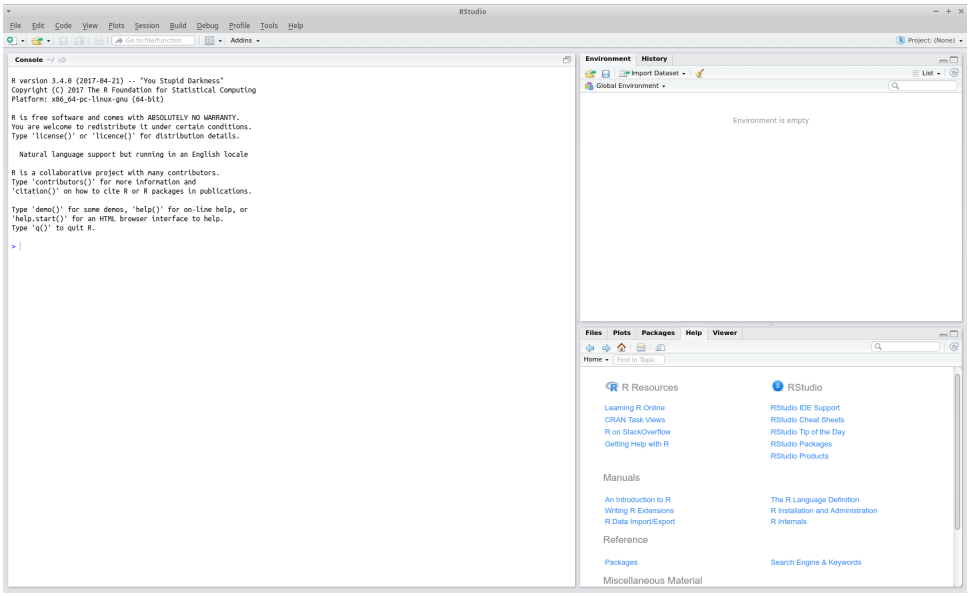
Erste Schritte in R

Karsten Lübke

Hinweise

R ist der Name eines Programms für Statistik und Datenanalyse, RStudio ist eine komfortable Entwicklungsumgebung für R.

Nach dem Start von RStudio erscheint folgender Bildschirm, wobei die Version neuer sein kann.



Links, in der *Console* werden die Befehle eingegeben. Rechts oben können Sie z. B. die Daten, aber auch andere Objekte, mit denen Sie arbeiten, betrachten, auch die Historie der Befehle wird dort angezeigt. Rechts unten können Sie u. a. Dateien und Abbildungen auswählen, aber auch Hilfeseiten und Tipps betrachten.

Wir werden zunächst in der Konsole arbeiten.

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 von R dann ignoriert.
- R wendet Befehle direkt an
- R ist objektorientiert, d. h. dieselbe Funktion hat evtl. je nach Funktionsargument unterschiedliche Rückgabewerte
- Hilfe zu einem Befehl erhält man über ein vorgestelltes Fragezeichen `?`
- Zusätzliche Funktionalität kann über Zusatzpakete hinzugeladen werden. Diese müssen ggf. zunächst installiert werden
- Mit der Pfeiltaste nach oben können Sie einen vorherigen Befehl wieder aufrufen
- Sofern Sie das Skriptfenster verwenden: einzelne Befehle aus dem Skriptfenster in R Studio können Sie auch mit `Str` und `Enter` an die Console schicken

R als Taschenrechner

Auch wenn Statistik nicht Mathe ist, so kann man mit R auch rechnen. Geben Sie zum Üben die Befehle in der R Konsole hinter der Eingabeaufforderung `>` ein und beenden Sie die Eingabe mit `Return` bzw. `Enter`.

```
4+2
```

```
## [1] 6
```

Das Ergebnis wird direkt angezeigt. Bei

```
x <- 4+2
```

erscheint zunächst kein Ergebnis. Über `<-` wird der Variable `x` der Wert `4+2` zugewiesen. Wenn Sie jetzt

```
x
```

eingeben, wird das Ergebnis

```
## [1] 6
```

angezeigt. Sie können jetzt auch mit `x` weiterrechnen.

```
x/4
```

```
## [1] 1.5
```

Vielleicht fragen Sie sich was die `[1]` vor dem Ergebnis bedeutet. R arbeitet vektororientiert, und die `[1]` zeigt an, dass es sich um das erste (und hier auch letzte) Element des Vektors handelt.

R zur Datenanalyse

Wir wollen R aber als Tool zur Datenanalyse verwenden. Daher müssen wir zunächst Daten einlesen.

Zunächst laden wir die Daten als `csv` Datei herunter

```
download.file("https://goo.gl/whKjnl", destfile = "tips.csv")
```

Der Inhalt der Datei ist jetzt als Tabelle `tips` in R verfügbar.

Hier können Sie mehr über die Daten erfahren.

Das Einlesen von `csv` Dateien aus dem Arbeitsverzeichnis in R kann erfolgen über

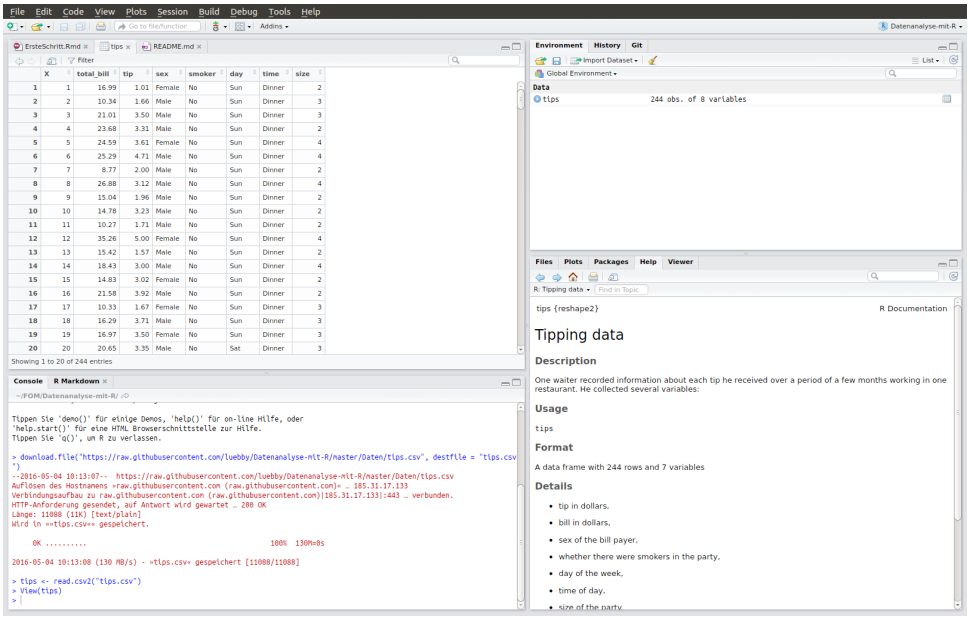
```
tips <- read.csv2("tips.csv")
```

Wo das lokale Verzeichnis (“working directory”) ist, können Sie über

```
getwd()
```

erfahren.

Der Datensatz `tips` taucht jetzt im `Enviroment` Fenster rechts oben in RStudio auf. Durch Klicken auf den Namen können Sie diese betrachten.



Alternativ können Sie Daten in RStudio komfortabel mit dem Button `Import Dataset` (im Fenster `Environment` oder über das Menü `File`) öffnen.

Erste Analyse des tips Datensatzes

Dieser Datensatz aus

Bryant, P. G. and Smith, M (1995) Practical Data Analysis: Case Studies in Business Statistics. Homewood, IL: Richard D. Irwin Publishing

enthält Trinkgelddaten. Diese sind in tabellarischer Form dargestellt, d. h. üblicherweise, dass die Beobachtungen zeilenweise untereinander stehen, die einzelnen Variablen spaltenweise nebeneinander. In R heißen solche Daten *data frame*. Um einen ersten Überblick über die verschiedenen Variablen zu erhalten geben wir den Befehl `str()` ein:

```
str(tips)
```

```
## 'data.frame':    244 obs. of  7 variables:
##  $ total_bill: num  17 10.3 21 23.7 24.6 ...
##  $ tip       : num  1.01 1.66 3.5 3.31 3.61 4.71 2 3.12 1.96 3.23 ...
##  $ sex       : Factor w/ 2 levels "Female","Male": 1 2 2 2 1 2 2 2 2 2 ...
##  $ smoker    : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ day       : Factor w/ 4 levels "Fri","Sat","Sun",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ time      : Factor w/ 2 levels "Dinner","Lunch": 1 1 1 1 1 1 1 1 1 1 ...
##  $ size      : int  2 3 3 2 4 4 2 4 2 2 ...
```

Dieser enthält also 244 Zeilen (Beobachtungen) und 7 Spalten (Variablen). Alternativ kann man diese Information auch über

```
dim(tips)
```

```
## [1] 244   7
```

erhalten.

Numerische (metrische) Variablen sind in R in der Regel vom Typ `numeric` (stetig) oder `int` (Ganze Zahlen), kategorielle (nominale, ordinale) Variablen vom Typ `factor` (bei ordinal: Option `ordered = TRUE`) oder `character`. `str()` und `dim()` sind erste Befehle, d. h., Funktionen in R, denen in der Klammer das jeweilige Funktionsargument übergeben wird.

```
head(tips) # Obere Zeilen
tail(tips) # Untere Zeilen
```

Ermöglichen ebenfalls einen Einblick über die Daten. Der Befehl

```
names(tips)
```

gibt die Variablennamen zurück. Mit Hilfe des `$` Operators kann auf einzelne Variablen eines Dataframes zugegriffen werden:

```
tips$sex
```

erhalten Sie bspw. das Geschlecht des Rechnungszahlers.

Übung: Lassen Sie sich die Variable Rechnungshöhe (`total_bill`) anzeigen.

---

mosaic

`mosaic` ist ein Zusatzpaket, welches die Analyse mit R erleichtert. Sofern noch nicht geschehen, muss es *einmalig* über

```
install.packages("mosaic")
```

installiert werden.

Um es verwenden zu können, muss es - wie jedes Paket - für *jede* neue R-Sitzung über `library(mosaic)`geladen werden. Die angegebenen Hinweise sind keine Fehlermeldung!

```
library(mosaic)
```

```
## Loading required package: dplyr
##
## Attaching package: 'dplyr'
## The following objects are masked from 'package:stats':
##
##     filter, lag
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
## Loading required package: lattice
## Loading required package: ggplot2
## Loading required package: mosaicData
## Loading required package: Matrix
##
## The 'mosaic' package masks several functions from core packages in order to add additional featur
## The original behavior of these functions should not be affected by this.
##
## Attaching package: 'mosaic'
## The following object is masked from 'package:Matrix':
##
##     mean
## The following objects are masked from 'package:dplyr':
##
##     count, do, tally
```

```
## The following objects are masked from 'package:stats':
##
##     binom.test, cor, cov, D, fivenum, IQR, median, prop.test,
##     quantile, sd, t.test, var

## The following objects are masked from 'package:base':
##
##     max, mean, min, prod, range, sample, sum
```

Sollte beim Laden eines Paketes eine Meldung wie

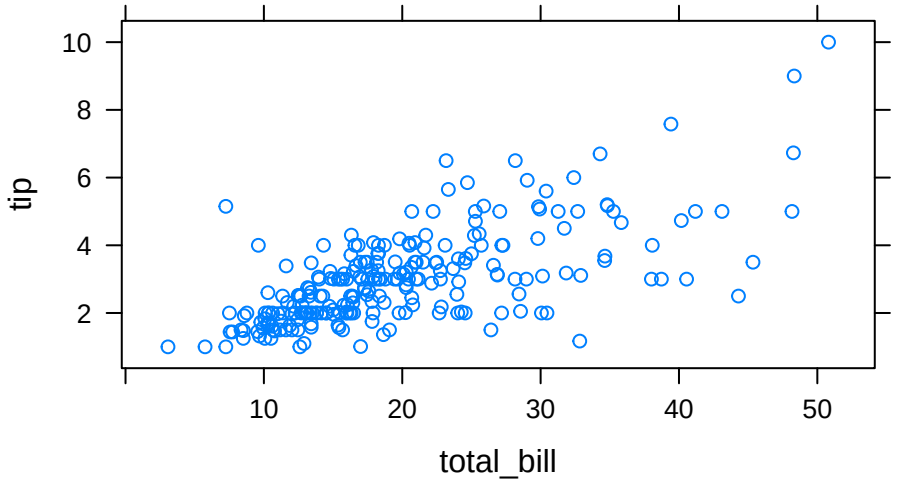
```
library(xyz)
```

```
## Error in library(xyz): there is no package called 'xyz'
```

erscheinen, muss das Paket `xyz` zunächst installiert werden (siehe oben).

Der Grundgedanke von `mosaic` ist *Modellierung*. In R und insbesondere in mosaic wir dafür die Tilde `~` verwendet. `y~x` kann dabei gelesen werden wie "y ist eine Funktion von x". Beispielsweise um eine Abbildung (Scatterplot) des Trinkgeldes `tip` (auf der Y-Achse) und Rechnungshöhe `total_bill` (auf der X-Achse) zu erhalten, kann man in R folgenden Befehl eingeben:

```
xyplot(tip ~ total_bill, data=tips)
```



Das Argument `data=tips` stellt klar, aus welchen Datensatz die Variablen kommen. Die Abbildung ist im RStudio jetzt rechts unten im Reiter *Plots* zu sehen.

Übung: Wie würden Sie den Trend beschreiben?

Wie oben erwähnt können wir R auch gut als Taschenrechner benutzen, sollten aber bedenken, dass R vektorweise arbeitet. D. h.

```
tips$tip/tips$total_bill
```

gibt für *jede Beobachtung* die relative Trinkgeldhöhe bezogen auf die Rechnungshöhe an. Über

```
(tips$tip/tips$total_bill)<0.10
```

erhalten wir einen Vektor vom Typ `logical`. Dieser nimmt nur zwei Werte an, nämlich `TRUE` und `FALSE`, je nach dem ob der jeweilige Wert kleiner als 0.10 ist oder nicht. Neben `<` und `>` bzw. `<=` und `>=` gibt es ja auch noch die Prüfung auf Gleichheit. Hierfür werden in R gleich *zwei* Gleichheitszeichen verwendet, also `==`.

Übung: Was gibt folgender der Befehl zurück?

```
tips$sex=="Female"
```

Logische Vektoren können z.B. mit "und" `&` oder "oder" `|` verknüpft werden:

```
tips$sex=="Female" & tips$smoker=="Yes"
```

gibt die Tischgesellschaften als `TRUE` wieder, in denen die Rechnung von Frauen beglichen wurde *und* geraucht wurde,

```
tips$sex=="Female" | tips$smoker=="Yes"
```

gibt die Tischgesellschaften als `TRUE` wieder, in denen die Rechnung von Frauen beglichen wurde *oder* geraucht wurde.

Intern wird `TRUE` in R mit der Zahl 1 hinterlegt, `FALSE` mit 0. Mit dem Befehl `sum()` kann man daher die Elemente eines Vektor aufsummieren, also erfahren wir über

```
sum(tips$sex=="Female" & tips$smoker=="Yes")
```

dass bei 33 Tischgesellschaften bei denen geraucht wurde, eine Frau die Rechnung bezahlte. Im Verhältnis zu allen Tischgesellschaften, bei denen eine Frau zahlte, liegt der Raucheranteil also bei 0.3793103:

```
sum(tips$sex=="Female" & tips$smoker=="Yes") / sum(tips$sex=="Female")
```

Übung: Wurde bei den Tischgesellschaften, bei denen ein Mann zahlte, relativ häufiger geraucht als bei den Frauen?

Übung: Teaching Rating

Dieser Datensatz analysiert u. a. den Zusammenhang zwischen Schönheit und Evaluierungsergebnis:

Hamermesh, D.S., and Parker, A. (2005). Beauty in the Classroom: Instructors' Pulchritude and Putative Pedagogical Productivity. Economics of Education Review, 24, 369–376.

Sie können ihn von https://goo.gl/6Y3KoK herunterladen. Hier gibt es eine Beschreibung.

1. Lesen Sie den Datensatz in R ein.
2. Wie viele Zeilen, wie viele Spalten liegen vor?
3. Wie heißen die Variablen?
4. Betrachten Sie visuell den Zusammenhang von dem Evaluierungsergebnis `eval` und Schönheit `beauty`. Was können Sie erkennen?
5. Sind relativ mehr Frauen oder mehr Männer (`gender`) in einem unbefristeten Arbeitsverhältnis (*Tenure Track*, `tenure`)?

Daten importieren

Der Datenimport in R ist in vielen unterschiedlichen Dateiformaten möglich. Das `csv` Format eignet sich besonders zum Übertragen von Datendateien. Im deutschsprachigen Raum wird dabei als *Dezimaltrennzeichen* das Komma `,` und als *Datentrennzeichen* das Semikolon `;` verwendet. In der ersten Zeile sollten die Variablennamen stehen. Das Einlesen in einen R Data-Frame (hier `meineDaten`) kann dann über

```
meineDaten <- read.csv2(file.choose()) # Datei auswählen
```

erfolgen.

Der Befehl `file.choose()` öffnet dabei den Dateiordner. Bei "internationalen" `csv` Dateien ist das Datentrennzeichen i. d. R. ein Komma `,`, das Dezimaltrennzeichen ein Punkt `.`. Hier funktioniert der Import in R dann über den Befehl `read.csv`.

In R Studio gibt es im Reiter `Environment` im Fenster Rechts oben einen Menüpunkt `Import Dataset` der mehr Einstellmöglichkeiten bietet.

Excel Dateien können unter anderem mit Hilfe des Zusatzpaketes `readxl` können eingelesen werden:

```
library(readxl) # Paket laden
meineDaten <- read_excel(file.choose())  # Datei auswählen und in R einlesen
```

Hier finden Sie eine Linksammlung zu verschiedenen Datenquellen.

Literatur

- Nicholas J. Horton, Randall Pruim, Daniel T. Kaplan (2015): Project MOSAIC Little Books *A Student's Guide to R* https://github.com/ProjectMOSAIC/LittleBooks/raw/master/StudentGuide/MOSAIC-StudentGuide.pdf, Kapitel 1, 2, 13
- Chester Ismay (2016): Getting used to R, RStudio, and R Markdown https://ismayc.github.io/rbasics-book/
- Maike Luhmann (2015): *R für Einsteiger*, Kapitel 1-8
- Daniel Wollschläger (2014): *Grundlagen der Datenanalyse mit R*, Kapitel 1-4

Lizenz

Versionshinweise:

- Datum erstellt: 2017-05-10
- R Version: 3.4.0
- `mosaic` Version: 0.14.4