

# Praxis der Datenanalyse

Skript zum Modul

*Sebastian Sauer. Mit Beiträgen von Oliver Gansser, Matthias Gehrke,  
Karsten Lübke und Norman Markgraf*

*06 July, 2017*



# Inhaltsverzeichnis

Placeholder	ix
Organisatorisches	xi
<b>I Grundlagen</b>	<b>1</b>
<b>1 Rahmen</b>	<b>3</b>
1.1 Software installieren . . . . .	4
1.1.1 R und RStudio installieren . . . . .	5
1.1.2 Pakete . . . . .	6
1.1.3 Hilfe! R startet nicht! . . . . .	8
1.1.4 Vertiefung: Zuordnung von Paketen zu Befehlen . . . . .	9
1.1.5 Datensätze . . . . .	11
1.2 ERRRstkontakt . . . . .	12
1.2.1 R-Skript-Dateien . . . . .	12
1.2.2 Datentypen in R . . . . .	12
1.2.3 Hinweise . . . . .	13
1.2.4 Text und Variablen zuweisen . . . . .	14
1.2.5 Funktionen aufrufen . . . . .	14
1.2.6 Das Arbeitsverzeichnis . . . . .	15
1.3 Hier werden Sie geholfen . . . . .	16
1.3.1 Wo finde ich Hilfe? . . . . .	16
1.3.2 Einfache reproduzierbare Beispiele (ERBies) . . . . .	16
1.4 Was ist Statistik? Wozu ist sie gut? . . . . .	17
1.5 Aufgaben . . . . .	19
1.6 Befehlsübersicht . . . . .	20
1.7 Verweise . . . . .	21
<b>2 Daten einlesen</b>	<b>23</b>
<b>3 Datenjudo</b>	<b>25</b>
<b>4 Praxisprobleme der Datenaufbereitung</b>	<b>27</b>

5	Fallstudie ‘movies’	29
6	Daten visualisieren	31
7	Fallstudie zur Visualisierung	33
8	Grundlagen des Modellierens	35
9	Der p-Wert, Inferenzstatistik und Alternativen	37
10	Lineare Regression	39
11	Klassifizierende Regression	41
12	Fallstudien zum geleiteten Modellieren	43
13	Vertiefung: Clusteranalyse	45
14	Vertiefung: Dimensionsreduktion	47
15	Vertiefung: Grundlagen des Textmining	49
16	Placeholder	51
17	Probeklausur	53
18	Hinweise	55
19	Literaturverzeichnis	57

# Tabellenverzeichnis

1.1	Wichtige Datentypen in R . . . . .	12
1.2	Befehle des Kapitels 'Rahmen' . . . . .	21



# Abbildungsverzeichnis

1.1	Der Prozess der Datenanalyse . . . . .	4
1.2	RStudio . . . . .	5
1.3	So installiert man Pakete in RStudio . . . . .	7
1.4	Hier werden Sie geholfen: Die Dokumentation der R-Pakete . . . . .	10
1.5	Das Arbeitsverzeichnis mit RStudio auswählen . . . . .	15
1.6	Sinnbild für die Deskriptiv- und die Inferenzstatistik . . . . .	18





# Placeholder



# Organisatorisches



# Teil I

## Grundlagen



# Kapitel 1

## Rahmen

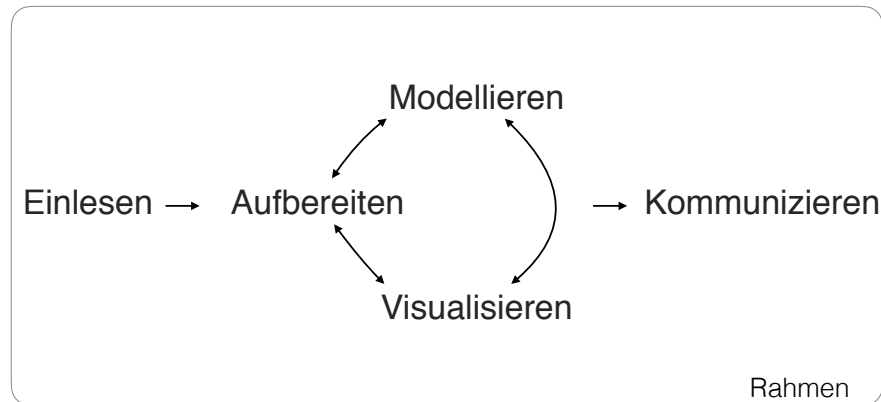


### Lernziele:

- Einen Überblick über die fünf wesentliche Schritte der Datenanalyse gewinnen.
- R und RStudio installieren können.
- Einige häufige technische Probleme zu lösen wissen.
- R-Pakete installieren können.
- Einige grundlegende R-Funktionalitäten verstehen.
- Auf die Frage “Was ist Statistik?” eine Antwort geben können.

In diesem Skript geht es um die Praxis der Datenanalyse. Mit Rahmen ist das “Drumherum” oder der Kontext der eigentlichen Datenanalyse gemeint. Dazu gehören einige praktische Vorbereitungen und ein paar Überlegungen. Zum Beispiel brauchen wir einen Überblick über das Thema. Voilà (Abb. 1.1):

Datenanalyse, praktisch betrachtet, kann man in fünf Schritte einteilen (Wickham und Grolemund 2016). Zuerst muss man die Daten *einlesen*, die Daten also in R (oder einer anderen Software) verfügbar machen (laden). Fügen wir hinzu: In *schöner Form* verfügbar machen; man nennt dies auch *tidy data* (hört sich cooler an). Sobald die Daten in geeigneter



**Abbildung 1.1:** Der Prozess der Datenanalyse

Form in R geladen sind, folgt das *Aufbereiten*. Das beinhaltet Zusammenfassen, Umformen oder Anreichern je nach Bedarf. Ein nächster wesentlicher Schritt ist das *Visualisieren* der Daten. Ein Bild sagt bekanntlich mehr als viele Worte. Schließlich folgt das *Modellieren* oder das Hypothesen prüfen: Man überlegt sich, wie sich die Daten erklären lassen könnten. Zu beachten ist, dass diese drei Schritte - Aufbereiten, Visualisieren, Modellieren - keine starre Abfolge sind, sondern eher ein munteres Hin-und-Her-Springen, ein aufbauendes Abwechseln. Der letzte Schritt ist das *Kommunizieren* der Ergebnisse der Analyse - nicht der Daten. Niemand ist an Zahlenwüsten interessiert; es gilt, spannende Einblicke zu vermitteln.

Der Prozess der Datenanalyse vollzieht sich nicht im luftleeren Raum, sondern ist in einem *Rahmen* eingebettet. Dieser beinhaltet praktische Aspekte - wie Software, Datensätze - und grundsätzliche Überlegungen - wie Ziele und Grundannahmen.

## 1.1 Software installieren

Als Haupt-Analysewerkzeug nutzen wir R; daneben wird uns die sog. “Entwicklungsumgebung” RStudio einiges an komfortabler Funktionalität beschere. Eine Reihe von R-Paketen (“Packages”; d.h. Erweiterungen) werden wir auch nutzen. R ist eine recht alte Sprache; viele Neuerungen finden in Paketen Niederschlag, da der “harte Kern” von R lieber nicht so stark geändert wird. Stellen Sie sich vor: Seit 29 Jahren nutzen Sie eine Befehl, der Ihnen einen Mittelwert ausrechnet, sagen wir die mittlere Anzahl von Tassen Kaffee am Tag. Und auf einmal wird der Mittelwert anders berechnet?! Eine Welt stürzt ein! Naja, vielleicht nicht ganz so tragisch in dem Beispiel, aber grundsätzlich sind Änderungen in viel benutzen Befehlen potenziell problematisch. Das ist wohl ein Grund, warum sich am “R-Kern” nicht so viel ändert. Die Innovationen in R passieren in den Paketen. Und es gibt viele davon; als ich diese Zeilen schreibe, sind es fast schon 10.000! Genauer: 9937 nach dieser Quelle: <https://cran.r-project.org/web/packages/>. Übrigens können R-Pakete auch Daten enthalten.



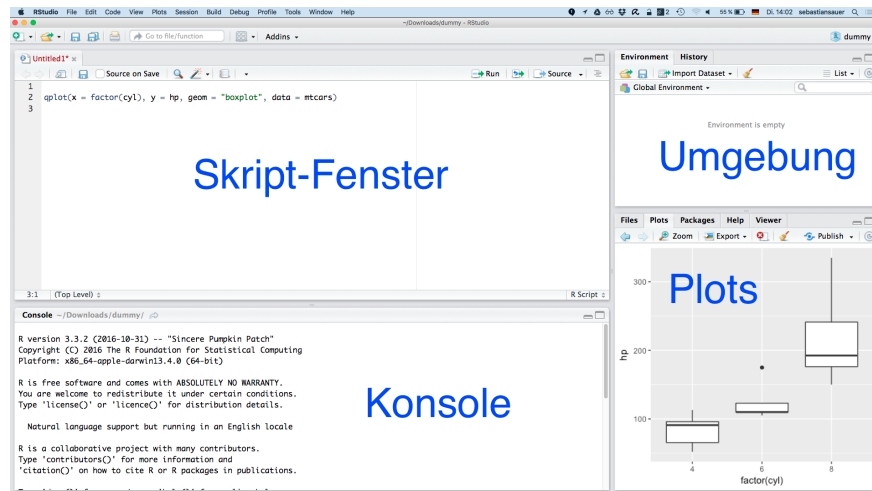


Abbildung 1.2: RStudio


### 1.1.1 R und RStudio installieren



Sie können R unter <https://cran.r-project.org> herunterladen und installieren (für Windows, Mac oder Linux). RStudio finden Sie auf der gleichnamigen Homepage: <https://www.rstudio.com>; laden Sie die “Desktop-Version” für Ihr Betriebssystem herunter.

RStudio ist “nur” eine Oberfläche (“GUI”) für R, mit einer Reihe von praktischen Zusatzfunktionen. Die eigentliche Arbeit verrichtet das “normale” R, welches automatisch gestartet wird, wenn Sie RStudio starten (sofern R installiert ist).

Die Oberfläche von RStudio sieht (unter allen Betriebssystemen etwa gleich) so aus wie in Abbildung 1.2 dargestellt.

Das *Skript-Fenster* ähnelt einem normalen Text-Editor; praktischerweise finden Sie aber einen Button “run”, der die aktuelle Zeile oder die Auswahl “abschickt”, d.h. in die Konsole gibt, wo die Syntax ausgeführt wird. Wenn Sie ein Skript-Fenster öffnen möchten, so können Sie das Icon  klicken (Alternativ: Ctrl-Shift-N oder File > New File > R Script).

Aus dem Fenster der *Konsole* spricht R zu uns bzw. wir mit R. Wird ein Befehl (synonym: *Funktion*) hier eingegeben, so führt R ihn aus. Es ist aber viel praktischer, Befehle in das Skript-Fenster einzugeben, als in die Konsole. Behalten Sie dieses Fenster im Blick, wenn Sie Antwort von R erwarten.

Im Fenster *Umgebung* (engl. “environment”) zeigt R, welche Variablen (Objekte) vorhanden

sind. Stellen Sie sich die Umgebung wie einen Karpfenteich vor, in dem die Datensätze und andere Objekte herumschwimmen. Was nicht in der Umgebung angezeigt wird, existiert nicht für R.

Im Fenster rechts unten werden mehrere Informationen bereit gestellt, z.B. werden Diagramme (Plots) dort ausgegeben. Klicken Sie mal die anderen Reiter im Fenster rechts unten durch.

Wer Shortcuts mag, wird in RStudio überschwänglich beschenkt; der Shortcut für die Shortcuts ist **Shift-Alt-K**.

Wenn Sie RStudio starten, startet R automatisch auch. Starten Sie daher, wenn Sie RStudio gestartet haben, *nicht* noch extra R. Damit hätten Sie sonst zwei Instanzen von R laufen, was zu Verwirrungen (bei R und beim Nutzer) führen kann.

### 1.1.2 Pakete

Ein Großteil der Neuentwicklungen bei R passiert in sog. ‘Paketen’ (engl. *packages*), das sind Erweiterungen für R. Jeder, der sich berufen fühlt, kann ein R-Paket schreiben und es zum ‘R-Appstore’ (CRAN<sup>1</sup>) hochladen. Von dort kann es dann frei (frei wie in Bier) heruntergeladen werden.

Am einfachsten installiert man R-Pakete in RStudio über den Button “Install” im Reiter “Packages” (s. Abb. 1.3).

Ein R-Paket, welches wir gleich benötigen, heißt **devtools**. Bitte installieren Sie es schon einmal (sofern noch nicht geschehen). Sie können auch folgenden Befehl verwenden, um Pakete zu installieren.

```
install.packages("devtools", dependencies = TRUE)
```

Aber einfacher geht es über die RStudio-Oberfläche.

Alle Pakete außer **devtools**, die wir hier benötigen, können über das R-Paket **prada** installiert werden. Sie müssen also nur noch das Paket **prada** installieren. Mit dem Befehl **install\_prada\_pckgs** (aus dem Paket **prada**) werden dann ggf. eine Reihe weiterer Pakete installiert. Allerdings wohnt **prada** nicht im R-Appstore (CRAN), sondern bei Github<sup>2,3</sup>. Um Pakete von Github zu installieren, nutzen wir diesen Befehl (Sie müssen natürlich online sein):

```
devtools::install_github("sebastiansauer/prada")  
library(prada)
```

---

<sup>1</sup><https://cran.r-project.org/>

<sup>2</sup>[www.github.com](https://www.github.com)

<sup>3</sup>einer Online-Plattform, auf der man Dateien bereistellen und ihre Veränderungen nachverfolgen kann

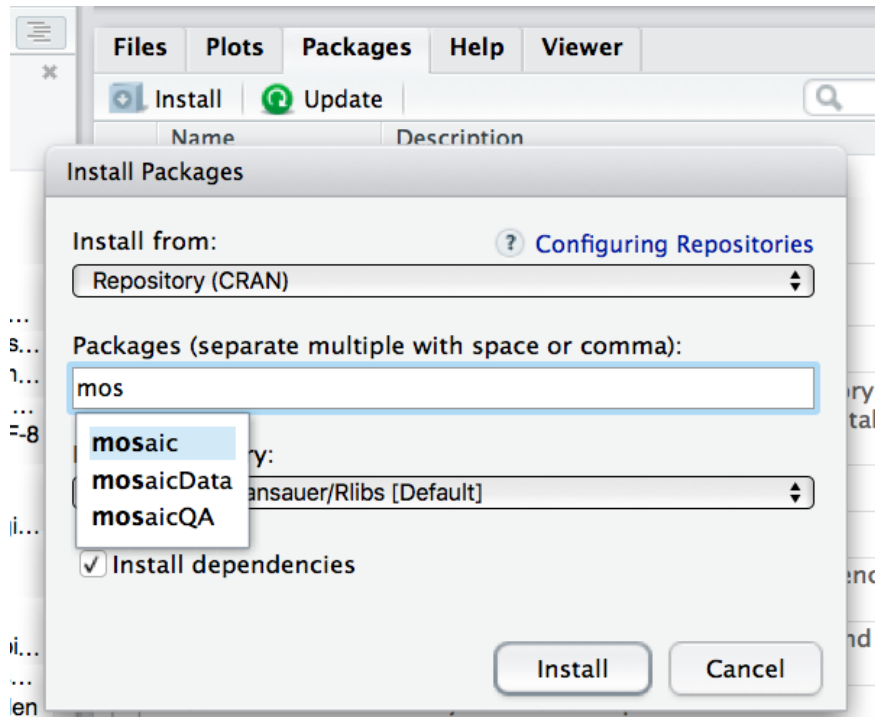


Abbildung 1.3: So installiert man Pakete in RStudio

Sofern Sie online sind, sollte das Paket **prada** jetzt installiert sein. Installieren Sie jetzt alle Pakete, die für diesen Kurs benötigt werden mit dem Befehl `install_prada_pckgs()`.



Beim Installieren von R-Paketen könnten Sie gefragt werden, welchen “Mirror” Sie verwenden möchten. Das hat folgenden Hintergrund: R-Pakete sind in einer Art “App-Store”, mit Namen CRAN (Comprehensive R Archive Network) gespeichert. Damit nicht ein armer, kleiner Server überlastet wird, wenn alle Studis dieser Welt just gerade beschließen, ein Paket herunterzuladen, gibt es viele Kopien dieses Servers - seine Spiegelbilder (engl. “mirrors”). Suchen Sie sich einfach einen aus, der in der Nähe ist.

Bei der Installation von Paketen mit `install.packages("name_des_pakets")` sollte stets der Parameter `dependencies = TRUE` angefügt werden. Also `install.packages("name_des_pakets", dependencies = TRUE)`. Hintergrund ist: Falls das zu installierende Paket seinerseits Pakete benötigt, die noch nicht installiert sind (gut möglich), dann werden diese sog. “dependencies” gleich mitinstalliert (wenn Sie `dependencies = TRUE` setzen).

Nicht vergessen: Installieren muss man eine Software *nur einmal*; *starten* (laden) muss man die R-Pakete jedes Mal, wenn man sie vorher geschlossen hat und wieder nutzen möchte.

Wenn Sie R bzw. RStudio schließen, werden alle Pakete ebenfalls geschlossen. Sie müssen die benötigten Pakete beim erneuten Öffnen von RStudio wieder starten.

```
library(dplyr)
```

Der Befehl bedeutet sinngemäß: “Hey R, geh in die Bücherei (library) und hole das Buch (package) dplyr!”.



Wann benutzt man bei R Anführungszeichen? Das ist etwas verwirrend im Detail, aber die Grundregel lautet: wenn man Text anspricht. Im Beispiel oben “library(dplyr)” ist “dplyr” hier erst mal für R nichts Bekanntes, weil noch nicht geladen. Demnach müssten *eigentlich* Anführungsstriche stehen. Allerdings meinte ein Programmierer, dass es doch so bequemer ist. Hat er Recht. Aber bedenken Sie, dass es sich um die Ausnahme einer Regel handelt. Sie können also auch schreiben: library(“dplyr”) oder library(‘dplyr’); beides geht.

### 1.1.3 Hilfe! R startet nicht!

Manntje, Manntje, Timpe Te,  
Buttje, Buttje inne See,  
myne Fru de Ilsebill  
will nich so, as ik wol will.

*Gebrüder Grimm, Märchen vom Fischer und seiner Frau*<sup>4</sup>

Ihr R startet nicht oder nicht richtig? Die drei wichtigsten Heilmittel sind:

1. Schließen Sie die Augen für eine Minute. Denken Sie an etwas Schönes und was Rs Problem sein könnte.
2. Schalten Sie den Rechner aus und probieren Sie es morgen noch einmal.
3. Googeln.

Sorry für die schnoddrigen Tipps. Aber: Es passiert allzu leicht, dass man *Fehler* wie diese macht:



OH NO:

- install.packages(dplyr)
- install.packages(“dliar”)
- install.packages(“derpyler”)
- install.packages(“dplyr”) # dependencies vergessen
- Keine Internet-Verbindung

---

<sup>4</sup>[https://de.wikipedia.org/wiki/Vom\\_Fischer\\_und\\_seiner\\_Frau](https://de.wikipedia.org/wiki/Vom_Fischer_und_seiner_Frau)

```
– library(dplyr) # ohne vorher zu installieren
```

Wenn R oder RStudio dann immer noch nicht starten oder nicht richtig laufen, probieren Sie dieses:

- Sehen Sie eine Fehlermeldung, die von einem fehlenden Paket spricht (z.B. “Package ‘Rcpp’ not available”) oder davon spricht, dass ein Paket nicht installiert werden konnte (z.B. “Package ‘Rcpp’ could not be installed” oder “es gibt kein Paket namens ‘Rcpp’” oder “unable to move temporary installation XXX to YYY”), dann tun Sie folgendes:
- Schließen Sie R und starten Sie es neu.
- Installieren Sie das oder die angesprochenen Pakete<sup>5</sup>.
- Starten Sie das entsprechende Paket mit `library(name_des_pakets)`.
- Gerade bei Windows 10 scheinen die Schreibrechte für R (und damit RStudio oder RCommander) eingeschränkt zu sein. Ohne Schreibrechte kann R aber nicht die Pakete (“packages”) installieren, die Sie für bestimmte R-Funktionen benötigen. Daher schließen Sie R bzw. RStudio und suchen Sie das Icon von R oder wenn Sie RStudio verwenden von RStudio. Rechtsklicken Sie das Icon und wählen Sie “als Administrator ausführen”. Damit geben Sie dem Programm Schreibrechte. Jetzt können Sie etwaige fehlende Pakete installieren.
- Ein weiterer Grund, warum R bzw. RStudio die Schreibrechte verwehrt werden könnten (und damit die Installation von Paketen), ist ein Virens Scanner. Der Virens Scanner sagt, nicht ganz zu Unrecht: “Moment, einfach hier Software zu installieren, das geht nicht, zu gefährlich”. Grundsätzlich gut, in diesem Fall unnötig. Schließen Sie R/RStudio und schalten Sie dann den Virens Scanner *komplett* (!) aus. Öffnen Sie dann R/RStudio wieder und versuchen Sie fehlende Pakete zu installieren.



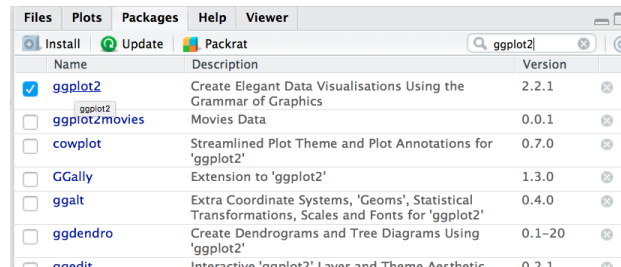
Verwenden Sie möglichst die neueste Version von R, RStudio und Ihres Betriebssystems. Ältere Versionen führen u.U. zu Problemen; je älter, desto Problem... Updaten Sie Ihre Packages regelmäßig z.B. mit `update.packages()` oder dem Button “Update” bei RStudio (Reiter Packages).

### 1.1.4 Vertiefung: Zuordnung von Paketen zu Befehlen

*Woher weiß man, welche Befehle (oder auch Daten) in einem Paket enthalten sind?*

Eine einfache Möglichkeit ist es, beim Reiter ‘Pakete’ auf den Namen eines der installierten Pakete zu klicken. Daraufhin öffnet sich die Dokumentation des Pakets und man sieht dort alle Befehle und Daten aufgeführt (s. Abbildung 1.4). Übrigens sehen Sie dort auch die Version

<sup>5</sup>`install.packages("name_des_pakets", dependencies = TRUE)` oder mit dem entsprechenden Klick in RStudio



**Abbildung 1.4:** Hier werden Sie geholfen: Die Dokumentation der R-Pakete

eines Pakets (vielleicht sagt jemand mal zu Ihnen, “Sie sind ja outdated”, dann schauen Sie mal auf die die Paket-Versionen).

Für geladenen Pakete kann man auch den Befehl `help` nutzen, z.B. `help(ggplot2)`.

*Und umgekehrt, woher weiß ich, in welchem Paket ein Befehl ‘wohnt’?*

Probieren Sie den Befehl `help.search("qplot")`, wenn Sie wissen möchten, in welchem Paket `qplot` zuhause ist. `help.search` sucht alle Hilfeseiten von *installierten* Paketen, in der der Suchbegriff irgendwie vorkommt. Um das Paket eines *geladenen* Befehl zu finden, hilft der Befehl `find::find("qplot")`.

Sie können auch den Befehl `find_funs` aus dem Paket `prada` nutzen:

```
prada::find_funs("select")
#> # A tibble: 5 x 3
#>   package_name builtin_pckage loaded
#>   <chr>          <lgl>   <lgl>
#> 1      dplyr      FALSE  FALSE
#> 2      MASS      TRUE   FALSE
#> 3     plotly      FALSE  FALSE
#> 4     raster      FALSE  FALSE
#> 5      VGAM      FALSE  FALSE
```

In diesem Skript sind am Ende jedes Kapitels die jeweils besprochenen (neuen) Befehle aufgeführt - inklusive ihres Paketes. Falls bei einem Befehl kein Paket angegeben ist, heißt das, dass der Befehl im ‘Standard-R’ wohnt - Sie müssen kein weiteres Paket laden<sup>6</sup>. Also zum Beispiel `ggplot2::qplot`: Der Befehl `qplot` ist im Paket `ggplot2` enthalten. Das Zeichen `::` trennt also Paket von Befehl.



Manche Befehle haben Allerweltsnamen (z.B. ‘filter’). Manchmal gibt es Befehle mit gleichem Namen in verschiedenen Paketen; besonders Befehle mit Allerweltsnamen (wie ‘filter’) sind betroffen (‘`mosaic::filter`’ vs. ‘`dplyr::filter`’). Falls Sie von wirre Ausgaben bekommen oder diffuse Fehlermeldung kann es sein, kann es sein, dass R einen Befehl

<sup>6</sup>Eine Liste der Pakete, die beim Standard-R enthalten sind (also bereits installiert sind) finden Sie hier<sup>7</sup>

mit dem richtigen Namen aber aus dem ‘falschen’ Paket zieht. Geben Sie im Zweifel lieber den Namen des Pakets vor dem Paketnamen an, z.B. so `dplyr::filter`. Der ‘doppelte Doppelpunkt’ trennt den Paketnamen vom Namen der Funktion.

Außerdem sind zu Beginn jedes Kapitels die in diesem Kapitel benötigten Pakete angegeben. Wenn sie diese Pakete laden, werden alle Befehle dieses Kapitels funktionieren<sup>8</sup>.

*Wie weiß ich, ob ein Paket geladen ist?*

Wenn der Haken im Reiter ‘Packages’ gesetzt ist (s. Abbildung 1.4), dann ist das Paket geladen. Sonst nicht.

### 1.1.5 Datensätze

Die folgenden Datensätze sind entweder im Paket **prada** enthalten oder können aus anderen Paketen geladen werden. Um Daten aus einem Paket zu laden, gibt es den Befehl `data("name_datenobjekt", package = "Paketname")`. Also zum Beispiel:

```
data("stats_test", package = "prada")
```

Wenn ein bestimmtes Paket geladen ist, können Sie auch auf den Parameter `package = ...` verzichten, wenn ihr Datensatz in jedem Paket wohnt: Geladene Pakete werden vom Befehl `data` automatisch durchsucht.

Alternativ können Sie die Daten auch im Ordner **data** im Github-Repository herunterladen. Gehen Sie auf die Github-Seite dieses Kurses<sup>9</sup>, klicken Sie auf den großen grünen Button “Clone or download”, wählen Sie dann “Download ZIP”, um alle Dateien herunterzuladen. Nach dem Entzippen können Sie dann auf alle Dateien, inklusive Daten, zugreifen.

Die Daten dieses Kurses liegen im Ordner ‘data’.

- Datensatz **profiles** aus dem R-Paket {okcupiddata} (Kim und Escobedo-Land 2015); es handelt sich um Daten von einer Online-Singlebörse
- Datensatz **stats\_test** aus dem R-Paket {prada} (Sauer 2017a); es handelt sich um Ergebnisse einer Statistikklausur (einer Probeklausur)
- Datensatz **flights** aus dem R-Paket {nycflights13} (RITA 2013); es handelt sich um Abflüge von den New Yorker Flughäfen
- Datensatz **wo\_men**, URL: <https://osf.io/ja9dw> (Sauer 2017b); es handelt sich um Körper- und Schuhgröße von Studierenden
- Datensatz **extra** aus dem R-Paket {prada} (Sauer 2016); es handelt sich die Ergebnisse einer Umfrage zu Extraversion
- Datensatz **titanic\_train** aus dem Paket {titanic} von kaggle<sup>10</sup>; es handelt sich um

<sup>8</sup>es sei denn, sie tun es nicht

<sup>9</sup>[https://github.com/sebastiansauer/Praxis\\_der\\_Datenanalyse](https://github.com/sebastiansauer/Praxis_der_Datenanalyse)

<sup>10</sup><https://www.kaggle.com/c/titanic/data>

**Tabelle 1.1:** Wichtige Datentypen in R

Name	Beschreibung	Beispiel
NULL	die leere Menge	NULL
logical	Logische Ausdrücke	TRUE
integer	Ganze Zahl	3
factor	nominale Variablen	"weiblich"
numeric	Reelle Zahl	2.71
character	Text	"Schorsch"

Überlebensraten vom Titanic-Unglück.

- Datensatz **Affairs** aus dem Paket {AER} (Fair 1978); es handelt sich um eine Umfrage zu außerehehlichen Affären.

Wie man Daten in R ‘einlädt’ (Studierende sagen gerne ‘ins R hochladen’), besprechen wir im Kapitel 2.

## 1.2 ERRRstkontakt

### 1.2.1 R-Skript-Dateien

Ein neues *R-Skript* im RStudio können Sie z.B. öffnen mit **File-New File-R Script**. Schreiben Sie dort Ihre R-Befehle; Sie können die Skriptdatei speichern, öffnen, ausdrucken, übers Bett hängen... R-Skripte können Sie speichern (unter **File-Save**) und öffnen. R-Skripte sind einfache Textdateien, die jeder Texteditor verarbeiten kann. Nur statt der Endung `.txt`, sind R-Skripte stolzer Träger der Endung `.R`. Es bleibt aber eine schnöde Textdatei. Geben Sie Ihren R-Skript-Dateien die Endung `“.R”`, damit erkennt RStudio, dass es sich um ein R-Skript handelt und bietet ein paar praktische Funktionen wie den “Run-Button”.

### 1.2.2 Datentypen in R

Die (für diesen Kurs) wichtigsten Datentypen von R sind in Tabelle 1.1 aufgeführt (vgl. (*Programmieren mit R* 2009)).

All diese Datentypen (mit Ausnahme der leeren Menge) sind als *Vektoren* angelegt, also mehreren Elementen (z.B. Zahlen), die zu einem Ganzen (wie in einer Liste) verknüpft sind. *Faktoren* sind ganz interessant, weil die einzelnen Ausprägungen (*Faktorstufen* genannt) für R als Zahlen gespeichert sind (z.B. “Frau Müller und Herr Schorsch” = 1). Wenn ein Vektor aus 100 Mal diesem Text (“Frau Müller...”) besteht, muss R nur 100 mal 1 speichern und einmal die Zuordnung, was die 1 bedeutet. Spart Speicher. Außerdem kann man definieren, was alles eine Faktorstufe ist (z.B. nur “Mann” und “Frau”). Andere Eingaben sind dann nicht



möglich; das kann praktisch sein, wenn man von vornerein nur bestimmte Ausprägungen zulassen möchte. Textvariablen sind da entspannter: Jeglicher Art von Text ist erlaubt. Text ist in R immer in Anführungszeichen (einfach oder doppelt) zu setzen.

Für die praktische Datenanalyse ist der **dataframe** (*Dataframe*; auch Datentabelle oder Datensatz) am wichtigsten. Grob gesagt handelt es sich dabei um eine Tabelle, wie man sie aus Excel kennt. Etwas genauer ist eine Kombination von Vektoren mit gleicher Länge, so dass eine ‘rechteckige’ Datenstruktur entsteht. Alle Spalten (d.h. Vektoren) haben einen Namen, so dass es ‘Spaltenköpfe’ gibt. Eine neuere Variante von Dataframes sind *tibbles* (Tibbles), die *auch* Dataframes sind, aber ein paar praktische Zusatzeigenschaften aufweisen (normale Dataframes können sich manchmal in einfache Vektoren auflösen; Tibbles tun dies nie).

### 1.2.3 Hinweise

Unser erster Kontakt mit R! Ein paar Anmerkungen vorweg:

- R unterscheidet zwischen Groß- und Kleinbuchstaben, d.h. `Oma` und `oma` sind zwei verschiedene Dinge für R!
- R verwendet den Punkt `.` als Dezimaltrennzeichen.
- Fehlende Werte werden in R durch `NA` kodiert.
- Kommentare werden mit dem Rautezeichen `#` eingeleitet; der Rest der Zeile von `#` wird dann ignoriert.
- *Variablen*namen in R sollten mit Buchstaben beginnen; ansonsten dürfen nur Zahlen und Unterstriche (`_`) enthalten sein. Leerzeichen sollte man meiden. Das gilt auch für Spaltennamen.
- Um den Inhalt einer Variablen auszulesen, geben wir einfach den Namen des Objekts ein (und schicken den Befehl `ab`).
- Bleiben Sie konsistent, in der Art und Weise, wie Sie Ihre Syntax schreiben. Ein Vorschlag zum ‘Syntax-Stil’ finden Sie hier<sup>11</sup>.
- Variablen einen treffenden Namen zu geben, ist nicht immer leicht, aber wichtig. Namen sollten knapp, aber aussagekräftig sein.

```
# so nicht:
var
x
dummy
objekt
dieser_name_ist_etwas_lang_vielleicht

# gut:
tips_mw
lm1
```

---

<sup>11</sup><http://adv-r.had.co.nz/Style.html>

### 1.2.4 Text und Variablen zuweisen

Man kann einer Variablen auch Text zuweisen (im Gegensatz zu Zahlen):

```
y <- "Hallo R!"
```

Man kann auch einer Variablen eine andere zuweisen:

```
y <- x
```

Wird jetzt y mit dem Inhalt von x überschrieben oder umgekehrt? Der Zuweisungspfeil <- macht die Richtung der Zuweisung ganz klar. Zwar ist in R das Gleichheitszeichen synonym zum Zuweisungspfeil erlaubt, aber der Zuweisungspfeil macht die Sache glasklar und sollte daher bevorzugt werden.

Man kann auch einer Variablen *mehr als* einen Wert zuweisen:

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

Dieser Befehl erzeugt eine “Spalte” (einen Vektor). Will man einer Variablen *mehr als* einen Wert zuweisen, muss man die Werte erst in einen Vektor “zusammen binden”; das geht mit dem Befehl c (vom engl. “*combine*”).

### 1.2.5 Funktionen aufrufen

Um einen *Befehl* (präziser aber synonym hier: eine Funktion) aufzurufen, geben wir ihren Namen an und definieren sog. *Parameter* in einer runden Klammer, z.B. so:

```
wo_men <- read.csv("data/wo_men.csv")
```

Allgemein gesprochen:

```
funktionsname(parametername1 = wert1, parametername2 = wert2, ...)
```

Die drei Punkte ... sollen andeuten, dass evtl. weitere Parameter zu übergeben wären. Die Reihenfolge der Parameter ist *egal* - wenn man die Parameternamen anführt. Ansonsten muss man sich an die Standard-Reihenfolge, die eine Funktion vorgibt halten:

```
#ok:  
wo_men <- read.csv(file = "data/wo_men.csv", header = TRUE, sep = ",")  
wo_men <- read.csv("data/wo_men.csv", TRUE, ",")  
wo_men <- read.csv(header = TRUE, sep = ",", file = "data/wo_men.csv")
```

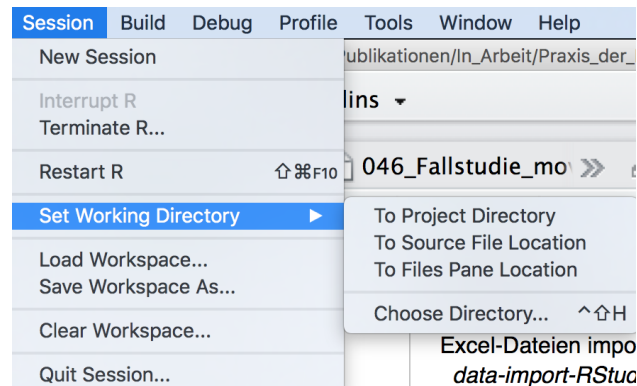


Abbildung 1.5: Das Arbeitsverzeichnis mit RStudio auswählen

```
# ohno:
wo_men <- read.csv(TRUE, "data/wo_men.csv", ",",")
```

In der Hilfe zu einem Befehl findet man die Standard-Syntax inklusive der möglichen Parameter, ihrer Reihenfolge und Standardwerten (default values) von Parametern. Zum Beispiel ist beim Befehl `read.csv` der Standardwert für `sep` mit `;` voreingestellt (schauen Sie mal in der Hilfe nach). Gibt man einen Parameter nicht an, für den ein Standardwert eingestellt ist, ‘befüllt’ R den Parameter mit diesem Standardwert.

### 1.2.6 Das Arbeitsverzeichnis

Das aktuelle Verzeichnis (Arbeitsverzeichnis; “working directory”) kann man mit `getwd()` erfragen und mit `setwd()` einstellen. Komfortabler ist es aber, das aktuelle Verzeichnis per Menü zu ändern (vgl. Abb. 1.5. In RStudio: **Session** > **Set Working Directory** > **Choose Directory ...** (oder per Shortcut, der dort angezeigt wird).

Es ist praktisch, das Arbeitsverzeichnis festzulegen, denn dann kann man z.B. eine Datendatei einlesen, ohne den Pfad eingeben zu müssen:

```
# nicht ausführen:
daten_deutsch <- read.csv("daten_deutsch.csv", sep = ";", dec = ".")
```

R geht dann davon aus, dass sich die Datei `daten_deutsch.csv` im Arbeitsverzeichnis befindet.

Für diesen Kurs ist es sinnvoll, das Arbeitsverzeichnis in einen “Hauptordner” zu legen (z.B. “Praxis\_der\_Datenanalyse”), in dem Daten und sonstiges Material als Unterordner abgelegt sind.



Übrigens: Wenn Sie keinen Pfad angeben, so geht R davon aus, dass die Daten im aktuellen Verzeichnis (dem *working directory*) liegen.

## 1.3 Hier werden Sie geholfen

### 1.3.1 Wo finde ich Hilfe?

Es ist keine Schande, nicht alle Befehle der ca. 10,000 R-Pakete auswendig zu wissen. Schlauer ist, zu wissen, wo man Antworten findet. Hier eine Auswahl:

- Zu diesen Paketen gibt es gute “Spickzettel” (cheatsheets): ggplot2, RMarkdown, dplyr, tidyr. Klicken Sie dazu in RStudio auf *Help > Cheatsheets > ...* oder gehen Sie auf <https://www.rstudio.com/resources/cheatsheets/>.
- In RStudio gibt es eine Reihe (viele) von Tastaturkürzeln (Shortcuts), die Sie hier finden: *Tools > Keyboard Shortcuts Help*.
- Für jeden Befehl aus einem *geladenen* Paket können Sie mit `help()` die Hilfe-Dokumentation anschauen, also z.B. `help("qplot")`.
- Im Internet finden sich zuhauf Tutorials.
- Der Reiter “Help” bei RStudio verweist auf die Hilfe-Seite des jeweiligen Pakets bzw. Befehls.
- Die bekannteste Seite um Fragen rund um R zu diskutieren ist <http://stackoverflow.com>.

### 1.3.2 Einfache reproduzierbare Beispiele (ERBies)

Sagen wir, Sie haben ein Problem. Mit R. Bevor Sie jemanden bitten, Ihr Problem zu lösen, haben Sie schon ~~drei dreizehn~~ dreißig Minuten recherchiert, ohne Erfolg. Sie entschließen sich, bei Stackoverflow<sup>12</sup> Ihr Problem zu posten. Außerdem kann sicher eine Mail zu einem Bekannten, einem Dozenten oder sonstwem, der sich auskennen sollte, nicht schaden. Sie formulieren also Ihr Problem: “Hallo, mein R startet nicht, und wenn es startet, dann macht es nicht, was ich soll, außerdem funktioniert der Befehl ‘mean’ bei mir nicht. Bitte lös mein Problem!”. Seltsamerweise reagieren die Empfänger Ihrer Nachricht nicht alle begeistert. Stattdessen verlangt jemand (dreist) nach einer genauen Beschreibung Ihres Problems, mit dem Hinweis, dass “Ferndiagnosen” schwierig sein. Genauer gesagt möchte ihr potenzieller Helfer ein ‘minimal reproducible example’ (MRE) oder, Deutsch, ein *einfaches reproduzierbares Beispiel* (ERBie).

<sup>12</sup>[www.stackoverflow.com](http://www.stackoverflow.com)

Wenn Sie jemanden um R-Hilfe bitten, dann sollten Sie Ihr Problem prägnant beschreiben.

Was sollte alles in einem ERBie enthalten sein?

Ein ERBie besteht aus vier Teilen: Syntax, Daten, Paketen und Infos zum laufenden System (R Version etc.)

Wie sollte so ein ERBie aussehen? Ich empfehle, folgende Eckpunkte zu beachten<sup>13</sup>:

- Syntax: Stellen Sie die R-Syntax bereit, die ein Problem bereit (d.h. die einen Fehler liefert).
- Einfach: Geben Sie so wenig Syntax wie möglich an. Es bereitet Ihrem Helfer nur wenig Spaß, sich durch 2000 Zeilen Code zu wühlen, wenn es 10 Zeilen auch getan hätten.
- Reproduzierbar Geben Sie soviel Syntax wie nötig, um den Fehler zu erzeugen (aber nicht mehr).
- Schreiben Sie Ihre Syntax übersichtlich, verständlich und kommentiert; z.B. sollten die Variablenamen informativ sein.
- Beschreiben Sie den Fehler genau ("läuft nicht" reicht nicht); z.B. ist es hilfreich, den Wortlaut einer Fehlermeldung bereitzustellen.
- Zu Beginn der Syntax sollten die benötigten Pakete geladen werden.
- Zu Ende des ERBie sollte der Output von `sessionInfo()` einkopiert werden; damit werden Informationen zum laufenden System (wie Version von R, Betriebssystem etc.) bereitgestellt.
- Beziehen Sie sich möglichst auf Daten, die in R schon "eingebaut sind" wie die Datensätze `iris` oder `mtcars`.

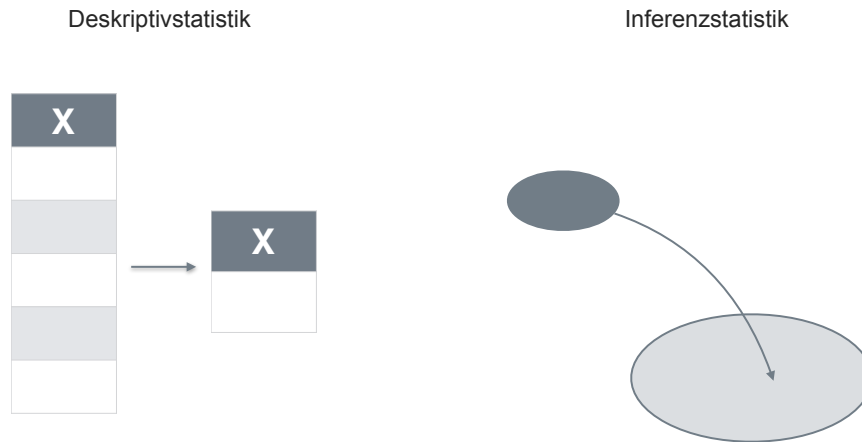
Natürlich sollte man immer erst selbst nach einer Lösung recherchieren, bevor man jemanden um Hilfe bittet. Viele Fragen wurden schon einmal diskutiert und oft auch gelöst.

## 1.4 Was ist Statistik? Wozu ist sie gut?

Zwei Fragen bieten sich sich am Anfang der Beschäftigung mit jedem Thema an: Was ist die Essenz des Themas? Warum ist das Thema (oder die Beschäftigung damit) wichtig?

Was ist Statistik? Eine Antwort dazu ist, dass Statistik die Wissenschaft von Sammlung, Analyse, Interpretation und Kommunikation von Daten ist mithilfe mathematischer Verfahren ist und zur Entscheidungshilfe beitragen sollte (*The Oxford Dictionary of Statistical Terms* 2006; Romeijn 2016). Damit hätten wir auch den Unterschied zur schönen Datenanalyse (ein Teil der Statistik) herausgemeißelt. Statistik wird häufig in die zwei Gebiete *deskriptive* und *inferierende* Statistik eingeteilt (vgl. Abb. 1.6). Erstere fasst viele Zahlen zusammen, so dass wir den Wald statt vieler Bäume sehen. Letztere verallgemeinert von den vorliegenden (sog. "Stichproben-") Daten auf eine zugrunde liegende Grundmenge (Population). Dabei spielt die Wahrscheinlichkeitsrechnung (Stochastik) eine große Rolle.

<sup>13</sup>Hier finden Sie weitere Hinweise zu ERBies: <https://stackoverflow.com/help/mcve> oder <https://gist.github.com/hadley/270442>



**Abbildung 1.6:** Sinnbild für die Deskriptiv- und die Inferenzstatistik

Aufgabe der deskriptiven Statistik ist es primär, Daten prägnant zusammenzufassen. Aufgabe der Inferenzstatistik ist es, zu prüfen, ob Daten einer Stichprobe auf eine Grundgesamtheit verallgemeinert werden können.

Dabei lässt sich der Begriff “Statistik” als Überbegriff von “Datenanalyse” verstehen, wenn diese Sicht auch nicht von allen geteilt wird (Grolemund und Wickham 2014). In diesem Buch steht die Aufbereitung, Analyse, Interpretation und Kommunikation von Daten im Vordergrund. Liegt der Schwerpunkt dieser Aktivitäten bei computerintensiven Methoden, so wird auch von *Data Science* gesprochen, wobei der Begriff nicht einheitlich verwendet wird (Wickham und Grolemund 2016; Hardin u. a. 2015)

*Daten* kann man definieren als *Informationen, die in einem Kontext stehen* (Moore 1990), wobei eine numerische Konnotation mitschwingt.

*Modellieren* kann man als *zentrale Aufgabe von Statistik* begreifen (Cobb 2007; Grolemund und Wickham 2014). Einfach gesprochen, bedeutet Modellieren in diesem Sinne, ein mathematisches Narrativ (“Geschichte”) zu finden, welches als Erklärung für gewisse Muster in den Daten fungiert; vgl. Kap. 8.

Statistisches Modellieren läuft gewöhnlich nach folgendem Muster ab (Grolemund und Wickham 2014):

Prämisse 1: Wenn Modell M wahr ist, dann sollten die Daten das Muster D aufweisen.

Prämisse 2: Die Daten weisen das Muster D auf.

---

Konklusion: Daher muss das Modell M wahr sein.

Die Konklusion ist *nicht* zwangsläufig richtig. Es ist falsch zu sagen, dass dieses Argumentationsmuster - Abduktion (Peirce 1955) genannt - wahre, sichere Schlüsse (Konklusionen) liefert. Die Konklusion *kann, muss aber nicht*, zutreffen.

Ein Beispiel: Auf dem Nachhauseweg eines langen Arbeitstags wartet, in einer dunklen Ecke, ein Mann, der sich als Statistik-Professor vorstellt und Sie zu einem Glücksspiel einlädt. Sofort sagen Sie zu. Der Statistiker will 10 Mal eine Münze werfen, er setzt auf Zahl (versteht

sich). Wenn er gewinnt, bekommt er 10€ von Ihnen; gewinnen Sie, bekommen Sie 11€ von ihm. Hört sich gut an, oder? Nun wirft er die Münze zehn Mal. Was passiert? Er gewinnt 10 Mal, natürlich (so will es die Geschichte). Sollten wir glauben, dass er ein Betrüger ist?

Ein Modell, welches wir hier verwenden könnten, lautet: Wenn die Münze gezinkt ist (Modell M zutrifft), dann wäre diese Datenlage D (10 von 10 Treffern) wahrscheinlich - Prämisse 1. Datenlage D ist tatsächlich der Fall; der Statistiker hat 10 von 10 Treffer erzielt - Prämisse 2. Die Daten D “passen” also zum Modell M; man entscheidet sich, dass der Professor ein Falschspieler ist.

Wichtig zu erkennen ist, dass Abduktion mit dem Wörtchen *wenn* beginnt. Also davon *ausgeht*, dass ein Modell M der Fall ist (der Professor also tatsächlich ein Betrüger ist). Das, worüber wir entscheiden wollen, wird bereits vorausgesetzt. Falls M gilt, gehen wir mal davon aus, wie gut passen dann die Daten dazu?

Wie gut passen die Daten D zum Modell M?

Das ist die Frage, die hier tatsächlich gestellt bzw. beantwortet wird.

Natürlich ist es keineswegs sicher, *dass* das Modell gilt. Darüber macht die Abduktion auch keine Aussage. Es könnte also sein, dass ein anderes Modell zutrifft: Der Professor könnte ein Heiliger sein, der uns auf etwas merkwürdige Art versucht, Geld zuzuschancen... Oder er hat einfach Glück gehabt.

Statistische Modelle beantworten i.d.R. nicht, wie wahrscheinlich es ist, dass ein Modell gilt. Statistische Modelle beurteilen, wie gut Daten zu einem Modell passen.

Häufig trifft ein Modell eine Reihe von Annahmen, die nicht immer explizit gemacht werden, aber die klar sein sollten. Z.B. sind die Münzwürfe unabhängig voneinander? Oder kann es sein, dass sich die Münze “einschießt” auf eine Seite? Dann wären die Münzwürfe nicht unabhängig voneinander. In diesem Fall klingt das reichlich unplausibel; in anderen Fällen kann dies eher der Fall sein<sup>14</sup>. Auch wenn die Münzwürfe unabhängig voneinander sind, ist die Wahrscheinlichkeit für Zahl jedes Mal gleich? Hier ist es wiederum unwahrscheinlich, dass sich die Münze verändert, ihre Masse verlagert, so dass eine Seite Unwucht bekommt. In anderen Situationen können sich Untersuchungsobjekte verändern (Menschen lernen manchmal etwas, sagt man), so dass die Wahrscheinlichkeiten für ein Ereignis unterschiedlich sein können, man dies aber nicht berücksichtigt.

## 1.5 Aufgaben

1. Öffnen Sie das Cheatsheet für RStudio und machen Sie sich mit dem Cheatsheet vertraut.

---

<sup>14</sup>Sind z.B. die Prüfungsergebnisse von Schülern unabhängig voneinander? Möglicherweise haben sie von einem “Superschüler” abgeschrieben. Wenn der Superschüler viel weiß, dann zeigen die Abschreiber auch gute Leistung.

2. Sichten Sie kurz die übrigen Cheatsheets; später werden die Ihnen vielleicht von Nutzen sein.
3. Führen Sie diese Syntax aus:

```
meine_coole_variable <- 10
meine_coole_variable
```

Woher rührt der Fehler?

4. Korrigieren Sie die Syntax:

```
install.packages(dplyer)
```

```
y <- Hallo R!
```

```
Hallo R <- 1
```

```
Hallo_R < - 1
```

Richtig oder Falsch???<sup>15</sup>



Richtig oder Falsch!?

1. Statistik wird gemeinhin in zwei Bereiche unterteilt: Deskriptivstatistik und Inferenzstatistik.
2. Unter Deskriptivstatistik versteht man, Daten zu beschreiben. Dazu ist jede Art von Beschreibung sinnvoll, vorausgesetzt es wird eine konsistente Regel eingesetzt.
3. Unter Abduktion versteht man den Schluss vom Allgemeinen auf das Konkrete.
4. Wirft jemand bei 10 von 10 Münzwürfen ‘Kopf’, so muss er ein Betrüger sein.
5. Wirft jemand bei 10 von 10 Münzwürfen ‘Kopf’, so ist die Wahrscheinlichkeit groß, dass er ein Betrüger ist.

## 1.6 Befehlsübersicht

Tabelle 1.2 stellt die Befehle dieses Kapitels dar.

Diese Befehle “wohnen” alle im Standard-R; es ist für diese Befehle nicht nötig, zusätzliche Pakete zu installieren/ laden.

---

<sup>15</sup>R, F: die Daten müssen sinnvoll zusammengefasst werden, F, F, F: Wenn er ehrlich sein sollte, dann ist das Ereignis ‘10 von 10’ selten



**Tabelle 1.2:** Befehle des Kapitels 'Rahmen'

Paket::Funktion	Beschreibung
<code>install.packages("x")</code>	Installiert Paket "x" (nicht: Paket "X")
<code>library</code>	lädt ein Paket
<code>&lt;-</code>	Weist einer Variablen einen Wert zu
<code>c</code>	erstellt eine Spalte/ einen Vektor

## 1.7 Verweise

- Chester Ismay erläutert einige Grundlagen von R und RStudio, die für Datenanalyse hilfreich sind: <https://bookdown.org/chesterismay/rbasics/>.
- Roger Peng und Kollegen bieten hier einen Einstieg in Data Science mit R: <https://bookdown.org/rdpeng/artofdatascience/>
- Wickham und Grolemund (2016) geben einen hervorragenden Überblick in das Thema dieses Buches; ihr Buch ist sehr zu empfehlen.
- Wer einen stärker an der Statistik orientierten Zugang sucht, aber “mathematisch sanft” behandelt werden möchte, wird bei James et al. (2013b) glücklich oder zumindest fündig werden.
- Uwe Ligges (*Programmieren mit R* 2009) ‘Programmieren mit R’ gibt einen tieferen Einstieg in die Grundlagen von R.
- Wer ganz tief ein- und abtauchen möchte in R, dem sei - solide Grundkenntnisse vorausgesetzt - Hadley Wickhams Wickham (2014a) ‘Advanced R’ ans Herz gelegt.



## Kapitel 2

### Daten einlesen



# Kapitel 3

## Datenjudo



# Kapitel 4

## Praxisprobleme der Datenaufbereitung





# Kapitel 5

## Fallstudie ‘movies’



# Kapitel 6

## Daten visualisieren



# Kapitel 7

## Fallstudie zur Visualisierung



# Kapitel 8

## Grundlagen des Modellierens





## Kapitel 9

# Der p-Wert, Inferenzstatistik und Alternativen



# Kapitel 10

## Lineare Regression



# Kapitel 11

## Klassifizierende Regression



# Kapitel 12

## Fallstudien zum geleiteten Modellieren





# Kapitel 13

## Vertiefung: Clusteranalyse



# Kapitel 14

## Vertiefung: Dimensionsreduktion



# Kapitel 15

## Vertiefung: Grundlagen des Textmining



# Kapitel 16

## Placeholder





# Kapitel 17

## Probeklausur



# Kapitel 18

## Hinweise



# Kapitel 19

## Literaturverzeichnis

Allaire, JJ, Joe Cheng, Yihui Xie, Jonathan McPherson, Winston Chang, Jeff Allen, Hadley Wickham, Aron Atkins, und Rob Hyndman. 2016a. *rmarkdown: Dynamic Documents for R*. <https://CRAN.R-project.org/package=rmarkdown>.

———. 2016b. *rmarkdown: Dynamic Documents for R*. <https://CRAN.R-project.org/package=rmarkdown>.

Auguie, Baptiste. 2016. *gridExtra: Miscellaneous Functions for „Grid“ Graphics*. <https://CRAN.R-project.org/package=gridExtra>.

Beaujean, A. Alexander. 2012. *BaylorEdPsych: R Package for Baylor University Educational Psychology Quantitative Courses*. <https://CRAN.R-project.org/package=BaylorEdPsych>.

Benoit, Kenneth, und Paul Nulty. 2016. *quanteda: Quantitative Analysis of Textual Data*. <https://CRAN.R-project.org/package=quanteda>.

Bouchet-Valat, Milan. 2014. *SnowballC: Snowball stemmers based on the C libstemmer UTF-8 library*. <https://CRAN.R-project.org/package=SnowballC>.

Briggs, William M. 2008a. *Breaking the Law of Averages: Real-Life Probability and Statistics in Plain English*. Lulu.com.

———. 2008b. *Breaking the Law of Averages: Real-Life Probability and Statistics in Plain English*. Lulu.com. <https://www.amazon.com/Breaking-Law-Averages-Probability-Statistics/dp/0557019907?SubscriptionId=0JYN1NVW651KCA56C102&tag=techkie-20&linkCode=xm2&camp=2025&creative=165953&creativeASIN=0557019907>.

———. 2016. *Uncertainty: The Soul of Modeling, Probability & Statistics*. Springer.

Bryant, PG, und MA Smith. 1995. „Practical Data Analysis: Case Studies in Business Statistics, Homewood, IL: Richard D“. Irwin Publishing.

Chang, Winston. 2015. *downloader: Download Files over HTTP and HTTPS*. <https://CRAN.R-project.org/package=downloader>.

Chapman, Chris, und Elea McDonnell Feit. 2015. *R for Marketing Research and Analytics*.

New York City: Springer. doi:10.1007/978-3-319-14436-8<sup>1</sup>.

Cleveland, William S. 1993. *Visualizing Data*. Hobart Press.

Clopper, Charles J, und Egon S Pearson. 1934. „The use of confidence or fiducial limits illustrated in the case of the binomial“. *Biometrika* 26 (4). JSTOR: 404–13.

Cobb, George W. 2007. „The introductory statistics course: a Ptolemaic curriculum?“. *Technology Innovations in Statistics Education* 1 (1).

Cohen, J. 1992. „A power primer“. *Psychological Bulletin* 112 (1): 155–59.

Cohen, Jacob. 1988. *Statistical Power Analysis for the Behavioral Sciences*. Routledge. <http://dx.doi.org/10.4324/9780203771587>.

Cortez, Paulo, António Cerdeira, Fernando Almeida, Telmo Matos, und José Reis. 2009. „Modeling wine preferences by data mining from physicochemical properties“. *Decision Support Systems* 47 (4). Elsevier: 547–53.

de Vries, Andrie, und Brian D. Ripley. 2016. *ggdendro: Create Dendrograms and Tree Diagrams Using 'ggplot2'*. <https://CRAN.R-project.org/package=ggdendro>.

Diez, David M, Christopher D Barr, und Mine Cetinkaya-Rundel. 2014. *Introductory Statistics with Randomization and Simulation*. North Charleston, South Carolina: CreateSpace Independent Publishing Platform.

Eid, Michael, Mario Gollwitzer, und Manfred Schmitt. 2010. *Statistik und Forschungsmethoden*. Göttingen: Hogrefe.

Etz, Alexander, Quentin Frederik Gronau, Fabian Dablander, Peter Edelsbrunner, und Beth Baribault. 2016. „How to become a Bayesian in eight easy steps: An annotated reading list“. PsyArXiv.

Fair, Ray C. 1978. „A theory of extramarital affairs“. *Journal of Political Economy* 86 (1). The University of Chicago Press: 45–61.

Feinerer, Ingo, und Kurt Hornik. 2015. *tm: Text Mining Package*. <https://CRAN.R-project.org/package=tm>.

Fellows, Ian. 2014. *wordcloud: Word Clouds*. <https://CRAN.R-project.org/package=wordcloud>.

Fjalnes. 2014. „Orthogonale Faktorrotation“. [https://de.wikipedia.org/wiki/Rotationsverfahren\\_\(Statistik\)#/media/File:Orthogonale\\_faktorrotation.svg](https://de.wikipedia.org/wiki/Rotationsverfahren_(Statistik)#/media/File:Orthogonale_faktorrotation.svg).

Fox, John, und Sanford Weisberg. 2016. *car: Companion to Applied Regression*. <https://CRAN.R-project.org/package=car>.

Gansser, Oliver. 2017. „Data for Principal Component Analysis and Common Factor Analysis“. Open Science Framework. [osf.io/zg89r](https://osf.io/zg89r).

Gigerenzer, Gerd. 1980. *Messung und Modellbildung in der Psychologie (Uni-Taschenbücher*.

---

<sup>1</sup><https://doi.org/10.1007/978-3-319-14436-8>

- Psychologie, Padagogik, Soziologie, Psychiatrie*) (German Edition). E. Reinhardt.
- . 2004. „Mindless statistics“. *The Journal of Socio-Economics* 33 (5). Elsevier BV: 587–606. doi:10.1016/j.socec.2004.09.033<sup>2</sup>.
- God. 2016. „I don’t care about you. Please share this with friends.“ Twitter Tweet. *TheTweetOfGod*. <https://twitter.com/TheTweetOfGod/status/688035049187454976>.
- Grolemund, Garrett, und Hadley Wickham. 2014. „A cognitive interpretation of data analysis“. *International Statistical Review* 82 (2). Wiley Online Library: 184–204.
- Hahsler, Michael, Christian Buchta, Bettina Gruen, und Kurt Hornik. 2016. *arules: Mining Association Rules and Frequent Itemsets*. <https://CRAN.R-project.org/package=arules>.
- Hahsler, Michael, und Sudheer Chelluboina. 2016. *arulesViz: Visualizing Association Rules and Frequent Itemsets*. <https://CRAN.R-project.org/package=arulesViz>.
- Hamermesh, Daniel S, und Amy Parker. 2005. „Beauty in the classroom: Instructors’ pulchritude and putative pedagogical productivity“. *Economics of Education Review* 24 (4). Elsevier: 369–76.
- Hardin, Johanna, Roger Hoerl, Nicholas J Horton, Deborah Nolan, Ben Baumer, Olaf Hall-Holt, Paul Murrell, u. a. 2015. „Data science in statistics curricula: Preparing students to ‘Think with Data’“. *The American Statistician* 69 (4). Taylor & Francis: 343–53.
- Hatzinger, Reinhold, Kurt Hornik, und Herbert Nagel. 2014. *R- Einfuehrung durch angewandte Statistik*. Pearson Studium.
- Head, Megan L., Luke Holman, Rob Lanfear, Andrew T. Kahn, und Michael D. Jennions. 2015. „The Extent and Consequences of P-Hacking in Science“. *PLOS Biology* 13 (3). Public Library of Science (PLoS): e1002106. doi:10.1371/journal.pbio.1002106<sup>3</sup>.
- Hendricks, Paul. 2015. *titanic: Titanic Passenger Survival Data Set*. <https://CRAN.R-project.org/package=titanic>.
- Hoekstra, Rink, Richard D Morey, Jeffrey N Rouder, und Eric-Jan Wagenmakers. 2014. „Robust misinterpretation of confidence intervals“. *Psychonomic bulletin & review* 21 (5). Springer: 1157–64.
- Hyndman, R.J., und G. Athanasopoulos. 2014. *Forecasting: principles and practice*: OTexts. <https://books.google.de/books?id=gDuRBAAAQBAJ>.
- Icon-Pond. 2016. „Education. 35 Icons.“ Flaticon. <http://www.flaticon.com/authors/popcorns-arts>.
- Ingo Feinerer, Kurt Hornik, und David Meyer. 2008. „Text Mining Infrastructure in R“. *Journal of Statistical Software* 25 (5): 1–54. <http://www.jstatsoft.org/v25/i05/>.
- Jackson, Simon. 2016. *corrr: Correlations in R*. <https://CRAN.R-project.org/package=corrr>.

<sup>2</sup><https://doi.org/10.1016/j.socec.2004.09.033>

<sup>3</sup><https://doi.org/10.1371/journal.pbio.1002106>

corrrr.

James, Gareth, Daniela Witten, Trevor Hastie, und Rob Tibshirani. 2013a. *ISLR: Data for An Introduction to Statistical Learning with Applications in R*. <https://CRAN.R-project.org/package=ISLR>.

James, Gareth, Daniela Witten, Trevor Hastie, und Robert Tibshirani. 2013b. *An introduction to statistical learning*. Bd. 6. Springer.

———. 2013c. *An introduction to statistical learning*. Bd. 6. Springer.

Julia, PhD Silge, und PhD Robinson David. 2017. *Text Mining with R: A tidy approach*. O'Reilly Media.

Kerby, Dave S. 2014. „The Simple Difference Formula: An Approach to Teaching Nonparametric Correlation“. *Comprehensive Psychology* 3: 11.IT.3.1. doi:10.2466/11.IT.3.1<sup>4</sup>.

Kim, Albert Y, und Adriana Escobedo-Land. 2015. „OkCupid Data for Introductory Statistics and Data Science Courses“. *Journal of Statistics Education* 23 (2). Citeseer: n2.

Kim, Albert Y., und Adriana Escobedo-Land. 2016. *okcupiddata: OkCupid Profile Data for Introductory Statistics and Data Science Courses*. <https://CRAN.R-project.org/package=okcupiddata>.

Krämer, W. 2011. *Wie wir uns von falschen Theorien täuschen lassen*. Berlin University Press. <https://books.google.de/books?id=HWUKaAEACAAJ>.

Kruschke, John K. 2010. „Bayesian data analysis“. *Wiley Interdisciplinary Reviews: Cognitive Science* 1 (5). Burlington, MA: Academic Press: 658–76.

Kuhn, Max, und Kjell Johnson. 2013. *Applied predictive modeling*. Bd. 26. Springer.

Ligges, Uwe, Martin Maechler, und Sarah Schnackenberg. 2017. *scatterplot3d: 3D Scatter Plot*. <https://CRAN.R-project.org/package=scatterplot3d>.

Lübke, Karsten, und Martin Vogt. 2014. *Angewandte Wirtschaftsstatistik: Daten und Zufall*. Berlin: Springer.

M7. 2004. „Savinelli’s Italian smoking pipe“. [https://commons.wikimedia.org/wiki/File:Pipa\\_savinelli.jpg](https://commons.wikimedia.org/wiki/File:Pipa_savinelli.jpg).

Matejka, Justin, und George Fitzmaurice. 2017. „Same stats, different graphs: Generating datasets with varied appearance and identical statistics through simulated annealing“. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, 1290–4. ACM.

Micceri, Theodore. 1989. „The unicorn, the normal curve, and other improbable creatures“. *Psychological Bulletin* 105 (1): 156–66. doi:10.1037/0033-2909.105.1.156<sup>5</sup>.

Michell, Joel. 2000. „Normal Science, Pathological Science and Psychometrics“. *Theory &*

<sup>4</sup><https://doi.org/10.2466/11.IT.3.1>

<sup>5</sup><https://doi.org/10.1037/0033-2909.105.1.156>



*Psychology* 10 (5): 639–67.

Milborrow, Stephen. 2017. *rpart.plot: Plot 'rpart' Models: An Enhanced Version of 'plot.rpart'*. <https://CRAN.R-project.org/package=rpart.plot>.

Moore, David S. 1990. „Uncertainty“. *On the shoulders of giants: New approaches to numeracy*. ERIC, 95–137.

Mullen, Lincoln. 2016. *tokenizers: A Consistent Interface to Tokenize Natural Language Text*. <https://CRAN.R-project.org/package=tokenizers>.

Neuwirth, Erich. 2014. *RColorBrewer: ColorBrewer Palettes*. <https://CRAN.R-project.org/package=RColorBrewer>.

Neyman, J., und E. S. Pearson. 1933. „On the Problem of the Most Efficient Tests of Statistical Hypotheses“. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 231 (694–706): 289–337. doi:10.1098/rsta.1933.0009<sup>6</sup>.

Neyman, Jerzy. 1935. „On the problem of confidence intervals“. *The annals of mathematical statistics* 6 (3). JSTOR: 111–16.

Neyman, Jerzy, und Egon S Pearson. 1992. „On the problem of the most efficient tests of statistical hypotheses“. In *Breakthroughs in statistics*, 73–108. New York City: Springer.

Ooms, Jeroen. 2016. *pdftools: Text Extraction and Rendering of PDF Documents*. <https://CRAN.R-project.org/package=pdftools>.

Peirce, Charles S. 1955. „Abduction and induction“. *Philosophical writings of Peirce* 11. New York.

Peng, Roger D, und Elizabeth Matsui. 2015. „The Art of Data Science“. *A Guide for Anyone Who Works with Data*. Skybrude Consulting 200: 162.

Prel, Jean-Baptist du, Bernd Roehrig, Gerhard Hommel, und Maria Blettner. 2010. *Deutsches Aerzteblatt Online*, Mai. Deutscher Aerzte-Verlag. doi:10.3238/arztebl.2010.0343<sup>7</sup>.

*Programmieren mit R*. 2009. Springer Berlin Heidelberg. doi:10.1007/978-3-540-79998-6<sup>8</sup>.

Raiche, Gilles, und David Magis. 2011. *nFactors: Parallel Analysis and Non Graphical Solutions to the Cattell Scree Test*. <https://CRAN.R-project.org/package=nFactors>.

Ram, Karthik, und Hadley Wickham. 2015. *wesanderson: A Wes Anderson Palette Generator*. <https://CRAN.R-project.org/package=wesanderson>.

Re, AC Del. 2014. *compute.es: Compute Effect Sizes*. <https://CRAN.R-project.org/package=compute.es>.

Remus, R., U. Quasthoff, und G. Heyer. 2010. „SentiWS – a Publicly Available German-language Resource for Sentiment Analysis“. In *Proceedings of the 7th International Language*

<sup>6</sup><https://doi.org/10.1098/rsta.1933.0009>

<sup>7</sup><https://doi.org/10.3238/arztebl.2010.0343>

<sup>8</sup><https://doi.org/10.1007/978-3-540-79998-6>

*Resources and Evaluation (LREC'10)*, 1168–71.

Ripley, Brian. 2016. *MASS: Support Functions and Datasets for Venables and Ripley's MASS*. <https://CRAN.R-project.org/package=MASS>.

RITA, Bureau of transportation statistics. 2013. „nycflights13“. [http://www.transtats.bts.gov/DL\\_SelectFields.asp?Table\\_ID=236](http://www.transtats.bts.gov/DL_SelectFields.asp?Table_ID=236).

Robinson, David. 2016. *gutenbergr: Download and Process Public Domain Works from Project Gutenberg*. <https://cran.rstudio.com/package=gutenbergr>.

Robinson, David, Matthieu Gomez, Boris Demeshev, Dieter Menne, Benjamin Nutter, Luke Johnston, Ben Bolker, Francois Briatte, und Hadley Wickham. 2015. *broom: Convert Statistical Analysis Objects into Tidy Data Frames*. <https://CRAN.R-project.org/package=broom>.

Robinson, David, und Julia Silge. 2016. *tidytext: Text Mining using 'dplyr', 'ggplot2', and Other Tidy Tools*. <https://CRAN.R-project.org/package=tidytext>.

Romeijn, Jan-Willem. 2016. „Philosophy of Statistics“. In *The Stanford Encyclopedia of Philosophy*, herausgegeben von Edward N. Zalta, Winter 2016. <http://plato.stanford.edu/archives/win2016/entries/statistics/>.

Rucker, Rudy. 2004. *Infinity and the Mind*. Princeton: Princeton University Press. <https://books.google.de/books?id=MDOUAAQBAJ>.

Sauer, Sebastian. 2016. „Extraversion Dataset“. Open Science Framework. doi:10.17605/OSF.IO/4KGZH<sup>9</sup>.

———. 2017a. „Dataset 'predictors of performance in stats test'“. Open Science Framework. doi:10.17605/OSF.IO/SJHUY<sup>10</sup>.

———. 2017b. „Dataset 'Height and shoe size'“. Open Science Framework. doi:10.17605/OSF.IO/JA9DW<sup>11</sup>.

Sauer, Sebastian, und Alexander Wolff. 2016. „The effect of a status symbol on success in online dating: an experimental study (data paper)“. *The Winnower*, August. doi:10.15200/winn.147241.13309<sup>12</sup>.

Sauer, Sebastian, Harald Walach, und Niko Kohls. 2010. „Gray's Behavioural Inhibition System as a mediator of mindfulness towards well-being“. *Personality and Individual Differences* 50 (4). Pergamon: 506–51. doi:10.1016/j.paid.2010.11.019<sup>13</sup>.

Schloerke, Barret, Jason Crowley, Di Cook, Francois Briatte, Moritz Marbach, Edwin Thoen, Amos Elberg, und Joseph Larmarange. 2016. *GGally: Extension to 'ggplot2'*. <https://CRAN.R-project.org/package=GGally>.

Schmidt, Peter, Sebastian Bamberg, Eldad Davidov, Johannes Herrmann, und Shalom H. Schwartz. 2007. „Die Messung von Werten mit dem Portraits Value Questionnaire“.

<sup>9</sup><https://doi.org/10.17605/OSF.IO/4KGZH>

<sup>10</sup><https://doi.org/10.17605/OSF.IO/SJHUY>

<sup>11</sup><https://doi.org/10.17605/OSF.IO/JA9DW>

<sup>12</sup><https://doi.org/10.15200/winn.147241.13309>

<sup>13</sup><https://doi.org/10.1016/j.paid.2010.11.019>

*Zeitschrift für Sozialpsychologie* 38 (4). Hogrefe Publishing Group: 261–75. doi:10.1024/0044-3514.38.4.261<sup>14</sup>.

Shmueli, Galit. 2010. „To Explain or to Predict?“, *Statistical Science* 25 (3): 289–310. doi:10.1214/10-STS330<sup>15</sup>.

Silge, Julia. 2016. *janeaustenr: Jane Austen’s Complete Novels*. <https://CRAN.R-project.org/package=janeaustenr>.

Silge, Julia, David Robinson, und Jim Hester. 2016. „tidytext: Text mining using dplyr, ggplot2, and other tidy tools“. doi:10.5281/zenodo.56714<sup>16</sup>.

Silge, Julia, und David Robinson. 2016. „tidytext: Text Mining and Analysis Using Tidy Data Principles in R“. *The Journal of Open Source Software* 1 (3). The Open Journal. doi:10.21105/joss.00037<sup>17</sup>.

Spurzem, Lothar. 2017. „VW 1303 von Wiking in 1:87“. [https://de.wikipedia.org/wiki/Modellautomobil#/media/File:Wiking-Modell\\_VW\\_1303\\_\(um\\_1975\).JPG](https://de.wikipedia.org/wiki/Modellautomobil#/media/File:Wiking-Modell_VW_1303_(um_1975).JPG).

Suppes, Patrick, und Joseph L Zinnes. 1962. *Basic measurement theory*. Institute for mathematical studies in the social sciences.

*The Oxford Dictionary of Statistical Terms*. 2006. Oxford University Press.

Therneau, Terry, Beth Atkinson, und Brian Ripley. 2015. *rpart: Recursive Partitioning and Regression Trees*. <https://CRAN.R-project.org/package=rpart>.

Tufte, Edward R. 1990. *Envisioning Information*. Graphics Press.

———. 2001. *The Visual Display of Quantitative Information*. Graphics Press.

———. 2006. *Beautiful Evidence*. Graphics Press.

Unrau, Sebastian. 2017. „No Title“. <https://unsplash.com/photos/CoD2Q92UaEg>.

VanDerWal, Jeremy, Lorena Falconi, Stephanie Januchowski, Luke Shoo, und Collin Storlie. 2014. *SDMTools: Species Distribution Modelling Tools: Tools for processing data associated with species distribution modelling exercises*. <https://CRAN.R-project.org/package=SDMTools>.

Wagenmakers, Eric-Jan. 2007. „A practical solution to the pervasive problems of p values“. *Psychonomic Bulletin & Review* 14 (5). Springer Nature: 779–804. doi:10.3758/bf03194105<sup>18</sup>.

Warnes, Gregory R., Ben Bolker, Lodewijk Bonebakker, Robert Gentleman, Wolfgang Huber, Andy Liaw, Thomas Lumley, Martin Maechler, u. a. 2016. *gplots: Various R Programming*

<sup>14</sup><https://doi.org/10.1024/0044-3514.38.4.261>

<sup>15</sup><https://doi.org/10.1214/10-STS330>

<sup>16</sup><https://doi.org/10.5281/zenodo.56714>

<sup>17</sup><https://doi.org/10.21105/joss.00037>

<sup>18</sup><https://doi.org/10.3758/bf03194105>

*Tools for Plotting Data*. <https://CRAN.R-project.org/package=gplots>.

Wei, Taiyun, und Viliam Simko. 2016. *corrplot: Visualization of a Correlation Matrix*. <https://CRAN.R-project.org/package=corrplot>.

Wicherts, Jelte M., Coosje L. S. Veldkamp, Hilde E. M. Augusteijn, Marjan Bakker, Robbie C. M. van Aert, und Marcel A. L. M. van Assen. 2016. „Degrees of Freedom in Planning, Running, Analyzing, and Reporting Psychological Studies: A Checklist to Avoid p-Hacking“. *Frontiers in Psychology* 7 (November). Frontiers Media SA. doi:10.3389/fpsyg.2016.01832<sup>19</sup>.

Wickham, Hadley. 2009. *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. <http://ggplot2.org>.

———. 2014a. *Advanced R*. Boca Raton, Florida: CRC Press.

———. 2014b. „Tidy Data“. *Journal of Statistical Software* 59 (1): 1–23. doi:10.18637/jss.v059.i10<sup>20</sup>.

———. 2016a. *reshape2: Flexibly Reshape Data: A Reboot of the Reshape Package*. <https://CRAN.R-project.org/package=reshape2>.

———. 2016b. *tidyr: Easily Tidy Data with ‘spread()’ and ‘gather()’ Functions*. <https://CRAN.R-project.org/package=tidyr>.

———. 2017a. *nycflights13: Flights that Departed NYC in 2013*. <https://CRAN.R-project.org/package=nycflights13>.

———. 2017b. *stringr: Simple, Consistent Wrappers for Common String Operations*. <https://CRAN.R-project.org/package=stringr>.

———. 2017c. *tidyverse: Easily Install and Load ‘Tidyverse’ Packages*. <https://CRAN.R-project.org/package=tidyverse>.

Wickham, Hadley, Jim Hester, und Romain Francois. 2016a. *readr: Read Tabular Data*. <https://CRAN.R-project.org/package=readr>.

———. 2016b. *readr: Read Tabular Data*. <https://CRAN.R-project.org/package=readr>.

Wickham, Hadley, und Romain Francois. 2016. *dplyr: A Grammar of Data Manipulation*. <https://CRAN.R-project.org/package=dplyr>.

Wickham, Hadley, und Garrett Golemund. 2016. *R for Data Science: Visualize, Model, Transform, Tidy, and Import Data*. O’Reilly Media.

Wikipedia. 2017. „Körpergröße — Wikipedia, Die freie Enzyklopädie“. <https://de.wikipedia.org/w/index.php?title=K%C3%B6rpergr%C3%B6%C3%9Fe&oldid=165047921>.

Wild, Chris J, und Maxine Pfannkuch. 1999. „Statistical thinking in empirical enquiry“. *International Statistical Review* 67 (3). Wiley Online Library: 223–48.

Wild, Fridolin. 2015. *lsa: Latent Semantic Analysis*. <https://CRAN.R-project.org/>

<sup>19</sup><https://doi.org/10.3389/fpsyg.2016.01832>

<sup>20</sup><https://doi.org/10.18637/jss.v059.i10>

`package=lsa.`

Wilkinson, Leland. 2006. *The grammar of graphics*. Springer Science & Business Media.

Xie, Yihui. 2015. *Dynamic Documents with R and knitr*. 2nd Aufl. Boca Raton, Florida: Chapman; Hall/CRC. <http://yihui.name/knitr/>.

———. 2016. *knitr: A General-Purpose Package for Dynamic Report Generation in R*. <https://CRAN.R-project.org/package=knitr>.

Zumel, Nina, John Mount, und Jim Porzak. 2014. *Practical data science with R*. Manning.

# Index

Befehl, Funktion, [14](#)

Dataframe, [13](#)

einfaches reproduzierbares Beispiel, [16](#)

Faktorstufen, [12](#)

Funktion, [5](#)

Konsole, [5](#)

Parameter eines R-Befehls, [14](#)

R-Skript, [12](#)

Skript-Fenster, [5](#)

Tibbles, [13](#)

Umgebung, [5](#)

Variablen, [13](#)

Vektoren, [12](#)