Einführung in R

– in 90 Minuten

Jonas Beste & Arne Bethmann

9. Oktober 2015

Überblick

- In den nächsten knapp 90 Minuten werden Sie lernen selbstständig statistische Analysen in R durchzuführen. Das beinhaltet

  Ziel 1 das Arbeiten mit R-Studio

  Ziel 2 mathematischen Operationen, Funktionen und Zuweisungen

  Ziel 3 die Erstellung von Grafiken

  Ziel 4 den Zugang zu interne und externe Hilfe und Dokumentationen

  Ziel 5 das Arbeiten mit Skripten um Ergebnisse reproduzieren zu können

  Ziel 6 das Arbeiten mit Schleifen

  Ziel 7 den Einblick in zusätzliche R-Pakete (`foreign`, `data.table`, `dplyr`, , `ggplot2`)

Was ist R?

- R ist eine Rechensoftware für statistische Analysen
- Die Software ist frei zugänglich
- Es stehen viele Zusatzpakete und hilfreiche Blogs zur Verfügung
- Weitere Informationen zu R finden sich auf `http://www.r-project.org/`

R-Studio

- Mit R ist umfangreiches Arbeiten möglich
- Es empfiehlt sich jedoch, das Programm mit der Freeware R-Studio zu kombinieren
- Diese bietet eine gut strukturierte Benutzeroberfläche und einige Zusatzfunktionen
- Verfügbar über `http://www.rstudio.org/`

Arbeitsverzeichnis

- Das Arbeitsverzeichnis ist der Ort auf dem Rechner, in dem aktuell gearbeitet wird
- Vor Beginn sollte R mitgeteilt werden, wo die Daten und Skripte liegen oder abgespeichert werden sollen:

```
> setwd("directoryname")
```

Mathematische Operationen und logische Abfragen

- In R können mathematische Operationen durchgeführt werden:

```
> 4^2 + 8
[1] 24
```

- Und Aussagen auf ihre Richtigkeit geprüft werden:

```
> 2 == 1
[1] False
> 2 >= 1
[1] TRUE
```

Mathematische Operationen und logische Abfragen

- **Aufgabe**
  1. Spielen Sie ein wenig mit den mathematischen Operationen und den logische Abfragen
  2. Wie viel Prozent der Zeit einer Woche verbringen Sie mit der Vor- und Nachbereitung von Univeranstaltungen?

Zuweisungen

- Man kann Werten einen Namen zuweisen:

```
> a <- 2
> a * 3
[1] 6
```

- Diese Objekte können jederzeit überschrieben werden:

```
> a <- a + 5
> a
[1] 7
```

Zuweisungen

- Neben Skalaren können auch Vektoren erstellt werden:

```
> b <- c(1,3,5)
> b > 3
[1] FALSE FALSE TRUE
```

- Für Matrizen kann u.a. die Funktion `matrix` verwendet werden:

```
> mat = matrix(data=c(1,2,3,4,5,6), ncol=3)
> mat
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
```

Zuweisungen

- **Aufgabe**
  1. Berechnen Sie den Monatslohn bei 40 Wochenstunden und 10 Euro Stundenlohn. Verwenden Sie dabei Namen für die beiden Werte.
  2. Variieren Sie nun die Werte beider Variablen.

Funktionen

- In R und weiteren Paketen sind viele Funktionen enthalten
- Der natürliche Logarithmus von 100:

```
> log(100)
[1] 4.60517
```

- Der Logarithmus von 100 zur Basis 10:

```
> log(100, base = 10)
[1] 2
```

- Der Logarithmus von 100 zur Basis 10:

```
> mean(c(1,3,5))
[1] 3
```

Funktionen

- **Aufgabe**

  1. Berechnen Sie die Summe der Zahlen 3, 13, 7, 24 und 11. Füge Sie diese hierfür zunächst in einen Vektor zusammen und verwenden Sie dann die Funktion `sum()`

Plots

- Mit der Funktion `rnorm()` lassen sich Zufallszahlen generieren:

```
1 > x=rnorm(100)
2 > y=rnorm(100)
```

- Mit der Funktion `plot()` lassen sich Graphiken erstellen:

```
1 > plot(x,y)
2 > plot(x,y,xlab="Das ist die x-Achse",ylab=
3 + "Das ist die y-Achse", main="Plot von X und Y")
```

Hilfe und Dokumentation

- Für R gibt es eine große Menge an Dokumentationen und Hilfen
- Eine Kurzhilfe kann über den Befehl `help()` abgerufen werden:

```
> help(mean)
```

- Der Befehl `help.start()` startet die Onlinehilfe im Internetbrowser:

```
> help.start(mean)
```

- Beispiele sind über den Befehl `example()` verfügbar:

```
> example(mean)
```

Links

- Ausführliches Manual: `http://cran.r-project.org/doc/manuals/R-intro.pdf`
- Kurze Reference Card: `http://cran.r-project.org/doc/contrib/Short-refcard.pdf`
- R-Wiki: `http://wiki.r-project.org/rwiki/doku.php`
- Geheimtipp: Google (z.B. "r project mean").

Skript

- R verarbeitet Kommandos
- Die Kommando können in Skripten dokumentiert werden
- R-Skripte haben die Endung .R, z.B. umasds.R
- Teile des Codes können ausgeführt werden (hierbei werden diese an die Kommandozeile übergeben), indem man die Zeile markiert und CTRL+ENTER drückt
- Das komplette Skript kann über den Befehlt `source()` ausgeführt werden:

```
1 > source("umasds.R")
```

- Kommentare sind möglich, indem ein Rautezeichen # davor geschrieben wird. Alles darauf Folgende bleibt unberücksichtigt

Skript

- **Aufgabe**
  Erstellen Sie ein Skript mit dem Namen ErstesSkript.R. Dieses Skript soll Code enthalten, bei dem 100 zufällige Zahlen erzeugen und geplottet werden und der Mittelwert der Zahlen bestimmt werden. Bei Bedarf informieren sie sich über die unterschiedlichen Quellen über die Funktionen. Lassen Sie das Skript wiederholt durchlaufen. Denken Sie daran das Skript zu kommentieren.

Zugriff auf Objekte

- Elemente von Objekten können über `[i]` direkt angesprochen werden:

```
> b
[1] 1 3 5
> b[2]
[2] 3
```

- Bei Matrix können auch Spalten oder Zeilen angegeben werden:

```
 > mat[1,]
[1] 1 3 5
```

- Hierüber können Elemente ersetzt werden:

```
> mat[1,1] <- 7
> mat
     [,1] [,2] [,3]
[1,]    7    3    5
[2,]    2    4    6
```

Zugriff auf Objekte

- **Aufgabe**
  Erstellen Sie eine 3x3 Matrix. Ersetzen Sie einzelne Zellen der Matrix durch neue Werte.

Schleifen

- Mit Schleifen lässt sich eine größere Anzahl von Befehlen wiederholt anwenden
- An häufigsten nutzt man for-Schleifen:

```
> d = seq(from=1, to=10)
> e = c()
> for(i in 2 :12)
    {
    e[i] = d[i] * 10
    }
> e
[1] NA 20 30 40 50 60 70 80 90 100 NA NA
```

- Daneben gibt es noch repeat-Schleifen und while-Schleifen

If-Bedingungen

- Ebenfalls hilfreich sind if/else-Bedingungen:

```
> Gr50 <- ifelse(e > 50, "Gr50", "KlGl50")
> table(Gr50)
```

- Es gibt auch sehr kurze Varianten:

```
> f <- c(1,2,3,4)
> g <- f[f==1 | f==4]
> g
[1] 1 4
```

Schleifen und if-Bedingungen

- **Aufgabe**
  Erstellen Sie einen Vektor von 1 bis 100. Nutzen Sie hierfür die `seq` Funktion. Informieren Sie sich hierüber über die Hilfefunktion. Multiplizieren Sie mit Hilfe einer for-Schleife alle Elemente die kleiner 26 und größer 74 sind mit 10 und alle anderen Elemente mit 0,1.

Datenfiles laden und speichern

- Datenfiles können in R in unterschiedlicher Weise geladen und gespeichert werden, hier eine Variante:

```
> data = data.frame(a = c(1,2,3), b = c(4,5,6))
> data
  a b
1 1 4
2 2 5
3 3 6
> write.table(data, file="test.txt", row.names=F)
> data2 = read.table(file="test.txt", header=T)
> data2
  a b
1 1 4
2 2 5
3 3 6
```

- R-Datenfiles mit der Endung `.Rdt` und `.RData` lassen sich über die Befehle `load()` oder `data()` laden

Pakete verwenden

- Pakete enthalten Erweiterungen zu R
- Das umfasst Datenfiles, erweiterte Rechenoperationen, neue statistische Verfahren usw.
- Die verfügbaren Pakete sind unter `https://cran.r-project.org/web/packages/` einsehbar
- Die Installation von Programmpaketen erfolgt über den Befehl:

```
1 > install.packages ("Paketname")
```

- Das Aufrufen eines Pakets aus der Programmbibliothek erfolgt über den Befehl:

```
1 > library(Paketname)
```

- Mit dem Befehl `search()` erhält man eine Übersicht über alle gestarteten, mit `library()` über alle installierten Pakete.

Pakete foreign

- Paket zum Öffnen anderer Datentypen wie .SAV (SPSS) und .dta (Stata)
- `https://cran.r-project.org/web/packages/foreign/foreign.pdf`
- Installieren und laden:

```
> install.packages("foreign")
> library(foreign)
> help(read.dta)
> Data <- read.dta(".../test.dta", to.data.frame=T)
> help(data.table)
```

Pakete data.table

- Paket enthält Befehle zum einfacheren und schnelleren Umgang mit größeren Objekten
- It combines database like operations such as subset, with and by and provides similar joins that merge provides but faster
- `https://cran.r-project.org/web/packages/data.table/data.table.pdf`
- Installieren und laden:

```
1 > install.packages("data.table")
2 > library(data.table)
3 > help(data.table)
```

Pakete dplyr

- Umfangreiche Sammlung von Befehlen für einfache Datenaufbereitung
- `https://cran.r-project.org/web/packages/dplyr/dplyr.pdf`
- Installieren und laden:

```
> install.packages("dplyr")
> library(dplyr)
> help(filter)
> help(arrange)
> help(select)
> help(distinct)
> help(mutate)
> help(summarise)
```

Paket ggplot2

- Paket um elegante und aufwendige Graphiken zu erstellen
- Funktion qplot() (steht für quick plot) wenn schnell gehen soll
- `https://cran.r-project.org/web/packages/ggplot2/ggplot2.pdf`
- Installieren und laden:

```
1 > install.packages("ggplot2")
2 > library(ggplot2)
3 > help(qplot)
```

Ende

Herzlichen Glückwunsch!

Sie haben die ersten Schritte mit R bewältigt. Jetzt heißt es weitermachen. Am besten lernt man R in der Anwendung.

Was kommt als nächstes?

- Wir konnten hier nicht alle Möglichkeiten von R ansprechen. Informieren Sie sich jetzt oder bei Bedarf u.a. über:
  - Weitere Funktionen wie `head()`, `str()`, `table`, `summary()` u.v.m.
  - Rechnen mit Vektoren und Matrizen
  - Umgang mit fehlenden Werten (`NaN`, `NA`, `na.omit()`)
  - Weitere Pakete, z.B. `stringr`, `lattice`, `knitr` u.v.m.
  - Datentypen, z.B. `is.matrix()` oder `is.data.frame()`
  - Möglichkeiten Daten zu trennen, zusammenzufügen und zu sortieren (`subset()`, `cbind()`, `rbind()`, `sort()`, `order()`)
  - Zufallszahlen und Stichproben (`sample()`)
  - Eigene Funktionen schreiben mit `function()`
  - Modellierungen, hierfür stehen wie Pakete bereit