

R ist als freie Software unter der General Public License (GNU) verfügbar und wird für verschiedene Betriebssysteme durch das CRAN (Comprehensive R Archive Network) zum Download bereitgestellt. Die R-Homepage² ist unter <https://www.r-project.org> zu finden, als Teil derer die CRAN-Seite <https://cran.r-project.org> verlinkt ist.

R besteht aus einzelnen Paketen, die für die eigenen Bedürfnisse kombiniert werden können. Pakete sind Dateien, die man gesondert installieren und laden muss. Die Basisversion, die man als Ausgangsprogramm installiert, enthält bereits eine Reihe von Paketen für statistische Auswertungen und graphische Darstellungen. Die verschiedenen Zusatzpakete werden von verschiedenen StatistikerInnen und AnwenderInnen programmiert und allen Personen über die CRAN-Seite zur Verfügung gestellt.

Für die Installation der *R Basisversion* wählt man auf der CRAN-Index-Seite das entsprechende Betriebssystem aus. Da sich die Installationsschritte je nach Betriebssystem unterscheiden, wird im Folgenden auf die Installation unter Linux, Windows und MacOS X eingegangen. Die Installation von zusätzlichen Paketen wird in Abschnitt 2.7 erläutert.

2.1 Empfohlene Installationen und Anpassungen von R

Aufgrund seiner modularen Struktur kann R den individuellen Bedürfnissen sehr gut angepasst werden. Dazu gehören einerseits Pakete, die die Funktionen für statistische Analysen und Graphiken enthalten, aber auch Eingabehilfen wie Zusatzeditoren und

2 CRAN bezeichnet das Server Netzwerk, das in verschiedenen Ländern eine exakte Kopie, sogenannte Mirrors, der Daten speichert. Mit R wird die Programmiersprache und Statistikumgebung bezeichnet.

graphische Benutzeroberflächen. Für das Arbeiten mit diesem Buch, aber auch für die sozialwissenschaftliche Datenanalyse überhaupt, wird an dieser Stelle die folgende Installation empfohlen:

- R Basisversion
- Task View SocialSciences (vgl. 2.7)
- die Pakete `car`, `cocor`, `corpcor`, `ctv`, `Deducer`, `descr`, `foreign`, `gmodels`, `Hmisc`, `installr`, `lsr`, `MASS`, `memisc`, `moments`, `nortest`, `odfWeave`, `plyr`, `PrettyR`, `psych`, `QuantPsyc`, `R2HTML`, `Rcmdr`, `ReporteRs`, `reshape`, `robustbase`, `soc.ca`, `survey`, `weights`, `vcd`.
- RStudio als graphische Benutzeroberfläche (vgl. 2.11.1)

Um aber das Spektrum möglicher Individualisierungen etwas aufzuzeigen, werden in den folgenden Abschnitten zusätzliche und alternative Ergänzungsmöglichkeiten wie Editoren und graphische Benutzeroberflächen vorgestellt.

2.2 R-Installation unter Windows

Die aktuellen R-Versionen setzen Windows XP oder spätere Versionen voraus. Für die Installation, aber auch für das Arbeiten mit R unter *Windows*, sollte man möglichst die Administratorenrechte für das verwendete Laufwerk haben. Die Dynamik von R macht es erforderlich, immer wieder Pakete herunterzuladen und zu installieren (vgl. 2.7). Andernfalls ist R nicht unter */Programme* sondern unter */Eigene Dateien* zu installieren.

Auf der gewählten CRAN-Seite wählt man *Download R for Windows* und dann bei den Subdirectories *base* aus. Die dort aufgeführte R-Version (R-3.X.X for Windows) ist die jeweils gültige und wird auf den PC heruntergeladen. Durch Doppelklicken auf die im Anschluss auf dem Desktop zu findende Datei *R-3.X.X-win.exe* wird die Installation gestartet. Während dieses Prozesses kann der Zielordner für die Programminstallation geändert werden. Ansonsten übernimmt man am einfachsten die voreingestellten Komponenten. Unter *Startoptionen Anpassen* können dann der Anzeigemodus (SDI: in mehreren Fenster, MDI: in einem großen Fenster), der Hilfe-Stil (als Text oder HTML) und Internetzugang eingestellt werden oder die Standardeinstellungen für R übernommen werden. Dann folgen noch Einstellungsmöglichkeiten für den Startmenüordner.

Nach Abschluss der Installation kann R aus dem Windows-Startmenü heraus gestartet werden.

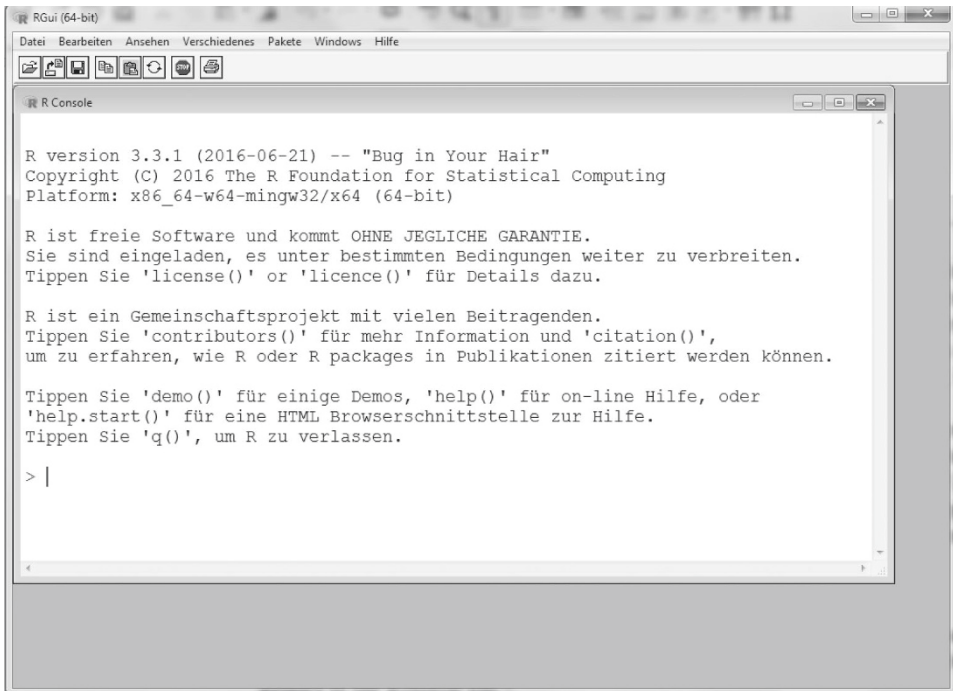


Abbildung 1 Startansicht R unter Windows: Die R Konsole

2.3 R-Installation unter Linux (Ubuntu)

Bei R handelt es sich um eine Open Source Software, die frei zugänglich und beliebig veränderbar ist. Damit steht R ideell den Linux-Systemen nahe, die ebenfalls auf der Idee der Open Source aufbauen, wodurch es den AnwenderInnen ermöglicht wird, das System und die Software an ihre Bedürfnisse anzupassen und zu verändern. Hingegen ist der Source Code von proprietären Systemen und Programmen geschützt und darf nicht verändert werden. Open Source Systeme werden von einer Vielzahl von ProgrammiererInnen, der Linux-Community, gemeinsam entwickelt (vgl. Alleyne 2011). Da nicht alle proprietären Statistikprogramme für Linux-Systeme verfügbar sind, ist R für diese Gruppe von besonderem Interesse. Zudem erscheint die Anwendung von R auf einem Linux-System, bezogen auf die zugrunde liegende Idee, konsequent.

An dieser Stelle wird beispielhaft für Linux-Systeme die R-Installation unter *Ubuntu* 16.04 (Xenial Xerus) erläutert. Weitere Anleitungen für andere Linux-Systeme finden sich auf der CRAN-Homepage. Auf der CRAN-Startseite wählt man *Download R for Linux*, und dann *ubuntu/* und dann die Version aus. Zunächst muss in der Datei

/etc/apt/sources.list ein entsprechender Server eingetragen werden durch eine Eingabe in die Systemkonsole, z. B.

```
deb http://<mein.cran.mirror>/bin/linux/ubuntu xenial/
```

wobei *<mein.cran.mirror>* durch eine geographisch nahe Serveradresse aus der Liste <https://cran.r-project.org/mirrors.html> ersetzt wird. Für die Installation gibt man dann folgendes in die Konsole ein:

```
sudo apt-get update
sudo apt-get install r-base
```

Alternativ zur Installation über die Konsole kann auch über ein Paketverwaltungsprogramm, z. B. Synaptic, nach *r-base-core* gesucht und dieses mit seinen Abhängigkeiten installiert werden.

Um R mit einer Auswahl von empfohlenen Paketen über das System auf dem neuesten Stand zu halten, bedarf es der Installation eines Keys für die CRAN-Archive über die Konsoleneingabe:

```
gpg --keyserver keyserver.ubuntu.com --recv-key E084DAB9
gpg -a --export E084DAB9 | sudo apt-key add -
```

Da sich dieser Key auch ändern kann, ist er auf der entsprechenden Installationsseite von R gegebenenfalls zu prüfen.

Die R-Pakete, die Teil von *r-base* sind, werden in das Verzeichnis */usr/lib/R/library* installiert. Diese können über

```
sudo apt-get update
sudo apt-get upgrade
```

auf den neuesten Stand gebracht werden. Weitere R-Pakete (vgl. 2.7) werden standardmäßig in das Verzeichnis */usr/lib/R/site-library* installiert und direkt in der R-Umgebung geupdatet. Dafür empfiehlt es sich, R jeweils als Superuser *sudo* zu starten. Nach der abgeschlossenen Installation startet R in der Konsole durch die Eingabe von R.

2.4 R-Installation unter Mac OS X

Voraussetzung für die Verwendung von R ist die Version *Mac OS X* Version 10.6 (Jaguar) oder höher. Für die Installation von R folgt man von der CRAN-Homepage dem Link auf *Mac OS X* und installiert von dort die Datei *R-3.X.X.pkg* (latest version) durch einen Doppelklick. Hierfür sind die Administrationsrechte notwendig. Es öffnet sich der Setup-Assistent, der durch die Installation führt.

Die Benutzeroberfläche sieht auf dem Mac manchmal etwas anders aus, grundsätzlich lassen sich aber alle Funktionen an denselben Stellen finden.

2.5 R updaten und deinstallieren

Um unter *Windows* zu einer neueren R-Version zu wechseln, muss diese, analog zur Erstinstallation, heruntergeladen und eingerichtet werden.³ Auch alle nachträglich und zusätzlich installierten Pakete (vgl. 2.7) müssen dann neu heruntergeladen werden bzw. können aus dem library-Ordner der älteren Version (z. B. *C:/Programme/R/R-3.X.X/library*) kopiert, in den entsprechenden Ordner der neuen Version eingefügt und dann unbedingt geupdatet werden (vgl. 2.7). Prinzipiell ist es auch möglich, mehrere R-Versionen parallel auf dem Rechner zu haben, und diese können gleichzeitig verwendet werden.

Unter *Linux* erfolgt das Updaten auf die jeweils neueste R-Version mit den empfohlenen Paketen, sofern man den CRAN-Server der Sourcelist hinzugefügt hat (vgl. 2.3), über die allgemeine Paketverwaltung des Systems.

Die Deinstallation von R erfolgt unter *Windows* über *Start* → *Programme* → *R* → *Uninstall R 3.X.X*. Darüber werden R und alle damit verbundenen Pakete gelöscht. Unter *Linux* kann R entweder über die menügeführte Paketverwaltung, beispielsweise über Synaptic, oder durch ein einfaches Löschen von */usr/local/lib/R* entfernt werden. Unter *Mac OS X* besteht R aus zwei Teilen, der GUI (R.APP) und dem R Framework. Diese finden sich standardmäßig im Ordner */Applications* bzw. */Library/Frameworks*. Um das Programm zu deinstallieren, löscht man am einfachsten die beiden Dateien durch das Ziehen in den Mülleimer.

3 Eine automatisierte Möglichkeit des Updatens unter *Windows* erlaubt das Paket *installr* (zu Paketen vgl. 2.7). Die Funktion `updateR()` prüft dann, ob eine neue R-Version existiert, lädt diese herunter und startet den Installationsprozess. Während der Installation wird man gefragt, ob man die existierende Library kopieren möchte und auf Updates prüfen möchte (vgl. Galili 2016).

2.6 Die R Konsole

Die R Arbeitsumgebung besteht in der Basisversion aus der R Konsole (Abbildung 1). Während die *Windowsversion* von R die Menüs *Datei*, *Bearbeiten*, *Ansehen*, *Verschiedenes*, *Pakete*, *Windows*, *Hilfe* enthält (vgl. Abbildung 1), läuft R auf *Linux* ganz ohne Menüs in der System-Konsole.

Die einfachste Möglichkeit, mit R zu arbeiten, besteht darin, in die Konsole Befehle einzugeben, auf die dann der entsprechende Output folgt. Nach der Eingabeaufforderung bzw. dem Prompt

```
>
```

werden die Befehle geschrieben und mit der Enter-Taste (↵) abgeschlossen. Wenn nichts schief gegangen ist, erscheint daraufhin die Ausgabe, sonst die Fehlermeldung direkt unter der Befehlszeile. Allerdings produziert nicht jede Eingabe in R auch ein Output, beispielsweise wenn Objekte zugeordnet oder Variablen umkodiert werden. Hier muss das Ergebnis der Eingabe separat aufgerufen werden.

Mit der Taste ↑ können vorher eingegebene Befehle zurück geholt und gegebenenfalls bearbeitet werden. Dadurch erspart man sich wiederholtes Tippen.

Mit dem Aufruf

```
> history()
```

öffnet sich ein Fenster, welches die vergangenen Eingaben in der Konsole auflistet. Dieses kann dann auch separat gespeichert werden. Für eine systematische Datenauswertung erscheint jedoch ein Arbeiten mit Skriptdateien über einen Editor (vgl. 2.9) oder eine graphische Benutzeroberfläche (2.11) sinnvoller, als die direkte Befehlseingabe in die Konsole.

2.7 Pakete und Themenbereiche installieren, laden, updaten und entfernen

Die Basisversion von R enthält Pakete für die wichtigsten statistischen Verfahren und Graphiken. Alle übrigen Pakete sind Dateien mit inhaltlich spezialisierten Zusatzfunktionen, die von NutzerInnen geschrieben werden und die man gesondert installieren und laden muss. Auf der CRAN-Website findet sich unter *Packages* eine alphabetische Liste der verfügbaren Pakete. Deren Zahl nimmt kontinuierlich zu und beträgt aktuell 8700.

Für die Erstellung einer Übersichtsliste der auf dem eigenen Rechner installierten Pakete und einer Kurzanzeige über deren Funktionen gibt man den folgenden Befehl ein:

```
> library()
```

Eine genauere Auflistung mit Versionsnamen, Pfadangaben und Abhängigkeiten erhält man über

```
> installed.packages()
```

Eine Übersicht über alle für R verfügbaren Pakete, die allerdings sehr lang ist, kann man sich über

```
> available.packages()
```

anzeigen lassen.

Nicht alle installierten Pakete werden automatisch mit dem Starten von R geladen, da sonst das Programm zu groß und dadurch langsam werden würde. Um diese weiteren, bereits installierten Pakete im Bedarfsfall zu verwenden, müssen diese erst aufgerufen werden über den Befehl

```
> library(paket)
```

Um hingegen zusätzliche Pakete zu installieren, verwendet man den Befehl

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

Die Anführungsstriche `" "` sind nur für die Installation, nicht aber für das Laden von Paketen notwendig. Nach diesem Installationsbefehl muss man in einem sich öffnenden Fenster einen Mirror-Server auswählen. Wenn man keinen anderen Installationspfad spezifiziert (mit dem Argument `, lib`), werden die Pakete standardmäßig in die R-Library gespeichert,⁴ unter *Linux* Ubuntu z.B. `/usr/local/lib/R/site-library/paketname`, unter *Windows* z.B. `/Programme Files/R/R-3.X.X./library/paketname`.

Tipp

Unter einem *Linuxsystem* empfiehlt es sich, Pakete als root zu installieren, wofür man R in der Konsole mit `sudo R` startet. Andernfalls werden neue Paketbibliotheken angelegt, was zu Konflikten zwischen verschiedenen Versionen führen kann.

Um mit einem neu installierten Paket arbeiten zu können, muss dieses ebenfalls mittels `library`-Befehl geladen werden.

⁴ Sofern man nicht als Administrator auf seinem Rechner arbeitet, sollte man für das Updaten R mit Administratorenrechten starten. Andernfalls wird eine zweite Library für die Pakete angelegt.

Auf der CRAN-Website finden sich zusätzlich zu den einzelnen Paketen auch Zusammenstellungen nach inhaltlichen Themenbereichen (vgl. Zeileis/Hornik 2015), wie *Econometrics*, *Official Statistics & Survey Methodology*, *Robust Statistical Methods*, *Psychometric Models and Methods* und viele andere unter CRAN Task Views. Diese Themenbereiche bzw. *Task Views* enthalten für die jeweiligen Gebiete relevante Pakete. Dort findet sich auch ein Themenbereich *Statistics for the Social Sciences*, der von John Fox gepflegt wird und Pakete für Linear and Generalized Linear Models, zur Analyse kategorialer Daten wie verschiedene Verfahren nicht-lineare Regressionen, Pakete zur Analyse von Missing Data, Bootstrapping, Model Selection, Social Network Analysis, Propensity Scores und Matching und weitere enthält.

Um ganze Themenbereiche installieren zu können, bedarf es zunächst des Paketes `ctv`:

```
> install.packages("ctv")
> library(ctv)
```

Den Namen des Themenbereichs übernimmt man von der Seite CRAN Task Views, z. B. den Themenbereich `SocialSciences`. Die nun folgende Installation eines Themenbereiches kann etwas Zeit in Anspruch nehmen, da viele Pakete enthalten sind.

```
> install.views("SocialSciences")
```

Alle Pakete werden fortwährend und unabhängig von der Gesamtversion R überarbeitet, so dass es sinnvoll ist, immer die neuesten Versionen zu verwenden. Um alle Pakete upzudaten, schreibt man

```
> update.packages()
```

Möchte man hingegen nur einzelne Pakete updaten, so gibt man den entsprechenden Namen in Klammern und Anführungsstrichen an.

Eine Liste veralteter Pakete erhält man mit

```
> old.packages()
```

Tipp

Für R und die Pakete sollten regelmäßig Updates durchgeführt werden, um enthaltene Fehler zu korrigieren und Konflikte zwischen verschiedenen Versionen zu vermeiden.

Selbstverständlich kann man Pakete und Themenbereiche auch wieder vom Rechner entfernen mit `remove.packages("paket")`.

Unter *Windows* kann man für die Paketinstallation alternativ Pakete auch auf der R-Website unter dem Menüpunkt *Packages* auswählen, als zip-Datei (sowie die dazu gehörige Dokumentation) herunterladen und dann aus R über das Menü in der R Konsole *Pakete* installieren. In dem angebotenen Selektionsfenster wählt man das entsprechende Verzeichnis und die Paket-Datei aus. Nach erfolgreicher Installation erhält man von R eine entsprechende Meldung:

```
Paket /Paketname/ erfolgreich ausgepackt und MD5 Summen abgeglichen  
aktualisiere HTML Paketbeschreibungen
```

Ähnlich kann man über die Windowsmenüführung der Basisversion von R auch Pakete updaten: *Pakete* → *Aktualisiere Pakete*. Veraltete Pakete im Library-Verzeichnis werden in einem Fenster angezeigt und werden durch OK aktualisiert. Die graphische Benutzeroberfläche RStudio (vgl. 2.11) verfügt ebenfalls über menügestützte Möglichkeiten der Paketverwaltung.

2.8 R beenden, Workspace speichern und Arbeitsverzeichnis wechseln

Um R zu beenden, gibt man

```
> q()
```

ein. Bevor R jedoch geschlossen wird, kommt die Frage, ob das Arbeitsfenster, der sogenannte Workspace gespeichert werden soll:

```
Save workspace image? [y/n/c]:
```

Unter *Windows* erscheint diese Frage als Fenster, unter *Linux* in der R Konsole. Nach Eingabe bzw. Anklicken des gewählten Buchstaben wird R geschlossen.

Tipp

Es empfiehlt sich, den *Workspace nicht zu speichern*, da das Vorhandensein alter Objekte (vgl. 3.4) verwirrend sein kann! Auf jeden Fall sollte der Workspace nie im voreinstellten Verzeichnis abgelegt werden. Statt dessen sollten die eingegebenen Funktionen als Skript über den Editor (vgl. 2.9) separat gespeichert werden. Bei Verwendung der graphischen Benutzeroberfläche RStudio (vgl. 2.11) besteht über das Menü *Tools* → *Global Options* → *General* die Möglichkeit, die Nachfrage, ob der Workspace beim Beenden gespeichert werden soll, abzuschalten. Dann wird der Workspace standardmäßig nicht gespeichert.

Im sogenannten Workspace, dem Arbeitsfenster, werden alle Objekte (vgl. 3.4) gespeichert, die in einer Session erstellt wurden. Dieser Workspace wird beim nächsten Starten wieder geladen. Um zu sehen, in welches Arbeitsverzeichnis der Workspace gespeichert wird, verwendet man:

```
> getwd()
```

und erhält als Antwort den entsprechenden Dateipfad, z. B.

```
[1] "C:/verzeichnis/"
```

Wenn man einen Workspace speichern möchte, sollte man das Verzeichnis wechseln:

```
> setwd("C:/meine.dateien/")
```

Unter *Windows* in der Basisversion kann das Arbeitsverzeichnis auch über das Menü *Datei* → *Verzeichnis wechseln* sowie über die graphische Benutzeroberfläche RStudio (vgl. 2.11) über das Menü *Session* → *Save Working Directory* festgelegt werden. Das entsprechende Verzeichnis muss bereits angelegt sein und es sollten unbedingt Sonderzeichen und Umlaute vermieden werden. Um nun den aktuellen Workspace zu speichern gibt man ein:

```
> save.image("C:/meine.dateien/mein.workspace.RData")
```

Im Anschluss verlässt man R, ohne den Workspace nochmals zu speichern. Der gesondert gespeicherte Workspace kann wieder geladen werden durch:

```
> load("C:/meine.dateien/mein.workspace.RData")
```

Unter *Windows* geht dies in der Basisversion ebenfalls über *Datei* → *Lade Workspace*, mit der graphischen Benutzeroberfläche RStudio (vgl. 2.11) über *Session* → *Load Workspace*.

2.9 Editoren: Wordpad, Tinn-R, Rgedit

In der *Windows* Version von R enthält die Basisversion einen einfachen Editor, in dem sich ein sogenanntes Skript schreiben lässt, das einzig die Befehle, und nicht die Ausgaben enthält. Die Befehle lassen sich über Ctrl+R direkt aus dem Skript ausführen. Die Trennung von Eingabe und Ausgabe hat den Vorteil – ähnlich wie in SPSS oder Stata die Syntaxfunktion –, dass die Befehle und Funktionen übersichtlich organisiert und jeder Zeit wieder ausgeführt werden können.

Ein neues Skript erstellt man über die Menüs in der R Konsole, die dafür im Vollbildmodus angezeigt werden muss: *Datei* → *Neues Skript*. Der Editor öffnet sich in einem neuen Fenster. Alternativ kann auch *Wordpad* als Editor verwendet werden, dann müssen die Befehle allerdings manuell in die R Konsole kopiert werden. Mit *Tinn-R* (<http://www.sciviews.org/Tinn-R/>) steht zudem ein spezieller, auf R zugeschnittener Texteditor zur Verfügung, der die Befehlseingabe unterstützt.

Unter *Linux* ist kein separater Editor in R eingebaut. Unter *Ubuntu* kann man jedoch den Standard-Editor *gedit* verwenden, der bereits eine Funktion enthält, um die R-Syntax hervorzuheben. Zusätzlich kann man mit dem Plugin *Rgedit* (<http://rgedit.sourceforge.net/>) spezielle R-Funktionalitäten hinzufügen, beispielsweise wird ein Menüpunkt *R* hinzugefügt, über den der Befehl direkt an die Konsole geschickt wird. Allerdings wird *Rgedit* offenbar nicht mehr weiterentwickelt.

2.10 Verwenden externer Dateien für Befehle und Output

Beim Arbeiten mit Datensätzen kann es hilfreich sein, mit mehreren Skriptdateien zu arbeiten. Beispielsweise kann ein Skript nur für Neu- und Umkodierungen von Variablen verwendet werden. Befehle, die in anderen Dateien abgespeichert sind, können jederzeit direkt aufgerufen und ausgeführt werden. Liegt eine solche Datei im Arbeitsverzeichnis, gibt man in die Konsole oder das aktuelle Skript

```
> source("meine.datei")
```

ein. Liegt die Datei in einem anderen als dem Arbeitsverzeichnis von R, so muss der gesamte Pfadname angegeben werden. Die *R-Windows* Version enthält den `source`-Befehl außerdem auch im Menü. Auf diese Weise können bestehende Skripte in neue Skripte eingebettet werden. Diese eingebettete Datei wird wie ein einfacher Befehl in ihrer Gesamtheit ausgeführt, ohne dass das Skript separat aufgerufen und ausgeführt werden müsste.

Möglich ist auch, den Output statt in der Konsole anzuzeigen direkt in eine externe Textdatei im Arbeitsverzeichnis umzuleiten. Hierfür verwendet man den `sink`-Befehl.

```
> sink("meine.output.datei")
```

Die Ergebnisse aller folgenden Operationen finden sich dann in dieser Datei. Möchte man den Output jedoch wieder direkt in der Konsole angezeigt bekommen, gibt man ein

```
> sink()
```

Auf die Möglichkeiten, Objekte wie Tabellen und Graphiken in andere Dateiformate umzuleiten, wird in Kapitel 8 ausführlich eingegangen.

2.11 Graphische Benutzeroberflächen

Graphische Benutzeroberflächen (GUI) und integrierte Entwicklungsumgebungen (IDE) erleichtern das Arbeiten mit R durch Autovervollständigkeitsfunktionen, automatische Einrückungen, Syntaxhervorhebung, integrierte Hilfsfunktion, Informationen zu Objekten im Workspace, menügestützten Oberflächen und Daten-Viewer. Inzwischen existiert eine Vielzahl von graphischen Benutzeroberflächen. Zu den populärsten gehören derzeit RStudio und R Commander.⁵ Beide sind mit *Linux*, *Windows* und *Mac OS X* kompatibel.

Darüber hinaus sind weitere graphische Benutzeroberflächen für verschiedene Zielgruppen und Systeme verfügbar, sowie Schnittstellen zu anderen Programmen, u. a. zu SPSS und mit RExcel, eine graphisch gestützte Integrationsmöglichkeit von R in MS Excel (vgl. Heiberger/Neuwirth 2009).

2.11.1 RStudio

RStudio ist als Desktop Version (mit einer Open Source oder einer kommerziellen Lizenz) und als Server Version erhältlich, die über den Webbrowser zugänglich ist. Zudem läuft RStudio auf allen drei Betriebssystemen.

Für die Installation unter *Windows* lädt man von der RStudio-Website (<https://www.rstudio.com/products/rstudio/download/>) die aktuelle Version auf den Desktop und führt anschließend diese RStudio-X.XX.XXX.exe Datei aus. RStudio verwendet jeweils die aktuelle R-Version, beide müssen separat geupdatet werden.

Für die Installation unter *Linux Ubuntu* kann man ebenfalls von der RStudio-Website die entsprechende Installationsdatei, diesmal als .zip-Datei, herunterladen. Oder aber man installiert das Programm über die Konsole:

```
sudo apt-get update
wget "http://download1.rstudio.org/rstudio-0.99.903-amd64.deb"
sudo dpkg -install rstudio-099.891.deb
```

5 In der ersten Auflage dieses Buches wurde mit JGR gearbeitet. Seither hat sich jedoch RStudio als komfortable Benutzeroberfläche durchgesetzt.

Für andere RStudio-Versionen muss die entsprechende Versionsnummer angepasst werden.

Voraussetzung ist die erfolgte Installation von R (vgl. 2.3). Dann wird das entsprechende Repository zur Source.list hinzugefügt:

```
sudo gedit /etc/apt/source.list
```

Am Ende der Datei wird, angepasst an die installierte Version, folgende Zeile eingefügt und dann gespeichert:

```
deb http://cran.rstudio.com/bin/linux/ubuntu xenial/
```

In der Standardansicht besteht RStudio aus vier Fenstern (Abbildung 2): Oben links findet man das Skript-Fenster, in das die Befehle eingegeben werden. Ausgeführt

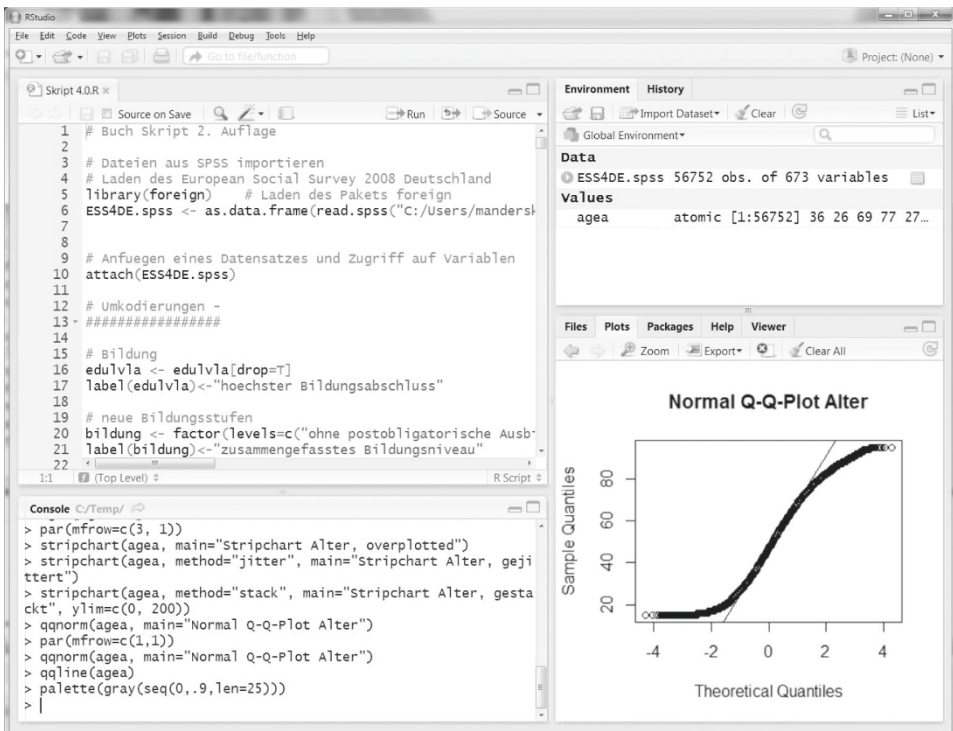


Abbildung 2 Die vier Fenster von RStudio

wird die jeweilige Zeile erst, wenn man `Ctrl+Enter` drückt. Darunter befindet sich das Ausgabefenster, in dem die Ergebnisse der Befehlsausführungen angezeigt werden. Rechts oben wird der Workspace, bezeichnet als Global Environment, angezeigt,

und man kann mit den Registerkarten zur History wechseln, eine Aufzeichnung aller ausgeführten Befehle. Rechts unten werden Graphiken, Hilfsseiten, die Verzeichnisstruktur oder die Paketverwaltung angezeigt. Zusätzlich sind eine Reihe von Menüs in die Oberfläche eingebaut, z. B. zum Speichern des Skripts, Ausführen von Befehlszeilen oder Leeren des Workspace. Die Ansicht kann über das Menü *View* geändert und den eigenen Bedürfnissen angepasst werden. Unter *Tools* → *Global Options* (Abbildung 3) finden sich eine Reihe weiterer Einstellungsoptionen.

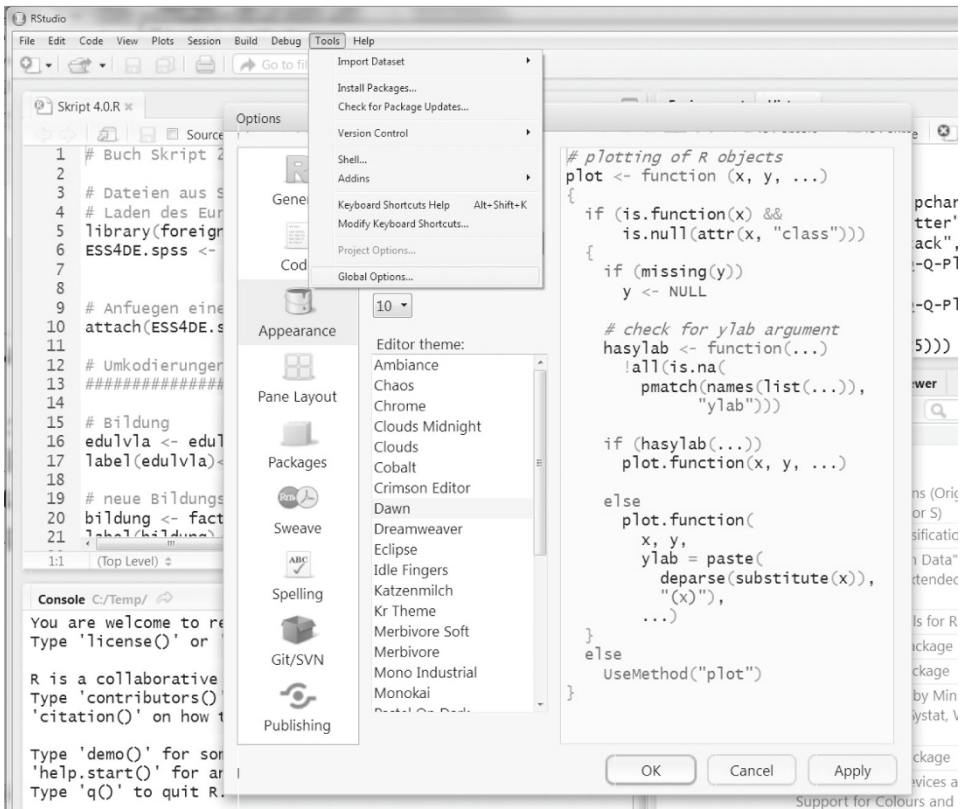


Abbildung 3 RStudio Global Options

Hinweis

Im Folgenden wird immer wieder auf Funktionen der graphischen Benutzeroberfläche *RStudio* verwiesen. Bei *RStudio* erfolgt die Befehlseingabe über ein *Skript* bzw. die Konsole, und nicht über Menüs, wodurch alle Pakete (und nicht nur, wie im *R Commander* (2.11.2) die integrierten Pakete) verwendet werden können.

2.11.2 R Commander

In vielen Einführungen zu R wird die graphische Benutzeroberfläche R Commander empfohlen, da sie den Einstieg in dieses Statistikprogramm erleichtert (vgl. Reisinger/Wagner 2015). Der R Commander (Fox 2005; vgl. Abbildung 4) ermöglicht eine rein menügesteuerte Datenauswertung ohne manuelle Befehlseingabe, ähnlich wie in kommerziellen Programmpaketen wie SPSS.

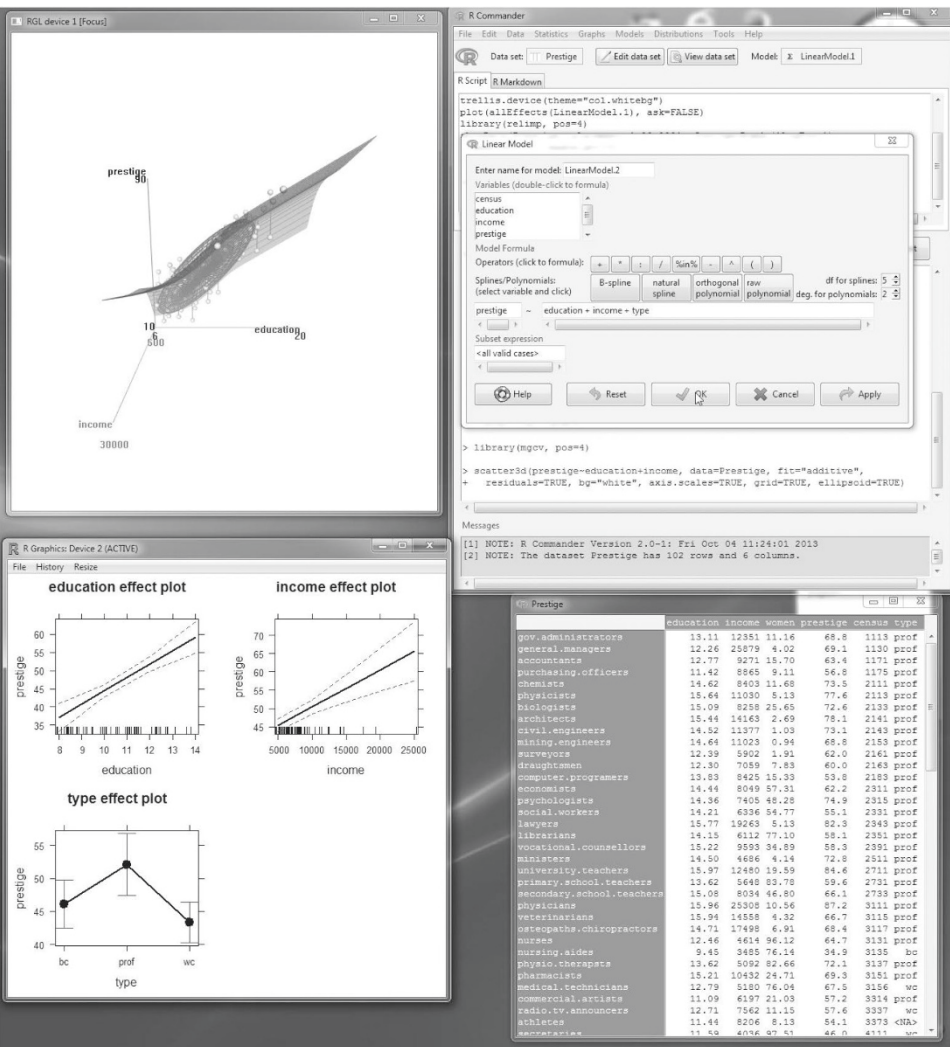


Abbildung 4 Screenshot R Commander (Fox 2015)

Zwar erscheint das Arbeiten mit R zunächst sehr viel einfacher, als die Verwendung eines Editors und die Eingabe von Befehlen, die Anwendung von R bleibt dabei jedoch auf die Menüs und deren zugrunde liegenden Funktionen beschränkt. Inzwischen liegen knapp 40 PlugIns vor, die aus dem R Commander die Funktionen von anderen Paketen zugänglich machen. Damit bleibt trotzdem nur ein Teil der Möglichkeiten von R verfügbar.

Installiert und daran anschließend geladen werden kann der R Commander wie jedes andere Paket über die Befehlszeile. Gegebenenfalls wird man auf die Notwendigkeit eines Nachinstallierens weiterer Pakete hingewiesen.

```
> install.packages("Rcmdr")  
> library(Rcmdr)
```

Ausführlicher wird auf der R Commander von Reisinger/Wagner (2015), Luhmann (2010: 14–17), Faes (2010: 132–139) und Fox (2005) behandelt.

2.12 Übersicht über die eingeführten R-Befehle

Befehl	Funktion
<code>history()</code>	Aufrufen der eingegebenen Befehle
<code>library()</code>	Auflisten und Kurzbeschreibung aller installierten Pakete
<code>installed.packages()</code>	Anzeige aller installierten Pakete mit Versionsnummer, Abhängigkeiten und Pfad
<code>available.packages()</code>	Übersicht über alle verfügbaren Pakete
<code>library(paket)</code>	Aufrufen eines bereits installierten Pakets, das nicht automatisch geladen wird
<code>install.packages("paket")</code>	Installieren eines Pakets von der CRAN-Seite
<code>old.packages()</code>	Anzeige aller Pakete, für die eine neuere Version verfügbar ist
<code>update.packages()</code>	Updaten aller installierten Pakete

Befehl	Funktion
<code>remove.packages ("paket")</code>	Deinstallieren eines Pakets
<code>q()</code>	Beenden von R
<code>getwd()</code>	Angabe des aktuellen Arbeitsverzeichnisses
<code>setwd("dateipfad")</code>	Festlegen des Ortes des aktuellen Arbeitsverzeichnisses
<code>save.image("dateipfad/ datei.RData")</code>	Speichern des aktuellen Arbeitsverzeichnisses
<code>load("dateipfad/datei. RData")</code>	Aufrufen eines früher gespeicherten Arbeitsverzeichnisses
<code>source("datei")</code>	Aufrufen einer im Arbeitsverzeichnis gespeicherten Skriptdatei
<code>sink("ausgabe")</code>	Ausgabe in eine externe Datei schicken
<code>sink()</code>	Ausgabe wieder in Konsole schicken
Paket <code>ctv</code>	
<code>install.views("view")</code>	Installieren eines Themenbereichs von der CRAN-Seite

Sozialwissenschaftliche Datenanalyse mit R

Eine Einführung

Manderscheid, K.

2017, XX, 278 S. 45 Abb., Softcover

ISBN: 978-3-658-15901-6