O'REILLY®

R

kurz & gut

O'Reillys Taschenbibliothek

Jörg Staudemeyer,
Ralf C. Staudemeyer

R

kurz & gut

Jörg Staudemeyer,
Ralf C. Staudemeyer

O'REILLY®

Jörg Staudemeyer, Ralf C. Staudemeyer

Lektorat: Alexandra Follenius
Fachgutachten: Joachim Zuckarelli
Korrektorat: Sibylle Feldmann, *www.richtiger-text.de*
Satz: III-satz, *www.drei-satz.de*
Herstellung: Stefanie Weidner
Umschlaggestaltung: Michael Oréal, *www.oreal.de*

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über *http://dnb.d-nb.de* abrufbar.

ISBN
Print 978-3-96009-133-2
PDF 978-3-96010-471-1
ePub 978-3-96010-472-8
mobi 978-3-96010-473-5

1. Auflage 2022

Wieblinger Weg 17
69123 Heidelberg

Dieses Buch erscheint in Kooperation mit O'Reilly Media, Inc. unter dem Imprint »O'REILLY«. O'REILLY ist ein Markenzeichen und eine eingetragene Marke von O'Reilly Media, Inc. und wird mit Einwilligung des Eigentümers verwendet.

Hinweis:
Dieses Buch wurde auf PEFC-zertifiziertem Papier aus nachhaltiger Waldwirtschaft gedruckt. Der Umwelt zuliebe verzichten wir zusätzlich auf die Einschweißfolie.

Schreiben Sie uns:
Falls Sie Anregungen, Wünsche und Kommentare haben, lassen Sie es uns wissen: *kommentar@oreilly.de*.

5 4 3 2 1 0

Inhalt

Vorwort

In Zeiten von Big Data, künstlicher Intelligenz und selbstlernenden Systemen ist die Verarbeitung großer Datenmengen ein Thema, dessen Bedeutung in vielen Branchen der Wirtschaft und vielen Forschungsbereichen in der letzten Zeit enorm zugenommen hat. Für diesen Zweck gibt es verschiedene teure kommerzielle Anwendungspakete wie SPSS und SAS, aber auch eine ganze Reihe von Produkten, die von Entwicklergemeinschaften entwickelte und unter freien Lizenzen verbreitet werden. Unter Letzteren hat eine Open-Source-Anwendung namens **R** besondere Popularität erreicht. **R** ist eine spezialisierte Programmiersprache und Arbeitsumgebung, die als (nicht vollständig kompatible) Implementierung einer Sprachdefinition namens *S*[1] entstanden ist und deren Syntax und Funktionsumfang besonders auf die Erfordernisse der numerischen und grafischen Datenanalyse abgestimmt ist. Sie wird als Teil des GNU-Projekts (*https://www.gnu.org/*) entwickelt und dementsprechend unter der liberalen GNU General Public License (GPL) (*https://www.gnu.org/licenses/*) verbreitet.

R – kurz & gut hat das Ziel, Sie schnell und effizient mit **R** vertraut zu machen und Ihnen die prinzipielle Funktionsweise sowie die Möglichkeiten, die diese Programmiersprache bietet, zu vermitteln. Dies ist aber

weder ein Statistiklehrbuch, das Ihnen Techniken und Algorithmen für die Auswertung von Datensätzen erklären will, noch eine umfassende Sprachdokumentation. Für Ersteres gibt es ein breites Angebot an Literatur, mit deren Hilfe Sie das statistische Grundwissen unabhängig von der verwendeten Software erlangen können, und für Letzteres verfügt **R** über umfassende und aktuelle Dokumentationen mit allen Details zu den umfangreichen in **R** enthaltenen Funktionen und den unzähligen Erweiterungspaketen.

So einfach die Sprache **R** auf den ersten Blick erscheint, so komplex ist sie doch, sobald man sich mit den inneren Details auseinandersetzt. Wir haben versucht, die Zusammenhänge zwar etwas vereinfacht, aber auch umfassend genug darstellen, dass man die Sprache sinnvoll anwenden und ihre Features effektiv nutzen kann.

Das Buch richtet sich also in erster Linie an Menschen, die sich zumindest schon mit statistischen Grundbegriffen auskennen, die am besten auch schon Erfahrungen mit der Programmierung in anderen Programmiersprachen gemacht haben – und die möglichst schnell mit **R** produktiv arbeiten möchten.

Der Text basiert auf der bei der Erstellung des Buchs aktuellen **R**-Version 4.2.0 namens »Vigorous Calisthenics«. Da die Sprache mit ihren elementaren Funktionen bereits sehr stabil ist, sollte die Anwendung des hier Dargestellten auf spätere Versionen in der Regel kein Problem sein. Es ist aber durchaus möglich, dass in der Version, die Sie verwenden, bereits neue Funktionen oder zusätzliche Funktionsparameter eingeführt worden sind, die in dieser Auflage des Buchs noch nicht erwähnt sind.

R – kurz & gut ist folgendermaßen gegliedert:

- Kapitel 1, *Arbeiten mit R*, führt Sie in die praktische Arbeit mit **R** ein und zeigt, wie Sie die Anwendung auf Ihrem Rechner installieren, welche Tools zur Verfügung stehen und wie Sie die Arbeitsumgebung nutzen.
- Kapitel 2, *Grundlagen der Sprache R*, zeigt, wie **R** grundsätzlich funktioniert: Wie ist ein Programm aufgebaut, aus welchen Arten von Konstrukten wird es zusammengesetzt, wie werden Daten abgebildet?
- Kapitel 3, *Elementare Datentypen*, stellt Ihnen die Basisdatentypen vor, mit denen Sie es in jedem **R**-Programm zu tun haben, und zeigt, was man in ihnen speichern kann, wie sie erzeugt und wie sie verwendet werden.
- Kapitel 4, *Vektoren und Listen*, beschreibt Vektoren als grundlegende Datenstruktur und erläutert Ihnen, wozu sie dienen, wie sie angelegt werden und was man mit ihnen alles machen kann.
- Kapitel 5, *Variablen*, zeigt, wie Sie in **R** Objekte als Variablen ablegen und wie Umgebungen funktionieren.
- Kapitel 6, *Erweiterte Datenstrukturen*, stellt verschiedene häufig genutzte Datenstrukturen vor, die auf der Basis von Vektoren gebildet werden, und zeigt, wie und wozu sie verwendet werden.
- Kapitel 7, *Funktionen*, erklärt, wie Funktionen definiert und verwendet werden, wie sie als Operatoren genutzt werden können und wie man mit ihrer Hilfe objektorientiert programmiert.
- Kapitel 8, *Programmsteuerung*, behandelt verschiedene Konstrukte und Funktionen, mit denen Sie Verzweigungen und Schleifen programmieren und Fehler behandeln können.

Das Buch ist mit zwei Anhängen zum Nachschlagen versehen.

- Anhang A, *Funktionsübersicht*: Eine gegliederte Übersicht aller Operatoren und der wichtigsten Funktionen, die im **R**-Basispaket mitgeliefert werden.
- Anhang B, *Übersicht der Standardpakete*: Eine Übersicht über die zur Basisinstallation gehörenden Pakete.

Weitere Informationen

Zu **R** stehen sehr umfangreiche Informationen zur Verfügung, die jeden denkbaren Winkel ausleuchten, aber überwiegend englischsprachig und nicht immer sehr übersichtlich sind.

Hilfefunktion

In der Praxis außerordentlich nützlich sind die im Interpreter eingebauten Hilfefunktionen, die direkt aus der Konsole – wie auch aus den entsprechenden Menüanwahlen der Entwicklungsumgebungen – heraus aufgerufen werden können.

Wie Sie die Hilfefunktionen nutzen, erfahren Sie in Kapitel 1, Abschnitt »Interaktive Hilfe« auf Seite 23.

Webseiten

Weitere wichtige Informationsquellen sind natürlich die sehr ausführlichen und detaillierten, aber nicht immer einfach verständlichen Handbücher des GNU-R-Projekts

unter *https://cran.r-project.org/manuals.html* sowie die FAQs unter *https://cran.r-project.org/doc/FAQ/R-FAQ.html*.

Daneben gibt es zahlreiche weitere Adressen, die ergänzende Informationen bieten, darunter:

- *https://www.rdocumentation.org/* ist eine Suchmaschine für alle Dokumentationen der im CRAN und im Bioconductor-Projekt verfügbaren Basis- und Erweiterungspakete.
- Unter *https://rstudio.com/resources/cheatsheets/* finden Sie einige hilfreiche Kurzübersichten, die von den Herstellern der Entwicklungsumgebung *RStudio* zur Verfügung gestellt werden.

Literatur

Im Buchhandel finden Sie zahlreiche englisch- und auch deutschsprachige Bücher, in denen die Arbeit mit **R** mehr oder weniger verständlich beschrieben wird. Die meisten von ihnen legen allerdings – was naheliegt – den Schwerpunkt auf den Aufbau, die Auswertung und die Darstellung statistischer Daten und betrachten die Programmiersprache eher als Hilfsmittel für diesen Zweck.

Auch O'Reilly hat einiges an Literatur zu bieten, darunter:

- *R für Data Science* von Hadley Wickham und Garrett Grolemund, O'Reilly 2017 (*https://oreilly.de/produkt/r-fuer-data-science/*)
- *Statistik mit R* von Joachim Zuckarelli, O'Reilly 2017 (*https://oreilly.de/produkt/statistik-mit-r/*)
- *R in a Nutshell* (deutsch) von Joseph Adler, O'Reilly Verlag 2010 (*https://oreilly.de/produkt/r-in-a-nutshell/*)

Wenn Sie den Innereien von **R** ganz tief auf den Grund gehen wollen, empfiehlt sich das folgende englischsprachige Buch:

- *Advanced R* (2nd Edition) von Hadley Wickham, CRC Press 2019 (auch kostenlos online verfügbar unter *https://adv-r.hadley.nz/*)

Hinweise zur Benutzung des Buchs

Typografische Konventionen

In diesem Buch werden bestimmte Regeln für die Nutzung von Schriftarten angewendet.

Kursiv
: Neue Begriffe, URLs, E-Mail-Adressen sowie Dateinamen und -pfade.

`Konstante Breite`
: Programmlistings sowie innerhalb von Absätzen Verweise auf programmiersprachliche Elemente wie Variablen- und Funktionsnamen, Datentypen, Umgebungsvariablen und Schlüsselwörter.

`Konstante Breite, fett`
: Befehle und sonstige Textteile, die vom Anwender wörtlich eingegeben werden, sowie Hervorhebungen innerhalb von Listings.

`Konstante Breite, kursiv`
: Text, der in einem konkreten Programm durch einen spezifischen Wert zu ersetzen ist.

Hinweise, Tipps und Warnungen

Die folgendermaßen gekennzeichneten Textblöcke sollen auf Einzelheiten aufmerksam machen, die es besonders zu beachten gilt.

Ergänzende Informationen.

Zusätzliche Tipps für die praktische Umsetzung.

Warnung vor möglichen Fehlern.

Codeblöcke im Text

Codeblöcke im Text dienen dazu, die Funktionen der Programmiersprache beispielhaft anhand kurzer Programmsequenzen zu illustrieren. Sie können sie in der Regel leicht nachvollziehen, indem Sie den Text in die **R**-Konsole oder den Editor kopieren. Die im interaktiven Modus gegebenenfalls zu erwartende Programmausgabe wird mit doppelten Kommentarzeichen (##) gekennzeichnet. Dadurch enthalten diese Blöcke immer gültigen **R**-Programmcode. Das folgende Beispiel zeigt in

den ersten beiden Zeilen einen auszuführenden Befehl und in der dritten Zeile das zu erwartende Ergebnis mit vorangestellten doppelten Kommentarzeichen.

```
print (1 +

   1)

## [1] 2
```

Beachten Sie bitte, dass viele Programmbeispiele im Text voraussetzen, dass die im Kapitel vorangehenden Beispiele ausgeführt worden sind.

KAPITEL 1

Arbeiten mit R

Zu **R** gehört nicht nur die formale Sprache, die auf die Ausführung statistischer Operationen spezialisiert ist, sondern dazu gehören auch eine umfangreiche Standardfunktionsbibliothek und eine einfache Arbeitsumgebung, mit deren Hilfe Sie einzelne Befehle eingeben und ausführen sowie programmierte Skripte ablaufen lassen können. In diesem Kapitel wollen wir zunächst diese Arbeitsumgebung kennenlernen, damit wir die später behandelten Features der Sprache und der Bibliothek gleich ausprobieren können.

Es geht auch ohne Installation

Sie müssen **R** nicht unbedingt auf Ihrem eigenen Rechner installieren, um damit arbeiten zu können. Alternativ können Sie auch einen Onlinedienst in Anspruch nehmen und statistische Auswertungen über das Internet betreiben. Diese Möglichkeit bietet sich zum Beispiel dann an, wenn Sie etwas mit der Sprache experimentieren möchten, wenn Sie mit mehreren Rechnern arbeiten, die nicht über ein gemeinsames Netzlaufwerk verfügen, oder wenn Sie beispielsweise ein Tablet oder einen anderen Rechner

haben, auf dem **R** nicht installiert werden kann. Voraussetzung ist natürlich immer, dass Sie bei der Arbeit dauerhaft mit dem Internet verbunden sind.

Es gibt einige Webseiten, die es ermöglichen, Code in **R**-Code online zu erproben, darunter:

- *R Package Documentation* (*https://rdrr.io/snippets/*)
- *Rextester* (*https://rextester.com/l/r_online_compiler*)
- *JDoodle* (*https://www.jdoodle.com/execute-r-online*)
- *myCompiler* (*https://www.mycompiler.io/online-r-compiler*)
- *W2Schools* (*https://www.w3schools.com/r/r_compiler.asp*)

Außerdem bietet der Hersteller von *R-Studio* eine Möglichkeit an, diese sehr komfortable **R**-Entwicklungsumgebung komplett in der Cloud zu verwenden (zum Experimentieren oder für Unterrichtszwecke kostenlos): *https://rstudio.cloud/*.

Beachten Sie die mögliche Datenschutzproblematik, die sich ergibt, wenn Sie personenbezogene oder in anderer Hinsicht sensible Daten über das Internet senden oder auf fremden Systemen speichern.

R installieren

Um lokal arbeiten zu können, brauchen Sie einen halbwegs aktuellen Desktop- oder Laptop-Rechner mit einer neueren Version eines der Betriebssysteme GNU/Linux, Windows oder macOS. Das Gerät muss im Prinzip nicht besonders

performant sein, aber es versteht sich von selbst, dass Sie komplexe Auswertungen mit umfangreichen Datensätzen nur »fahren« können, wenn Ihr Gerät über einen entsprechend leistungsfähigen Prozessor und den nötigen Arbeitsspeicher verfügt. Dabei spielt auch eine Rolle, dass **R** alle zu verarbeitenden Daten im Arbeitsspeicher hält; dieser sollte also ausreichend dimensioniert sein, wenn Sie mit Massendaten arbeiten wollen.

Die Installation ist recht einfach. Öffnen Sie im Browser die generische Download-Seite des *Comprehensive R Archive Network* (CRAN) unter *https://cloud.r-project.org/*. Dort finden Sie die aktuell verfügbaren Softwarestände und -versionen für die drei unterstützten Betriebssysteme sowie verschiedene Quellcodeversionen für den Eigenbau.

Wählen Sie hier Ihr Betriebssystem aus und folgen Sie den Anweisungen, um zum neuesten passenden Installationspaket *R-base* zu gelangen und es auf Ihrem Rechner zu installieren.

Unter GNU/Linux können Sie auch den Paketmanager Ihrer Distribution für die Installation und Aktualisierung von **R** auf Ihrem Rechner verwenden. So brauchen Sie beispielsweise unter Debian und Ubuntu nur diese beiden Zeilen im Konsolenfenster einzugeben:

```
$ sudo apt update

$ sudo apt install r-base
```

Wenn Sie unter Windows einen Paketmanager wie *Npackd* oder *Chocolatey* nutzen, ist die Installation von **R** auch damit möglich.

Die Installation aktualisieren

Die Programmiersprache **R** ist im Prinzip sehr stabil. Um Verbesserungen und Erweiterungen zu implementieren, die insbesondere aufgrund der Entwicklung neuer statistischer Methoden erforderlich werden, gibt es jedoch recht häufig neue Versionen.

Unter GNU/Linux können Sie dafür einfach die gewohnten Mechanismen des Betriebssystems verwenden (hier wieder für Debian und Ubuntu):

```
$ sudo apt upgrade r-base
```

Die GNU/Linux-Paketmanager liefern nicht immer die aktuelle Version von **R**. Wenn Sie eine neuere Version benötigen, gehen Sie am besten zur Website des CRAN-Projekts und folgen den dortigen Anweisungen. Alternativ können Sie auch den Download-Server des CRAN in den Paketmanager einbinden.

Unter Windows verwenden Sie am besten ein **R**-Package namens *installr*, das Ihnen die meiste Arbeit abnimmt. Geben Sie dazu in der **R**-Konsole (siehe unten) die folgenden Befehle ein:

```
install.packages("installr")

library(installr)

updateR()
```

Die R-Konsole starten

Nach der erfolgreichen Installation steht Ihnen die **R**-Arbeitsumgebung in der Form einer Shell-ähnlichen Benutzerschnittstelle namens **R**-Konsole zur Verfügung, deren Aussehen im Detail von Ihrem Betriebssystem abhängt.

Nach einer korrekten Installation können Sie unter allen drei Betriebssystemen die **R**-Konsole im Terminalfenster mit einem Befehl starten, der nur aus dem großgeschriebenen Buchstaben »R« besteht. Daraufhin werden Sie mit einer ausführlichen Willkommensnachricht begrüßt, die so beginnt:

```
$ R

R version  4.2.0 (2022-04-22) -- "Vigorous Calisthenics"

Copyright (C) 2022 The R Foundation for Statistical Computing

. . .

>
```

Dabei erscheint kein gesondertes Fenster, stattdessen läuft die **R**-Konsole direkt im Shell-Fenster.

Windows

Wenn Sie unter Windows die Befehlszeile für die Arbeit mit **R** benutzen wollen, müssen Sie zuvor die Umgebungsvariable `PATH` um den Pfad zu dem Verzeichnis erweitern, in dem die ausführbaren EXE-Dateien von **R** liegen. Abhängig vom System und von der installierten

Version lautet dieser beispielsweise *C:\Programme\R\R-4.2.0\bin* oder ähnlich.

Die grafische Version starten Sie unter Windows aus dem Startfenster heraus als eine Anwendung mit dem Namen *R x64 4.2.0* (entsprechend der von Ihnen installierten Version). Es öffnet sich eine Mehrfensterapplikation namens *RGui* (siehe Abbildung 1-1). Darin meldet sich das Eingangsfenster der **R**-Konsole mit ihrem Willkommensgruß.

RGui verfügt über eine Reihe von Menüoptionen und Buttons, mit denen Grundfunktionen eines einfachen Texteditors ausgelöst werden können und die weitgehend selbsterklärend sind. Zum Teil sind diese Funktionen nur ein Ersatz für bestimmte Befehle, die Sie im Arbeitsfenster selbst eingeben können und denen wir uns im Folgenden nicht eingehender widmen wollen.

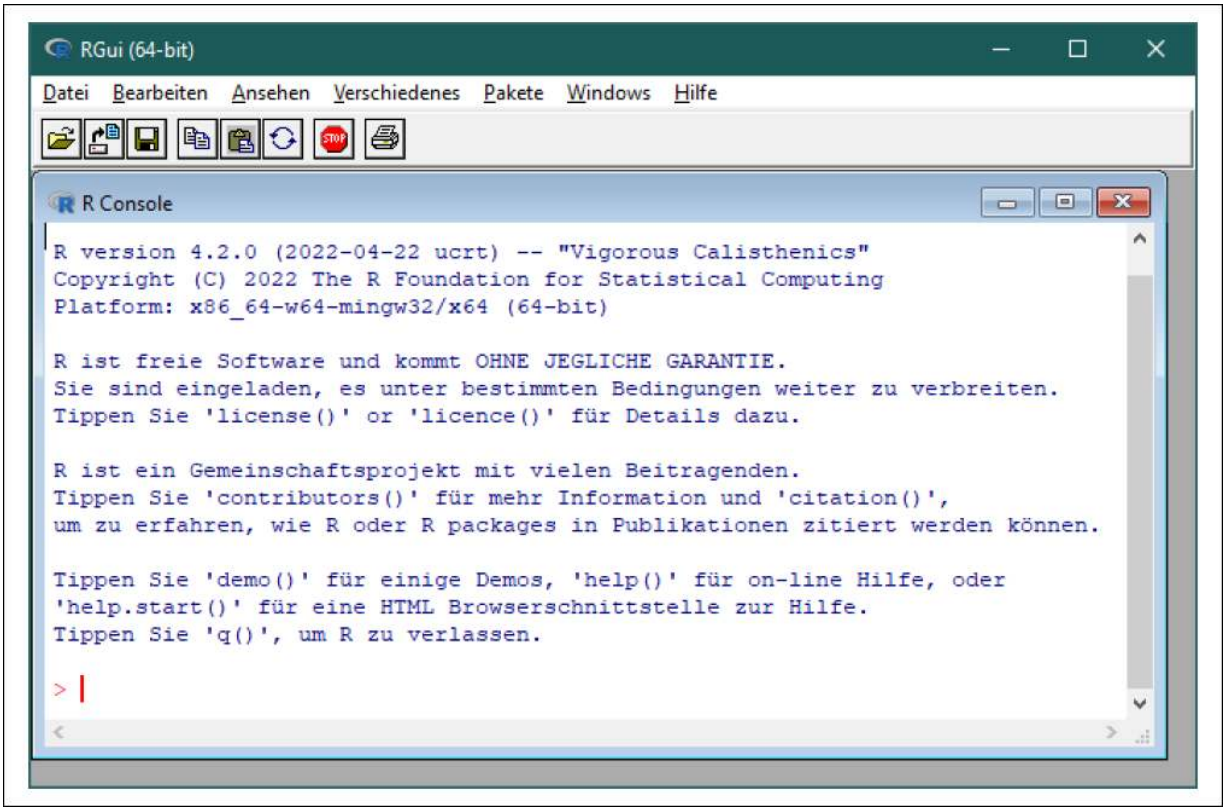

Abbildung 1-1: RGui unter Windows

GNU/Linux

Wenn Sie die Installation erfolgreich ausgeführt haben, geben Sie einfach im Terminalfenster den Befehl `R` ein, um die zeichenorientierte **R**-Konsole zu starten. Wenn Sie Ihr GNU/Linux mit einer grafischen Oberfläche betreiben, steht aber auch hier eine Anwendung mit eigenem Fenster zur Verfügung, die sich dort *tk-R* nennt (siehe Abbildung 1-2). Um diese zu starten, muss der `R`-Befehl mit einem Parameter aufgerufen werden:

```
$ R -g Tk &
```

Der Parameter `-g Tk` sorgt dafür, dass **R** in einer eigenen grafischen Umgebung läuft, die mithilfe des GUI-Werkzeugs *TK* programmiert worden ist, während das `&`-Zeichen am Ende der Zeile notwendig ist, damit ein eigener Prozess gestartet wird und Ihr Terminalfenster nicht blockiert.

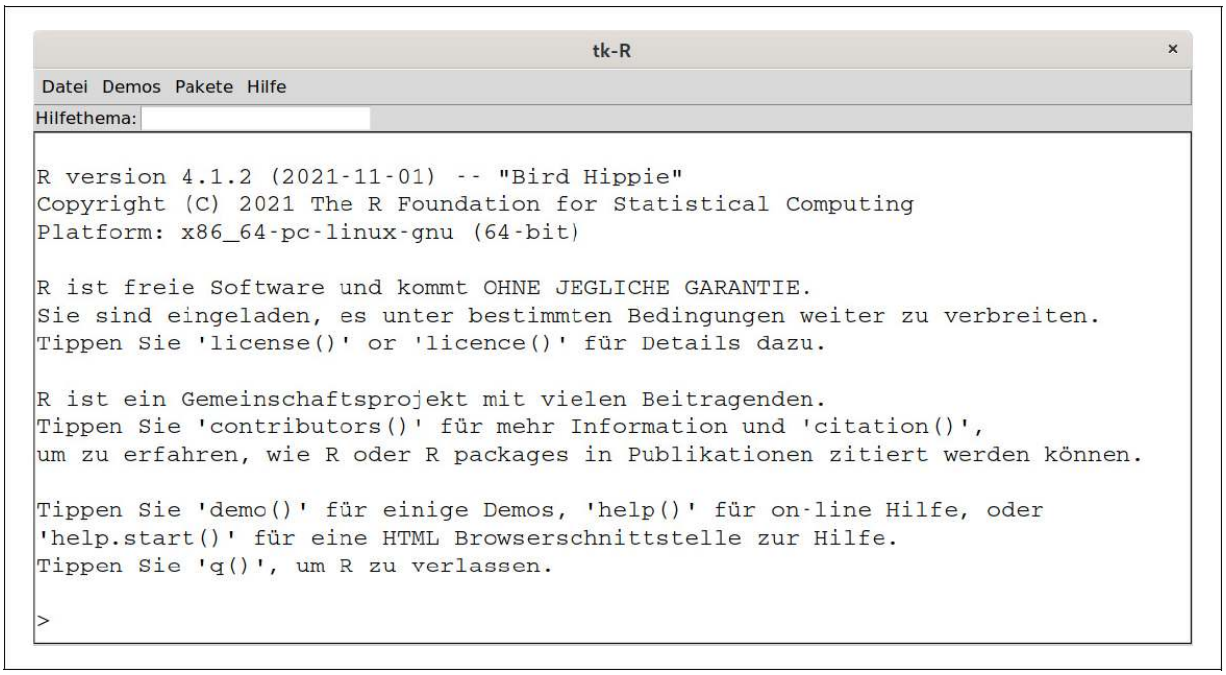

Abbildung 1-2: RGui unter Linux (hier noch in der Version 4.1.2)

tk-R unter Linux enthält etwas weniger Menüoptionen und keine Buttonleiste, verhält sich ansonsten aber ähnlich wie RGui unter Windows.

macOS

Die grafische **R**-Version mit eigenem Fenster starten Sie auf dem Mac mit einem der Programme *R.app* (32-Bit-Version) oder *R64.app* (64-Bit-Version). Das sich daraufhin öffnende Fenster ähnelt weitgehend dem Windows-RGui.

Interaktiv arbeiten

Im Prinzip gibt es zwei Arbeitsmodi für **R**: den interaktiven Modus, in dem Sie jede Anweisung Zeile für Zeile eingeben, und den Skriptmodus, bei dem die auszuführenden Anweisungen in einer Textdatei stehen und automatisch nacheinander ausgeführt werden. Zunächst

sehen wir uns den interaktiven Modus an, der voraussetzt, dass Sie die **R**-Konsole wie oben beschrieben gestartet haben.

Das Arbeitsverzeichnis

Generell ist es empfehlenswert, für jedes Projekt ein eigenes Arbeitsverzeichnis zu verwenden, damit zwischengespeicherte Daten nicht durcheinandergeraten. Daher richten Sie sich, bevor Sie nun mit der Arbeit beginnen, auf Ihrem Rechner ein Verzeichnis für Ihr erstes **R**-Projekt ein, beispielsweise folgendermaßen als Unterverzeichnis zu Ihrem Home-Verzeichnis: *r-projekte/erste-versuche* (GNU/Linux und macOS) oder *r-projekte\erste-versuche* (Windows).

Bevor Sie **R** per Befehlszeile starten, wechseln Sie am besten in das gewünschte Arbeitsverzeichnis, damit das aktuelle Verzeichnis zugleich das Arbeitsverzeichnis ist.

Andernfalls können Sie das Arbeitsverzeichnis auch innerhalb der **R**-Arbeitsumgebung mit der Funktion `setwd("_verzeichnis_")` setzen. Mit `getwd()` rufen Sie das aktuelle Verzeichnis ab.

```
setwd("~/r-projekte/erste-versuche")

getwd()

## [1] "C:/Users/demo/r-projekte/erste-versuche"
```

Beachten Sie: **R** verwendet unter Windows genauso wie unter GNU/Linux in Verzeichnispfaden nicht den Backslash (\), sondern den vorwärts geneigten Schrägstrich (/) und

die Tilde (~) als Kürzel für das Home-Verzeichnis des Benutzers.

Befehle eingeben

Mit **R** arbeiten heißt mit der Tastatur arbeiten. Per Klick und Wischgeste können Sie nichts ausrichten. Vielmehr geben Sie ähnlich wie in der Shell bzw. dem Befehlszeilenfenster der Reihe nach, Zeile für Zeile, hinter dem Promptzeichen (>) Anweisungen ein und senden diese mit der Enter-Taste ab. Wenn ein Befehl ein sichtbares Ergebnis hat, erscheint dieses jeweils in der darauffolgenden Zeile.

Eine Anweisung kann auch ein arithmetischer Ausdruck, ein Funktionsaufruf oder eine Kombination daraus sein. Auf diese Weise können Sie **R** einfach als Taschenrechner nutzen und beispielsweise den Sinuswert einer Zahl ermitteln.

```
sin(pi/4)
```

```
## [1] 0.7071068
```

Die in eckige Klammern gesetzte Zahl `[1]` in der zweiten Zeile, die das Ergebnis der Berechnung ausgibt, besagt, dass dies der erste Wert eines Vektors ist. Ein Vektor ist in **R** eine endliche Folge von Daten desselben Typs (der in diesem Fall aus nur einem Element besteht).

Klarer wird das, wenn Sie beispielsweise mit dem Doppelpunktoperator (siehe Kapitel 4) einen Vektor aus 50 Werten erzeugen und diese ausgeben.

```
1:50
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18
## [19] 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36
## [37] 37 38 39 40 41 42 43 44 45 46 47 48 49 50
```

Hier sehen Sie zu Beginn der zweiten Ergebniszeile das Präfix `[19]`, auf das genau das 19. Element des Ergebnisvektors folgt. Wie viele Elemente in einer Zeile ausgegeben werden, hängt von der Art der Daten und der Breite des Ausgabefensters ab.

Anstelle des `>` erscheint ein Pluszeichen (+) als Eingabeaufforderung, wenn die Anweisung syntaktisch noch nicht komplett ist und weitere Eingaben erforderlich sind.

```
sin(pi            # Eingabeaufforderung: >
   /4)            # Eingabeaufforderung: +
## [1] 0.7071068
```

Die Befehlshistorie

Die **R**-Konsole merkt sich immer Ihre zuletzt eingegebenen Befehle in einer als *Historie* bezeichneten Datenstruktur. Mittels der Pfeiltasten nach oben und unten können Sie sich durch die früheren Befehle bewegen und einen zuvor abgesandten Befehl verändern und erneut aufrufen.

Der Befehl `history()` zeigt die 25 letzten Befehle an (entweder direkt in der Konsole oder in einem Extrafenster). Wenn Ihnen das zu viel oder zu wenig ist, können Sie auch eine andere Anzahl von Zeilen als

Argument angeben. So zeigt folgender Befehl die vier zuletzt eingegebenen Befehle und zusätzlich den `history()`-Befehl selbst an:

```
history(4)

## sin(pi/4)

## 1:50

## sin(pi

## /4)

## history(4)
```

Sofern Sie nicht wissen, wie Sie das Extrafenster mit der Historie wieder verlassen können, versuchen Sie es mit einem Druck auf die Taste *Q*.

Die Befehlshistorie ist außerordentlich hilfreich, wenn Sie nach einer langen interaktiven Sitzung nicht mehr genau wissen, wie Sie einen Datensatz bearbeitet oder wie Sie einen bestimmten Befehl formuliert haben.

Es gibt Funktionen, mit denen Sie die Befehlshistorie in einer Datei speichern oder aus einer Datei einlesen können. Mehr dazu in Anhang A. Referenz in Appendix funktioniert nicht

Verarbeitung abbrechen

Es kann passieren, dass Sie sich bei der Eingabe vertun und daraufhin die Anweisung nicht sinnvoll beenden

können, dafür aber bei jedem Zeilenwechsel wieder den +-Prompt bekommen. Drücken Sie die Tastenkombination *Strg*+*C*, um die Eingabe abzubrechen, ohne dass der Interpreter versucht, sie auszuführen.

Die Tastenkombination *Strg*+*C* können Sie auch dazu verwenden, eine – z. B. infolge einer Endlosschleife im Skript – sehr lange laufende Anweisung abzubrechen. Allerdings ist dies nicht in jedem Fall möglich; wenn *Strg*+*C* nicht funktioniert, müssen Sie den Prozess mit den Mitteln des Betriebssystems beenden (`kill`-Befehl in Linux bzw. unter Windows mit dem Task-Manager).

Interaktive Hilfe

R bietet verschiedene Möglichkeiten, aus der Arbeitsumgebung heraus auf Elemente der Dokumentation zuzugreifen. Hier sind die wichtigsten:

- Die Funktion `help(_thema_)` gibt Informationen zu einem bestimmten Thema aus, z. B. zu einer Funktion. Alternativ können Sie auch einfach `?_thema_` eingeben.
- Die Funktion `help.search("_thema_")` ermöglicht Ihnen, das gesamte Hilfesystem nach einem Schlüsselwort zu durchsuchen. Die Kurzfassung dieser Funktion verwendet zwei Fragezeichen: `??_thema_`.
- Die Funktion `help.start()` fährt einen Webserver mit lokal gespeicherten Hilfeinformationen hoch und greift mit dem Standardbrowser des Rechners darauf zu.

Konsole schließen und Arbeitsbereich speichern