhalten Sie auch nach der Adjustierung der p-Werte ein signifikantes Ergebnis?
5. Verwenden Sie den ITS-Datensatz und vergleichen Sie männliche und weibliche Patienten Alter, maximale Herzfrequenz, maximale Körpertemperatur, SAPS-II Score, Leberversagen, Aufenthaltsdauer und Outcome (Ergebnis). Verwenden Sie hierfür jeweils geeignete Tests. Speichern Sie die p-Werte und adjustieren Sie diese mit der Methode von Holm. Verwenden Sie hierfür die Funktion `p.adjust`. Hinsichtlich welcher Variablen erhalten Sie auch nach der Adjustierung der p-Werte ein signifikantes Ergebnis?
6. Verwenden Sie den ITS-Datensatz und vergleichen Sie die Patienten, die verstorben sind, mit den Patienten, die nicht verstorben sind, hinsichtlich Geschlecht, Alter, maximale Herzfrequenz, maximale Körpertemperatur, SAPS-II Score, Leberversagen und Aufenthaltsdauer.

```
1 ITSDaten$Verstorben <- as.integer(ITSDaten$Ergebnis == "Verstorben")
```

Verwenden Sie hierfür jeweils geeignete Tests. Speichern Sie die p-Werte und adjustieren Sie diese mit der Methode von Holm. Verwenden Sie hierfür die Funktion `p.adjust`. Hinsichtlich welcher Variablen erhalten Sie auch nach der Adjustierung der p-Werte ein signifikantes Ergebnis?

7. Laden Sie die Datei `normDaten.RData` von meinem GitHub-Account herunter (Link: <https://github.com/stamats/ESDR/raw/master/normDaten.RData>).
 - a) Wenden Sie den Welch t-Test auf die Zeilen des Datensatzes `normDaten` an, wobei die beiden Gruppen durch den Faktor `gruppe` gegeben sind. Beide Variablen sind in der Datei `normData.RData` enthalten. Zeichnen Sie ein Histogramm der (unadjustierten) p-Werte. Erhalten Sie mehr signifikante Ergebnisse als per Zufall zu erwarten sind?
 - b) Berechnen Sie die adjustierten p-Werte mit Hilfe der Funktion `mt.rawp2adjp` des Paketes "multtest" (Pollard et al. (2012)). Vergleichen Sie die Ergebnisse der verschiedenen Adjustierungsverfahren, indem Sie berechnen, wie viele der adjustierten p-Werte jeweils kleiner als 0.05 sind.
 - c) Die Daten in `normDaten` sind \log_2 -transformiert. Berechnen Sie für die Zeilen die \log_2 -fachen Veränderungen und stellen Sie diese zusammen mit den Benjamini-Hochberg adjustierten p-Werten in einem Vulkanplot graphisch dar.

Softwareversionen

Für die Erstellung des Buches wurden die folgenden Softwareversionen verwendet:

- R version 4.2.2 Patched (2022-11-12 r83339), 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 2.58.0, BiocGenerics 0.44.0, boot 1.3-28, coin 1.4-2, datarium 0.1.0, DescTools 0.99.47, distr 2.9.1, distr6 1.6.11, distrEx 2.9.0, distrMod 2.9.0, evd 2.3-6.1, exactRankTests 0.8-35, ggplot2 3.4.0, ggpibr 0.5.0, ggscli 2.9, gridExtra 2.3, knitr 1.41, MASS 7.3-58.1, MKclass 0.3, MKdescr 0.8, MKinfer 0.9, MKomics 0.7, MKpower 0.5, multtest 2.54.0, qqplotr 0.0.5.99999, RandVar 1.2.1, RColorBrewer 1.1-3, rmx 0.8, RobAStBase 1.2.3, RobExtremes 1.2.0, RobLox 1.2.0, robustbase 0.95-0, ROptEst 1.3.1, rrcov 1.7-2, 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 3.3-1, arrangements 1.1.9, backports 1.4.1, benchmarkme 1.0.8, benchmarkmeData 1.0.4, bitops 1.0-7, broom 1.0.1, car 3.1-1, carData 3.0-5, caTools 1.18.2, cellranger 1.1.0, checkmate 2.1.0, circlize 0.4.15, class 7.3-20, cli 3.4.1, clue 0.3-63, cluster 2.1.4, codetools 0.2-18, colorspace 2.0-3, compiler 4.2.2, ComplexHeatmap 2.14.0, crayon 1.5.2, data.table 1.14.6, DEoptimR 1.0-11, dictionar6 0.1.3, digest 0.6.30, doParallel 1.0.17, dplyr 1.0.10, e1071 1.7-12, ellipsis 0.3.2, evaluate 0.18, Exact 3.2, expint 0.1-8, 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 2.6.6, GlobalOptions 0.1.2, glue 1.6.2, gmp 0.6-8, grid 4.2.2, gtable 0.3.1, httr 1.4.4, IRanges 2.32.0, iterators 1.0.14, labeling 0.4.2, lattice 0.20-45, libcoin 1.0-9, lifecycle 1.0.3, limma 3.54.0, lmom 2.9, magrittr 2.0.3, Matrix 1.5-3, matrixStats 0.63.0, matrixTests 0.1.9.1, memuse 4.2-2, mgcv 1.8-41, modeltools 0.2-23, multcomp 1.4-20, munsell 0.5.0, mvtnorm 1.1-3, nlme 3.1-160, ooplah 0.2.0,

opdisDownsampling 0.8.2, param6 0.2.4, pcaPP 2.0-3, pillar 1.8.1, pkgconfig 2.0.3, png 0.1-7, pracma 2.4.2, 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 0.7.1, rstudioapi 0.14, S4Vectors 0.36.0, sandwich 3.0-2, set6 0.2.5, shape 1.4.6, splines 4.2.2, stringi 1.7.8, stringr 1.4.1, TH.data 1.1-1, tibble 3.1.8, tidyverse 1.2.1, tidyselect 1.2.0, tools 4.2.2, twosamples 2.0.0, utf8 1.2.2, vctrs 0.5.1, withr 2.5.0, xfun 0.35, zoo 1.8-11

Literaturverzeichnis

- Almeida, A., Loy, A., and Hofmann, H. (2018). *ggplot2 Compatible Quantile-Quantile Plots in R*. (Zitiert auf den Seiten 138, 140 und 143)
- 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. (Zitiert auf Seite 210)
- Andri et mult. al., S. (2022). *DescTools: Tools for Descriptive Statistics*. R package version 0.99.46. (Zitiert auf den Seiten 19, 21, 37, 38, 45, 49, 50, 52, 218, 254, 256 und 260)
- 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. (Zitiert auf Seite 153)
- Auguie, B. (2017). *gridExtra: Miscellaneous Functions for "Grid"Graphics*. R package version 2.3. (Zitiert auf den Seiten 151 und 249)
- 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. (Zitiert auf den Seiten 271 und 273)
- Benjamini, Y. and Yekutieli, D. (2001). The control of the false discovery rate in multiple testing under dependency. *Ann. Stat.*, 29. (Zitiert auf den Seiten 272 und 273)
- 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. (Zitiert auf Seite 154)
- Box, G. and Draper, N. (1987). *Empirical model-building and response surfaces*. Wiley series in probability and mathematical statistics: Applied probability and statistics. Wiley. (Zitiert auf Seite 10)
- Brier, G. W. (1950). Verification of forecasts expressed in terms of probability. *Mon. Wea. Rev.*, 78:1–3. (Zitiert auf Seite 159)
- 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. (Zitiert auf den Seiten 154 und 155)
- Broman, K. W. and Woo, K. H. (2018). Data organization in spreadsheets. *The American Statistician*, 72(1):2–10. (Zitiert auf Seite 12)
- Campbell, I. (2007). Chi-squared and Fisher-Irwin tests of two-by-two tables with small sample recommendations. *Stat Med*, 26(19):3661–3675. (Zitiert auf Seite 219)

- Canty, A. and Ripley, B. D. (2021). *boot: Bootstrap R (S-Plus) Functions*. R package version 1.3-28. (Zitiert auf den Seiten 128, 166, 167, 176 und 179)
- Chambers, J. (2000). Stages in the Evolution of S. <http://ect.bell-labs.com/sl/S/history.html> [Last access 29.08.2015]. (Zitiert auf den Seiten 1 und 2)
- Chambers, J. (2008). *Software for Data Analysis: Programming with R*. Springer. (Zitiert auf den Seiten 1 und 2)
- Chernick, M. R. and LaBudde, R. A. (2011). *An Introduction to Bootstrap Methods with Applications to R*. Wiley Publishing, 1st edition. (Zitiert auf den Seiten 166, 167 und 168)
- Cohen, J. (1994). The earth is round ($p < .05$). 49. (Zitiert auf Seite 210)
- Colquhoun, D. (2014). An investigation of the false discovery rate and the misinterpretation of p-values. *R Soc Open Sci*, 1(3):140216. (Zitiert auf Seite 209)
- 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. (Zitiert auf Seite 91)
- 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]. (Zitiert auf Seite 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]. (Zitiert auf Seite 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]. (Zitiert auf Seite 2)
- DataCamp Inc. (2022). Search all 25.258 R packages on CRAN and Bioconductor. <https://www.rdocumentation.org/> [Last access 2022 Oct 08]. (Zitiert auf Seite 3)
- Davison, A. C. and Hinkley, D. V. (1997). *Bootstrap Methods and Their Applications*. Cambridge University Press, Cambridge. ISBN 0-521-57391-2. (Zitiert auf Seite 167)
- Delignette-Muller, M. L. and Dutang, C. (2015). fitdistrplus: An R package for fitting distributions. *Journal of Statistical Software*, 64(4):1–34. (Zitiert auf Seite 134)
- Dickersin, K., Chan, S., Chalmersx, T., Sacks, H., and Smith, H. (1987). Publication bias and clinical trials. *Controlled Clinical Trials*, 8(4):343–353. (Zitiert auf Seite 209)
- 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. (Zitiert auf Seite 279)
- 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. (Zitiert auf Seite 210)

- Fisher, R. A. (1936). Has mendel's work been rediscovered? *Annals of science*, 1(2):115–137. (Zitiert auf Seite 138)
- Gentleman, R. and Temple Lang, D. (2007). Statistical analyses and reproducible research. *Journal of Computational and Graphical Statistics*, 16(1):1–23. (Zitiert auf Seite 130)
- 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>. (Zitiert auf Seite 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. (Zitiert auf Seite 257)
- Globus, A. (1994). Principles of information display for visualization practitioners. (Zitiert auf Seite 82)
- 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. (Zitiert auf Seite 273)
- Hagemann, O. (2014). TSH-basal. <http://www.laborlexikon.de/Lexikon/Infoframe/t/TSH-basal.htm> [Last access 2015 Aug 29]. (Zitiert auf Seite 113)
- 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. (Zitiert auf Seite 113)
- 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. (Zitiert auf Seite 159)
- Harjutsalo, V., Sund, R., Knip, M., and Groop, P. (2013). Incidence of type 1 diabetes in Finland. *JAMA*, 310(4):427–428. (Zitiert auf Seite 105)
- Harrell, F. (2014). *Regression Modeling Strategies*. Springer Series in Statistics, 2. edition. (Zitiert auf Seite 159)
- 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. (Zitiert auf den Seiten 71 und 83)
- 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. (Zitiert auf Seite 210)
- Hedderich, J. and Sachs, L. (2018). *Angewandte Statistik*. Springer Berlin Heidelberg. (Zitiert auf den Seiten 41, 131 und 168)

- Holm, S. (1979). A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6(2):65–70. (Zitiert auf den Seiten 240 und 265)
- 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]. (Zitiert auf Seite 2)
- Hornik, K. (2022). R FAQ. <http://cran.r-project.org/doc/manuals/R-FAQ.html> [Last access 2022 Oct 08]. (Zitiert auf Seite 3)
- Hothorn, T. and Hornik, K. (2022). *exactRankTests: Exact Distributions for Rank and Permutation Tests*. R package version 0.8-35. (Zitiert auf den Seiten 230 und 235)
- 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. (Zitiert auf den Seiten 217, 220, 224, 230, 235 und 239)
- Howson Ian (2022). R Package Documentation. <https://rdrr.io/> [Last access 2022 Oct 08]. (Zitiert auf Seite 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. (Zitiert auf Seite 128)
- 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]. (Zitiert auf Seite 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). (Zitiert auf Seite 159)
- 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]. (Zitiert auf Seite 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]. (Zitiert auf Seite 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]. (Zitiert auf Seite 70)
- Ioannidis, J. P. A. (2005). Why most published research findings are false. *PLOS Medicine*, 2(8). (Zitiert auf den Seiten 130 und 209)
- Kassambara, A. (2019). *datarium: Data Bank for Statistical Analysis and Visualization*. R package version 0.1.0. (Zitiert auf Seite 245)

- Kassambara, A. (2020). *ggpubr: 'ggplot2' Based Publication Ready Plots.* R package version 0.3.0. (Zitiert auf den Seiten 236 und 249)
- 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.* (Zitiert auf Seite 257)
- Kohl, M. (2005). *Numerical contributions to the asymptotic theory of robustness.* Dissertation, Universität Bayreuth, Bayreuth. (Zitiert auf Seite 149)
- Kohl, M. (2019). *RobLox: Optimally robust influence curves and estimators for location and scale.* R package version 1.2.0. (Zitiert auf den Seiten 128, 149, 152, 177, 193, 194 und 195)
- Kohl, M. (2020a). *MKclass: Statistical Classification.* R package version 0.3. (Zitiert auf den Seiten 155, 156, 157 und 159)
- Kohl, M. (2020b). *MKomics: Functions for Omics Data Analysis.* R package version 0.5. (Zitiert auf Seite 277)
- Kohl, M. (2020c). *MKpower: Power Analysis and Sample Size Calculation.* R package version 0.4. (Zitiert auf den Seiten 190, 197 und 258)
- Kohl, M. (2022a). *MKdescr: Descriptive Statistics.* R package version 0.8. (Zitiert auf den Seiten 19, 21, 27, 28, 30, 48 und 63)
- Kohl, M. (2022b). *MKinfer: Inferential Statistics.* R package version 0.8. (Zitiert auf den Seiten 163, 164, 167, 169, 173, 174, 213, 215, 221, 230, 231, 241, 246, 248, 249 und 277)
- Kohl, M. (2022c). *rmx: Radius-Minimax Estimators.* R package version 0.8. (Zitiert auf den Seiten 128, 149, 150, 152, 178, 193, 194 und 195)
- Kohl, M. and Münch, F. (2022a). Statistik teil 1: Der box- und whisker-plot. *Kardiotechnik*, 31.2022(1):15 – 17. (Zitiert auf Seite 29)
- Kohl, M. and Münch, F. (2022b). Statistik teil 2: Datenorganisation mit tabellenkalkulationsprogrammen. *Kardiotechnik*, 31.2022(2):62 – 65. (Zitiert auf Seite 12)
- Kohl, M. and Münch, F. (2022c). Statistik teil 3: Konfidenzintervalle. *Kardiotechnik*, 31.2022(3):95 – 98. (Zitiert auf den Seiten 160 und 167)
- Kohl, M. and Ruckdeschel, P. (2010). R package distrMod: S4 classes and methods for probability models. *Journal of Statistical Software*, 35(10):1–27. (Zitiert auf den Seiten 134, 135, 147, 164, 171 und 175)
- Kohl, M. and Ruckdeschel, P. (2019). *ROptEst: Optimally robust estimation.* R package version 1.2.1. (Zitiert auf den Seiten 152, 181, 195, 196 und 197)
- Kohl, M., Ruckdeschel, P., and Rieder, H. (2010). Infinitesimally robust estimation in general smoothly parametrized models. *Statistical Methods & Applications*, 19(3):333–354. (Zitiert auf Seite 149)

- 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. (Zitiert auf Seite 114)
- Limpert, E., Stahel, W. A., and Abbt, M. (2001). Log-normal distributions across the sciences: Keys and clues. *BioScience*, 51:341–352. (Zitiert auf Seite 114)
- MAQC-Consortium (2006). The MicroArray Quality Control (MAQC) project shows inter- and intra-platform reproducibility of gene expression measurements. *Nature Biotechnology*, 24:1151–1161. (Zitiert auf Seite 277)
- Mendel, G. (1866). Versuche über pflanzenhybriden. verhandlungen des naturforschenden vereines in brünn, bd. iv für das jahr 1865. *Abhandlungen*, pages 3–47. (Zitiert auf Seite 138)
- Menzel, U. (2021). *EMT: Exact Multinomial Test: Goodness-of-Fit Test for Discrete Multivariate Data*. R package version 1.2. (Zitiert auf Seite 214)
- Morgan, M. (2019). *BiocManager: Access the Bioconductor Project Package Repository*. R package version 1.30.10. (Zitiert auf Seite 128)
- Muenchen, R. A. (2022). The Popularity of Data Science Software. <http://r4stats.com/articles/popularity/> [Last access 2022 Oct 08]. (Zitiert auf Seite 2)
- Neuwirth, E. (2014). *RColorBrewer: ColorBrewer palettes*. R package version 1.1-2. (Zitiert auf den Seiten 71, 73, 76 und 88)
- Novitski, C. E. (2004). Revision of fisher's analysis of mendel's garden pea experiments. *Genetics*, 166(3):1139–1140. (Zitiert auf Seite 138)
- 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. (Zitiert auf Seite 262)
- Pedersen, T. L. and Robinson, D. (2022). *ggridge: A Grammar of Animated Graphics*. R package version 1.0.8. (Zitiert auf Seite 81)
- 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. (Zitiert auf den Seiten 273, 276 und 280)
- 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. (Zitiert auf Seite 154)
- R Consortium (2022). R Consortium. <https://www.r-consortium.org/> [Last access 2022 Oct 08]. (Zitiert auf Seite 2)

- R Core Team (2022a). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. (Zitiert auf den Seiten vi, 1, 80, 128, 134, 186, 210, 228, 241 und 279)
- R Core Team (2022b). *R Data Import/Export*. <http://cran.r-project.org/doc/manuals/r-release/R-data.pdf>. (Zitiert auf Seite 12)
- R Core Team (2022c). R Developer Page. <https://developer.r-project.org/> [Last access 2022 Oct 08]. (Zitiert auf Seite 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>. (Zitiert auf Seite 4)
- Ranke, J. (2022). Index of /bin/linux/debian. <https://cran.r-project.org/bin/linux/debian/> [Last access 2022 Oct 08]. (Zitiert auf Seite 3)
- Ranstam, J. (2012). Why the p-value culture is bad and confidence intervals a better alternative. *Osteoarthritis and Cartilage*, 20(8):805–808. (Zitiert auf Seite 210)
- 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. (Zitiert auf den Seiten 210, 234 und 257)
- Rieder, H. (1994). *Robust asymptotic statistics*. Springer Series in Statistics. Springer. (Zitiert auf Seite 149)
- 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. (Zitiert auf Seite 210)
- 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]. (Zitiert auf Seite 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. (Zitiert auf den Seiten 153, 196 und 197)
- 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. (Zitiert auf den Seiten 91, 96, 100, 103, 106, 111, 113, 116, 121, 124, 125, 135 und 142)
- Rutter, M. (2022). Index of /bin/linux/ubuntu. <https://cran.at.r-project.org/bin/linux/ubuntu/> [Last access 2022 Oct 08]. (Zitiert auf Seite 4)
- 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. (Zitiert auf Seite 234)
- 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. (Zitiert auf Seite 257)
- Sedgwick, P. (2013). P values or confidence intervals? *BMJ*, 346. (Zitiert auf Seite 210)

- 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*. (Zitiert auf Seite 124)
- 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/>. (Zitiert auf den Seiten 91, 96, 100, 104, 107, 111, 113, 116, 118, 120 und 121)
- Song, F., Parekh, S., Hooper, L., Loke, Y. K., Ryder, J., Sutton, A. J., Hing, C., Kwok, C. S., Pang, C., and Harvey, I. (2010). Dissemination and publication of research findings: an updated review of related biases. *Health Technol Assess*, 14(8):1–193. (Zitiert auf Seite 209)
- Sterne, J. A. and Davey Smith, G. (2001). Sifting the evidence-what's wrong with significance tests? *BMJ*, 322(7280):226–231. (Zitiert auf Seite 210)
- Steuer, D. (2022). Index of /bin/linux/suse. <https://cran.at.r-project.org/bin/linux/suse/> [Last access 2022 Oct 08]. (Zitiert auf Seite 3)
- The R Foundation (2022a). Contributors. <https://www.r-project.org/contributors.html> [Last access 2022 Oct 08]. (Zitiert auf Seite 2)
- The R Foundation (2022b). The R Foundation. <https://www.r-project.org/foundation/> [Last access 2022 Oct 08]. (Zitiert auf Seite 2)
- Úcar, I. (2022). Fedora Packages of R Software. <https://cran.at.r-project.org/bin/linux/fedora/> [Last access 2022 Oct 08]. (Zitiert auf Seite 4)
- Venables, W. N. and Ripley, B. D. (2002). *Modern Applied Statistics with S*. Springer, New York, fourth edition. ISBN 0-387-95457-0. (Zitiert auf den Seiten 128, 134, 149, 164 und 260)
- Wald, A. (1980). *A method of estimating plane vulnerability based on damage of survivors*. Center for Naval Analyses, crc 432 edition. (Zitiert auf Seite 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. (Zitiert auf Seite 210)
- 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]. (Zitiert auf Seite 98)
- WHO (2015b). Diabetes. <http://www.who.int/mediacentre/factsheets/fs312/en/> [Last access 29.08.2015]. (Zitiert auf den Seiten 93 und 101)
- Wickham, H. (2009). *ggplot2: Elegant graphics for data analysis*. Springer New York. <http://had.co.nz/ggplot2/book>. (Zitiert auf den Seiten 19, 21, 23, 30, 36, 39, 42, 52, 55, 58, 61, 67, 68, 75, 84, 85, 87, 88, 89, 137, 138, 140, 141, 143, 233, 236 und 244)
- Wickham, H. and Seidel, D. (2022). *scales: Scale Functions for Visualization*. R package version 1.2.1. (Zitiert auf den Seiten 19, 21 und 24)

- Wikipedia (2015). Körpergröße — Wikipedia, Die freie Enzyklopädie. <https://de.wikipedia.org/w/index.php?title=Körpergröße&oldid=145235809> [Last access 30.08.2015]. (Zitiert auf Seite 110)
- 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]. (Zitiert auf Seite 18)
- Wikipedia contributors (2022a). Andora — Wikipedia, the free encyclopedia. <https://en.wikipedia.org/w/index.php?title=Andora&oldid=1113739296> [last access 1-November-2022]. (Zitiert auf Seite 98)
- Wikipedia contributors (2022b). Bilirubin — Wikipedia, the free encyclopedia. <https://en.wikipedia.org/w/index.php?title=Bilirubin&oldid=1116981450> [last access 1-November-2022]. (Zitiert auf Seite 18)
- Wikipedia contributors (2022c). Finland — Wikipedia, the free encyclopedia. <https://en.wikipedia.org/w/index.php?title=Finland&oldid=1118943471> [last access 1-November-2022]. (Zitiert auf Seite 105)
- 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]. (Zitiert auf Seite 111)
- Wikipedia contributors (2022e). Markdown — Wikipedia, the free encyclopedia. <https://en.wikipedia.org/w/index.php?title=Markdown&oldid=1114072172> [Online; accessed 11-October-2022]. (Zitiert auf Seite 6)
- 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]. (Zitiert auf Seite 113)
- Xiao, N. (2018). *ggsci: Scientific Journal and Sci-Fi Themed Color Palettes for 'ggplot2'*. R package version 2.9. (Zitiert auf den Seiten 75, 76, 88, 234 und 243)
- Xie, Y. (2013). animation: An R package for creating animations and demonstrating statistical methods. *Journal of Statistical Software*, 53(1):1–27. (Zitiert auf Seite 81)
- Xie, Y. (2015a). *Dynamic Documents with R and knitr*. Chapman and Hall/CRC, Boca Raton, Florida, 2nd edition. ISBN 978-1498716963. (Zitiert auf den Seiten v, 6 und 7)
- 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>. (Zitiert auf Seite vi)
- Xie, Y., Allaire, J., and Grolemund, G. (2018). *R Markdown: The Definitive Guide*. Chapman and Hall/CRC, Boca Raton, Florida. ISBN 9781138359338. (Zitiert auf Seite 6)
- Youden, W. J. (1950). Index for rating diagnostic tests. *Cancer*, 3:32–35. (Zitiert auf Seite 154)

Zeileis, A., Hornik, K., and Murrell, P. (2009). Escaping RGBland: Selecting Colors for Statistical Graphics. *Computational Statistics & Data Analysis*, 53:3259–3270. (Zitiert auf Seite 71)

Sachverzeichnis

- Absolute Häufigkeit, 21
 - Kreuztabelle, 36, 37
- Absolute Stetigkeit, 107
- Achsenbeschriftung, 22, 24
- Adjustierter p-Wert, 264
 - Benjamini-Hochberg-Methode, 271
 - Benjamini-Yekutieli-Methode, 273
 - Bonferroni-Holm-Methode, 265
 - Bonferroni-Methode, 264, 265
 - Holm-Methode, 265
 - Simes-Hochberg-Methode, 267
 - step-down Verfahren, 265
 - step-up Verfahren, 267, 272
- Adjustiertes Signifikanzniveau, 264
- AL-Schätzer, 148
- Alpha Blending, 32, 42, 87
- α -Quantil, 25, 26
- Alternative (Hypothese), 201
- Ansari-Bradley Test, 251
- Anzahl an R Paketen, 3
- Anzahl R Pakete, 3
- Arbeitsverzeichnis kontrollieren, 15
- Arbeitsverzeichnis wechseln, 15
- Arithmetisches Mittel, 44, 50
 - biasfrei, 131
 - effizient, 131
 - Konfidenzintervall, 162
 - Logarithmierte Beobachtungen, 45
- Ausreißer, 30, 51, 66
 - Kendalls τ , 66
 - Median, 35
 - Quantile, 34
- Spearmans ρ , 66
- balancierte Genauigkeit (bACC), 154
- Balanciertes Design, 245
- Balkendiagramm, 22, 23, 39, 75
- base packages, 2
- Beispiel
 - Ausfallrate von Glühbirnen, 104
 - Diabetes in Andorra, 98
 - Intelligenzquotient, 111
 - Krankenhausaufenthaltsdauer, 117
 - Körpergröße in Deutschland, 110
 - Lebensdauer eines Akkus, 116, 119
 - Normbereich von Thyreotropin (TSH), 113
 - Prävalenz von Diabetes, 93, 102
 - Qualitätskontrolle von Glühbirnen, 92, 93, 95, 98, 101
 - Sonntagsfrage, 188
 - Typ-1-Diabetes in Finnland, 105
 - Windgeschwindigkeiten, 121
- Benjamini-Hochberg-Methode, 271
- Benjamini-Yekutieli-Methode, 273
- Bernoulli-Verteilung, 92–94
- Bias, 131
- Big data, 262
- Bilirubin, 18
- Bindungen, 230
- Binomialverteilung, 93, 94, 99, 101, 105
- Bitmap, 80
- bmp, 80
- Bonferroni-Holm-Methode, 265
- Bonferroni-Methode, 264, 265
- Bootstrap, 136, 157, 165

- BCa-Intervall, 167
- Konfidenzintervall, 166
- Normalapproximationsintervall, 167
- Perzentilintervall, 167
- stratifiziert, 157
 - Studentisiertes Intervall, 167
- Bootstrap-Konfidenzintervall, 166
- Box-und-Whisker Plot, 29, 30, 83, 84, 244
- χ^2 -Test, 216, 219
- χ^2 -Verteilung, 115, 122, 123
- Cochran–Mantel–Haenszel χ^2 -Test, 222, 224
- ColorBrewer, 71
 - Auswahlkriterien, 72
 - divergierende Farbpaletten, 72
 - qualitative Farbpaletten, 71
 - sequentielle Farbpaletten, 71
- contributed packages, 3
- Cramér-von-Mises Distanz, 146
- Cramérs V, 38
- CV, 48
- CvM-MD-Schätzer, 146
 - Konfidenzintervall, 175
- Data Scientist, 262
- Datenexport, 14
- Datenimport, 12, 14
 - Datenstruktur, 16
 - kontrollieren, 16
 - RStudio, 13
 - Textdatei, 12
- Deskriptive Statistik, 8
 - Ziel, 9
- Dichte, 107
- Dichteplot, 136, 137
- Dichteschätzung, 56–59
- Diskrete Verteilung, 91
- Diskrete Wahrscheinlichkeitsverteilung, 91
- Diskrete Zufallsvariable, 91
- Ein-Stichproben Binomialtest, 211
 - asymptotisch, 211
 - exakt, 211
- Ein-Stichproben Multinomialtest, 214
- Ein-Weg ANOVA, 237, 240
 - Messwiederholungen, 245
 - repeated measures, 245
 - Welch, 237
- Empfehlungen nach E. Tufte, 82
- Empfohlene Pakete, 3
- Empirische Häufigkeitsverteilung, 21
- Empirische Verteilungsfunktion, 35, 36, 60, 61
- Encapsulated PostScript, 80
- Endlichkeitskorrektur, 169
- Entwicklungsgeschichte von S und R, 1
- eps, 80
- Erlang-Verteilung, 115
- Erwartungswert, 92, 108
- Erweiterungspakete, 3
 - Installation, 19
 - Installation mit RStudio, 19
- Exakter Test von Fisher, 216
- Exponentialverteilung, 115, 119
- externe Validierung, 136
- Extremwertverteilung, 119
- F-Test, 250
- F-Verteilung, 123
- fair, 129
- Fallzahlplanung, 188, 205, 210
- false discovery rate (FDR), 271
- family-wise error rate (FWER), 262
- Farbcodierung, 74
- FDR, 271
- for-Schleife, 144
- Formel, 84
- Formeloperator \sim , 84
- Formmaß, 49
- Friedman Test, 245
- FWER, 262
 - unabhängige Tests, 263
- Gamma-Funktion, 114
- Gamma-Verteilung, 114, 115
- Gauß-Verteilung, 108

- Geometrische Standardabweichung, 49
 Geometrische Verteilung, 102, 115
 Geometrisches Mittel, 45
 gewichtete Genauigkeit (bACC), 155
 GNU Projekt, 2
 Goldstandard, 202
 Golub-Datensatz, 273
 Grafikexport, 79, 80
 - Bild, 79
 - pdf, 79
 - RStudio, 79
 Grafiksysteme, 23
 Grammar of graphics, 23
 gross-error Modell, 148
 Grundgesamtheit, 9
 Grundpakete, 2

 Hexadezimalcode, 74
 Histogramm, 53, 55, 57–59, 85, 136, 137
 HL-Schätzer, 241
 Hodges-Lehmann Schätzer, 241
 Holm-Methode, 265
 Hypergeometrische Verteilung, 98, 99
 Hypothese, 200
 - einseitig, 201
 - zweiseitig, 201
 Induktive Statistik, 8, 129
 - Ziel, 9
 Influenzfunktion, 148
 interne Validierung, 136
 Interquartilsabstand, 28
 Intervallschätzer, 160
 Inzidenz, 104
 Inzidenzrate, 104
 IQR, 28, 48, 51, 145
 - konsistent, 146
 ITS, 14
 ITS-Datensatz, 14
 - Beschreibung der Variablen, 18
 - Bilirubin, 45, 49, 155–159
 - Ergebnis, 218, 237–241, 244
 Geschlecht, 36–39, 214, 216, 218, 231, 251
 Herzfrequenz, 62, 64–67, 86, 87, 253
 Import, 15
 Leberversagen, 132, 155–159, 169, 171, 175, 211, 213, 214, 216, 218
 LOS, 41–44, 51, 52, 55
 OP, 21, 22, 24, 36–39, 75, 83, 84, 218
 SAPS II, 27–30, 35, 36, 41–43, 83, 84
 Temperatur, 44, 46–48, 50–54, 56–62, 64–67, 85–87, 132, 134–139, 142, 145, 147, 149, 163, 164, 173, 175, 226, 227, 231, 237–241, 244, 251, 253, 256
 ITSDaten.csv, 14

 Jittering, 33
 jpeg, 80

 Kendalls τ , 41
 Kolmogorov(-Smirnov) Distanz, 146
 Konfidenzintervall, 160, 161
 - Arithmetisches Mittel, 162
 - CvM-MD-Schätzer, 175
 - einseitig, 161
 - Konfidenzlevel, 161
 - Konfidenzsranken, 161
 - MAD, 172
 - Maximaler Schätzfehler, 161
 - MD-Schätzer, 175
 - Median, 172
 - Normalverteilungsapproximation, 162
 - Punktschätzer, 161
 - Relative Häufigkeit, 168, 169
 - Varianz, 162
 Kontaminationsumgebung, 148
 Kontingenzkoeffizient, 38
 Kontingenztafel, 36, 214
 Korrelationstest, 253
 - Kendall, 253
 - Pearson, 253
 - Spearman, 253
 Kovarianz, 62
 Kreuztabelle, 36

- Kreuzvalidierung, 136, 157
- Kruskal-Wallis Test, 237, 240
- KS-MD-Schätzer, 146
- Kuchendiagramm, 24
- Kumulative Häufigkeit, 22
- Kurtosis, 51
 - Normalverteilung, 52
- Likelihood-Funktion, 133
- limit of blank (LOB), 153
- limit of detection (LOD), 153, 154
- limit of quantification (LOQ), 153, 154
- limit of quantitation (LOQ), 153, 154
- linksschief, 50
- Log-Likelihood-Funktion, 134
- log-Normalverteilung, 112
- Lokations- und Skalenmodell, 135
- LOS, 18
- MAD, 28, 47, 48, 51, 145
 - Konfidenzintervall, 172
 - konsistent, 146
- Mann-Whitney U-Test, *siehe* Wilcoxon Rangsummentest
- Markdown, 6
- Maximum-Likelihood-Schätzer, *siehe* ML-Schätzer
- McNemar Test, 219
- MD-Schätzer, 146
 - Cramér-von-Mises, *siehe* CvM-MD-Schätzer
 - Kolmogorov(-Smirnov), 146
 - Konfidenzintervall, 175
 - konsistent, 146
- Median, 26, 27, 34, 44, 48, 50, 145
 - Konfidenzintervall, 172
 - konsistent, 146
- Median der absoluten Abweichungen von Median, *siehe* MAD
- Merkmal, 10
 - kategorial, 10
 - metrisch, 10
- qualitative, 10
- quantitativ, 10
- Merkmalsarten, 10
- Merkmalsausprägung, 10
- Minimum-Distanz-Schätzer, *siehe* MD-Schätzer
- ML-Schätzer, 133
 - Bernoulli-Modell, 134
 - effizient, 133
 - Exponentialmodell, 134
 - Normalverteilungsmodell, 134
 - Poisson-Modell, 134
- Modalwert, 21
- Modelldiagnostik, 135
- Modellvalidierung, 135
- Modus, 21
- MSE, 131
 - maximaler asymptotischer, 148
- Namensraum, 121
- Namespace, 121
- Negative Binomialverteilung, 101, 102
- Normalverteilung, 108, 109
- Normbereich, 113, 153
- Nullhypothese, 201
- Odds, 218
- Odds-Ratio, 218
- OR, 218
- p-Wert
 - adjustiert, 264
- Parametrische Familie, 130
- Parametrisches Modell, 130
- Pascal-Verteilung, 102
- pdf, 80
- pdf \LaTeX , v, vi
- Pearson-Korrelation, 62
- Pearsons Kontingenzkoeffizient, 38
- Perzentil, 26
- ϕ -Koeffizient, 38
- Plottitel, 22, 24
- png, 80
- Poisson-Verteilung, 104, 105

- Population, 9
- Portable document format, 80
- Portable network graphics, 80
- Post hoc Tests, 239
- PostScript, 80
- pp-Plot, 138
- Primärer Endpunkt, 262
 - Multiple, 262
- ps, 80
- Punktschätzer, *siehe* Schätzer
- Punktschätzung, *siehe* Schätzung
- Pólya-Verteilung, 102
- qq-Plot, 139, 142
- Quade Test, 245
- Quantil-Quantil-Plot, *siehe* qq-Plot
- Quantifunktion, 92, 94
- Quartil, 26, 28, 34, 48
- Quartilsdispersionskoeffizient, 48
- R Code Chunk, 7
- R Consortium, 2
- R Core Development Team, 2
- R Foundation, 2
- R Installation
 - Linux, 4
 - Mac OS X, 3
 - Windows, 3
- R Markdown, 6
- R Skript, 5
- R Windows GUI, 4
- R6-Objektorientierung, 96
- Radius Minimax Schätzer, 149
- Randomisierung, 92
- Rang, 40
- Rangkorrelation, 40–42, 62
- Realisation, 91
- rechtsschief, 50
- recommended packages, 3
- Referenzbereich, 113, 153
- Relative Häufigkeit, 21
 - Approximatives Konfidenzintervall, 168
- biasfrei, 131
- effizient, 131
- Exaktes Konfidenzintervall, 168
- Konfidenzintervall, 168, 169
- Kreuztabelle, 37
- Relatives Risiko, 218
- Reproduzierbare Forschung, 130
- Repräsentative Stichprobe, 9
- Resampling, 136
- resubstitution bias, 157
- Resubstitutions Bias, 136
- RGB-Code, 74
- RStudio
 - Fenster, 5
 - Fenster Environment, 13, 16
 - Fenster Help, 23
 - Fenster History, 13
 - Fenster Packages, 19
 - Interaktive Hilfe, 23
 - panes, 5
- RStudio IDE, 5
- S3, 2
- S4, 2
- S4-Objektorientierung, 96
- SAPS II, 18
- Scalable vector graphics, 80
- Schiefe, 49, 51
- Schätzer, 130, 131
 - biasfrei, 131
 - effizient, 131
 - konsistent, 131
- Schätzerkonstruktion, 133
- Schätzung, 130
- SE, 162
- SEM, 162
- Sensitivität, 154
- Signifikanzniveau
 - adjustiert, 264
- Simes-Hochberg-Methode, 267
- Six Sigma, 109
- Skalenniveau

- intervallskaliert, 10
- nominal, 10, 20, 36
- ordinal, 10, 25
- ratioskaliert, 10
- Skalenniveaus, 10
- Spearmans ρ , 41, 62
- Spezifität, 154
- Standardabweichung, 46, 47, 51, 92
 - Standardisierung, 46
- Standardfehler, 135, 162
- Standardnormalverteilung, 109
- Startschätzer, 149
- Statistik-Programmiersprache S, 1
- Statistischer Hypothesentest, *siehe* Statistischer Test
- Statistischer Test, 200
 - Ablehnungsbereich, 204
 - Annahmebereich, 204
 - Ansari-Bradley Test, *siehe* Ansari-Bradley Test
 - χ^2 -Test, *siehe* χ^2 -Test
 - Ein-Stichproben Binomialtest, 211
 - Ein-Weg ANOVA, *siehe* Ein-Weg ANOVA
 - Entscheidungen, 202
 - Exakter Test von Fisher, *siehe* Exakter Test von Fisher
 - F-Test, *siehe* F-Test
 - Fallzahlplanung, 205
 - Fehler 1. Art, 202, 203
 - Fehler 2. Art, 202, 203
 - Hoch signifikant, 203
 - Höchst signifikant, 203
 - Korrelationstest, *siehe* Korrelationstest
 - Kruskal-Wallis Test, *siehe* Kruskal-Wallis Test
 - p-Wert, 204
 - post hoc, 239
 - Power, 202
 - Relevanz, 204
 - Schritte, 203
 - Sensitivität, 202
 - Signifikant, 203
 - Spezifität, 202
 - t-Test, *siehe* t-Test
 - Test auf Normalverteilung, *siehe* Test auf Normalverteilung
 - Verteilungstest, 255
 - Wilcoxon Rangsummentest, 235
 - Statlib, 2
 - Stetige Verteilung, 107
 - Stetige Wahrscheinlichkeitsverteilung, 107
 - Stetige Zufallsvariable, 107
 - Stetigkeitskorrektur, 168
 - Streudiagramm, 42, 64, 86, 87
 - Struktur eines Funktionsaufrufs, 1
 - svg, 80
- t-Test
 - 1-Stichproben, 226
 - Bootstrap, 231
 - gepaart, 228
 - Hsu, 231
 - paarweise, 240
 - Permutation, 231
 - Student, 230
 - Welch, 231, 240
 - Zwei-Stichproben, 204–206, 208, 209, 230
- t-Verteilung, 123
- Tagged image file format, 80
- Test auf Normalverteilung, 255
 - Cramér-von Mises Test, 256
 - Lilliefors (Kolmogorov-Smirnov) Test, 256
 - Shapiro-Francia Test, 256
 - Shapiro-Wilk Test, 256
- Test von McNemar, 219
- tiff, 80
- Tortendiagramm, 24
 - Nachteile, 25
- Totalerhebung, 8
- Umgang mit Farben, 70, 71
- Umgebung, 148
- Variable, 10

- metrisch, 43
- Variablennamen, 18
- Varianz, 46, 47, 92, 108
 - biasfrei, 132
 - effizient, 132
 - Konfidenzintervall, 162
 - Standardisierung, 46, 132
- Variationskoeffizient, 48
- Verteilung
 - flachgipflig, 51
 - leptokurtisch, 51
 - linksschief, 49, 50
 - platykurtisch, 51
 - rechtsschief, 49, 50
 - steilgipflig, 51
- Verteilungsfunktion, 92, 94
- Verteilungstest, 255
- Vulkanplot (volcano plot), 277
- Wahrscheinlichkeit, 91
- Wahrscheinlichkeitsdichte, 107
- Wahrscheinlichkeitsfunktion, 92, 94
- Wahrscheinlichkeitsmodell, 9, 10
- Wahrscheinlichkeitsrechnung, 8
- Wartezeitverteilung, 102
- Weibull-Verteilung, 119
- Wilcoxon Rangsummentest, 235
 - paarweise, 240
- Wilcoxon Vorzeichen-Rangtest, 227
- Windows 64bit-Systeme, 2
- WMW-Test, *siehe* Wilcoxon Rangsummentest,
 - 240
 - paarweise, 240
- Wölbung, 51
- Youdens J Statistik, 154
- z-Score, 50
- z-Transformation, 50
- Zentraler Grenzwertsatz, 109, 162
- Zufallsstichprobe, 9
- Zufallsvariable, 91
- Zufallszahlen, 94
- Zugriffsoperator [, 50
 - negativer Index, 51
- Zugriffsoperator \$, 21
- Zuweisung, 13, 16
- Zuweisungsoperator, 16
- 2σ -Regel, 109
- 2×2 -Kontingenztafel, 214

R Verzeichnis

—, 6
.RData, 20
:;, 101
::, 120==, 86
?barplot, 23
[, 50, 86
\$, 21
“, 7
“r, 7

abline, 54
add = TRUE, 110
aes, 24
after_stat(count)/sum(after_stat(count)), 24
animation (Paket), 81
annotate, 64, 137, 234, 243
ansari.test, 251
ansari.text, 251

CramerV, 38, 218
CramerVonMisesTest, 256
curve, 110, 117, 119
CV, 48
CvMMDEstimator, 147, 175

data, 228
data.frame, 14, 16, 21, 233, 242, 260
dataarium (Paket), 245
datasets (Paket), 2, 228
dbinom, 94
decisionStump, 159
density, 56
DescTools (Paket), 19, 21, 37, 38, 45, 49, 50, 52,
 218, 254, 256, 260
detectCores, 186
dev.off, 80
dexp, 115
dgamma, 115
dhyper, 99
display.brewer.all, 71
distr (Paket), 91, 96, 100, 103, 106, 111, 113,
 116, 120, 121, 124, 125, 135, 142
distr6 (Paket), 91, 96, 100, 104, 107, 111, 113,
 116, 118, 120, 121
distrMod (Paket), 134, 135, 147, 164, 171, 175
distrModOptions, 135
dlnorm, 112
dnbinom, 102
dnorm, 109
do.points = FALSE, 61
dPlot, 151
dpois, 105

ecdf, 35, 60
EMT (Paket), 214
exactRankTests (Paket), 230, 235
exp, 46
expression, 278

Factor, 18
FALSE, 16
fill, 84

fisher.test, 216
fitdistr, 134, 135, 149, 164
fitdistrplus (Paket), 134
for, 144
foreign (Paket), 3
Freq, 21, 22

Gammad, 125
geom_bar, 24, 29, 39
geom_boxplot, 30, 84
geom_density, 58, 137
geom_histogram, 55, 58, 59, 137
geom_jitter, 33
geom_point, 31
geom_smooth, 64
geom_text, 234, 243
getOutliers, 150
gganimate (Paket), 81
ggpaired, 249
ggplot, 23, 29, 60, 137, 244
ggplot2 (Paket), 19, 21, 23, 30, 36, 39, 42, 52,
 55, 58, 61, 67, 68, 75, 84, 85, 87–89,
 137, 138, 140, 141, 143, 233, 236, 244
ggpubr, 249
ggpubr (Paket), 236
ggsci (Paket), 75, 76, 88, 234, 243
gttitle, 24, 29
Gmean, 45, 46
graphics (Paket), 2
grDevices (Paket), 2, 80
grid (Paket), 2
grid.arrange, 151, 249
gridExtra (Paket), 151, 249
Gsd, 49

head, 246
hist, 53, 57, 59, 89
hsu.t.test, 230

illustrate.boxplot, 30
illustrate.quantile, 27
install.packages, 19
install_github, 91

integer, 18
IQR, 28
iqrCV, 48
ITSDaten, 16
jpeg, 80
KendallTauB, 254, 255
KernSmooth (Paket), 3
knitr (Paket), v, vi, 6
KolmogorovMDEstimator, 147
kruskal.test, 237, 238
kruskal_test, 239
Kurt, 52
lattice (Paket), 3
legend, 86
legend.text = TRUE, 39
library, 20, 69, 91, 128, 200, 261
LillieTest, 256
lines, 57
load, 14
log, 45
lower.tail = FALSE, 95, 99, 105
lwd, 137
mad, 28
madCI, 173, 174
main, 22
mantelhaen.test, 222
MASS (Paket), 3, 128, 134, 149, 164, 260
Matrix (Paket), 3
mcnemar.test, 219
MDEstimator, 147
mean, 44, 134
meanCI, 163, 167
meanlog, 112
medCV, 48
median, 27
medianCI, 173, 174
methods (Paket), 2
mfrow, 144
mgcv (Paket), 3
mh_test, 220
MKclass (Paket), 155–157, 159
MKdescr (Paket), 19, 21, 27, 28, 30, 48, 63
MKinfer (Paket), 163, 164, 167, 169, 173, 174, 213, 215, 221, 230, 231, 241, 246, 248, 249, 277
MKomics (Paket), 277
MKpower (Paket), 190, 197, 258
MLEstimator, 135, 164, 171
mt.rawp2adjp, 276
multinomial.test, 214
multtest (Paket), 273, 276, 280
nlme (Paket), 3
nnet (Paket), 3
normCI, 164, 167
NormLocationScaleFamily, 135
nrow, 21
numeric, 18
oneway.test, 237
optCutOff, 155
optCutoff, 156, 157
outlier, 150
p.adjust, 266, 268, 273, 278, 279
pairwise.ext.t.test, 241, 249
pairwise.fun, 241, 248
pairwise.logfc, 277
pairwise.t.test, 239, 240
pairwise.wilcox.exact, 241, 249
pairwise.wilcox.test, 239, 240
pal_npg, 76, 78
par, 144
parallel (Paket), 2, 128, 186
paste, 278
path.package, 153
rbinom, 94
pdf, 80
percent_format, 24
PercTable, 37
perfMeasures, 155–157
perfScores, 159

- perm.t.test, 231
pexp, 115
pgamma, 115
Phi, 38
phyper, 99
pie, 24, 25
plnorm, 112
plot, 35, 42, 56, 60, 86, 89, 96, 97
plotmath, 278
pnbinom, 102
png, 80
pnorm, 109
points, 43, 86
postscript, 80
power.anova.test, 210
power.hsu.t.test, 258
power.prop.test, 210, 279
power.t.test, 205–207, 210
power.welch.t.test, 258, 270
ppois, 105
ppPlot, 151
prop.test, 211, 214
proportions, 37, 39

qbinom, 94
qexp, 115
qgamma, 115
qhyper, 99
qlnorm, 112
qnbinom, 102
qnorm, 109
qpois, 105
qqline, 139, 144, 260
qqnorm, 139, 144, 260
qqPlot, 151
qqplot, 142
qqplotr (Paket), 138, 140, 143
quantile, 26

rank, 67
rate, 115
rbinom, 94, 95

RColorBrewer (Paket), 71, 73, 76, 77, 88
read.*, 13, 14
read.csv, 12
read.csv2, 12, 16
read.delim, 12
read.delim2, 12
read.table, 12
readRDS, 14
remotes (Paket), 91
remotes::install_github, 91
remove.packages, 20
rep, 85
rev, 85
rexp, 115
rfrq, 37
rgamma, 115
rgb, 74
rhyper, 99
rlnorm, 112
rm.oneway.test, 246
rmarkdown (Paket), 6
rmx, 149, 178, 193–195
rmx (Paket), 128, 149, 150, 152, 153, 178, 193–
195
rnbinom, 102
rnorm, 109, 144, 208, 258
RobExtremes (Paket), 153, 196, 197
roblox, 177, 193–195
RobLox (Paket), 128, 149, 152, 177, 193–195
roptest, 152, 181, 195–197
ROptEst (Paket), 152, 153, 181, 195–197
round, 37, 44
rpart (Paket), 3
rpois, 105

save, 14
save.image, 14
saveRDS, 14
scale_colour_manual, 87
scale_fill_grey, 40
scale_y_*, 24
scales (Paket), 19, 21, 24

scan, 12
sd, 46, 134, 135
sdCI, 164, 167
sdlog, 112
selfesteem, 245
seq, 26
shapiro.test, 256
ShapiroFranciaTest, 256
sim ssize.wilcox.test, 258
simCorVars, 63
sIQR, 28
Skew, 50
sleep, 228
spatial (Paket), 3
SpearmanRho, 254
splines (Paket), 2
ssize.propCI, 190, 197
stat_compare_means, 236
stat_ecdf, 36
stat_edcf, 61
stat_function, 137
stat_pp_band, 138
stat_pp_line, 138
stat_pp_point, 138
stat_qq, 140
stat_qq_band, 143
stat_qq_line, 140, 143
stat_qq_point, 143
stats (Paket), 2, 210, 279
stats4 (Paket), 3, 134
str, 16
stringsAsFactors, 16, 17
summary, 97, 100, 104, 107, 111, 113, 116, 118,
 120, 121, 150
survival (Paket), 3
svg, 80

t.test, 163, 208, 209, 226, 228, 230
table, 21, 36, 39
tail, 246
tapply, 39, 40, 233, 241, 242, 251
tcltk (Paket), 3

theme, 31
tiff, 80
tools (Paket), 3
TRUE, 16, 17

utils (Paket), 3

var, 46
var.equal = TRUE, 230
var.test, 250
View, 16
vignette, 153
volcano, 277

Weibull, 120
wilcox.exact, 230, 235
wilcox.test, 235
wilcox_test, 235
wilcoxsign_test, 230
write.csv, 14
write.csv2, 14
write.table, 14

xlab, 31
xlim, 31, 54

ylab, 22, 24, 29