Einführung in die Statistiksoftware R und Anmerkungen zum Skript von Prof. Wellisch

(Prof. Dr. Wolfgang Bischof)

Rosenheim

16. März 2020

1 Einführung in die Statistiksoftware R

Unter dem Kürzel R verbirgt sich ein riesiges Softwarepaket, mit dem sich alle Schritte der grafischen und statistischen Datenanalyse durchführen lassen: vom Einlesen und Aufbereiten der Daten bis zur statischen Auswertung durch deskriptive, explorative und induktive Verfahren.
Die Tatsache, dass die ganze Welt - organisiert vom sogenannten R-core-Team - an der Weiterentwicklung des Open-source-Projekts R mitwirken kann (und dies auch fleißig tut), ist ein Garant dafür, dass Verfahren aus der gesamten Bandbreite der Statistik in R umgesetzt werden und einer kontinuierlichen Kontrolle unterliegen. Die unkomplizierte Möglichkeit, neue Verfahren in R umzusetzen, sorgt ferner dafür, dass diese Verfahren zuerst in R implementiert werden, bevor sie - wenn überhaupt - Eingang in kommerzieller statistischer Software finden. Mit diesem Konzept ist R im akademischen Bereich und zunehmend auch in der freien Wirtschaft zu *der* Standard-Statistiksoftware geworden. Die Anwendungsmöglichkeiten beinhalten inzwischen Verfahren aus der Künstlichen Intelligenz, dem maschinellen Lernen und dem Data Mining. Auch die automatische Erzeugung von regelmäßigen Berichten und die Erstellung von interaktive Web-Anwendungen sind mit R realisierbar.
R wurde ursprünglich als Open-Source-Variante von S und S-Plus entwickelt, einer ehemals verbreiteten Software für Statistik und Datenanalyse. Inzwischen ist R weiter verbreitet und viel umfangreicher als S und S-Plus.
Woher stammt der Name R? – Die ursprünglichen Entwickler von R, die Neuseeländer Rob Gentleman und Ross Ihaka spielten mit dem Namen R wohl auf ihre Vornamen an. Vor allem aber wollten sie erreichen, dass ihre Software alphabetisch vor S und S-Plus und anderer Statistiksoftware, deren Namen meist mit einem „S“ beginnt, steht.
R kann für alle gängigen Betriebssysteme auf der Internetseite

`https://cran.r-project.org`

heruntergeladen werden. Dort findet man auch zahlreiche Dokumentationen, unter anderem die empfehlenswerte Einleitung „An Introduction to R“.

1.1 Installation und Entwicklungsumgebung

R kann kostenlos von der Seite `https://cran.r-project.org` für das Windows-, Linux- und Mac-Betriebssystem heruntergeladen werden. Alle Varianten können automatisch auf Knopfdruck installiert werden. Selbst für die Smartphone-Betriebssysteme Android und iOS stehen R-Varianten zur Verfügung.
Als Entwicklungsumgebung empfehlen wir *RStudio*, das von der Seite `https://www.rstudio.com` ebenfalls kostenlos für das Windows-, Linux- und Mac-Betriebssystem heruntergeladen werden kann.
Typischerweise arbeitet man in R interaktiv in der sogenannten Konsole, die in *RStudio* als eigenes Fenster angezeigt wird. Im interaktiven Modus schreibt man einen Befehl, schickt ihn ab und R zeigt das Ergebnis an. Dann schreibt man den nächsten Befehl usw.

Beispiel:
Generiere zehn Binomial-verteilte Zufallszahlen (Anzahl Versuche: 100, Trefferwahrscheinlichkeit: 0.5) und berechne den Durchschnitt der quadrierten Zufallszahlen.
Dies bewerkstelligen wir, in dem wir im Konsolenfenster

```
mean(rbinom(10,100,0.5)^2)
```

eingeben. Diese Befehlskombination generiert zehn Binomial-verteilte Zufallszahlen, quadriert jede von ihnen, berechnet den Durchschnitt und gibt das Resultat aus.[1]

```
## [1] 2679.1
```

Die `[1]` *in der Ausgabezeile bedeutet, dass das erste Element, das in dieser Zeile ausgegeben wird, das Element mit Index 1 ist. Da in unserem Beispiel lediglich ein Element ausgegeben wird, ist die Angabe redundant. Die Notation ist aber hilfreich, wenn das Ergebnis aus vielen Elementen, die auf mehrere Zeilen verteilt sind, besteht. Wenn das Ergebnis beispielsweise aus zwei Zeilen mit bis zu 18 Elementen je Zeile besteht, wird die zweite Zeile mit* `[19]` *beschriftet:*

```
rbinom(30,100,0.5)

##  [1] 55 59 46 54 54 56 49 53 44 41 47 50 49 51 45 42 51 53
## [19] 55 44 48 46 50 42 39 43 52 47 44 53
```

Hier sieht man auf einen Blick, dass das 19.Element der Ausgabe den Wert 55 hat.

[1] Der String „`##`“ wird im R-Studio nicht ausgegeben. Er dient in unserem Skript dazu, die Ausgabe optisch von den Eingaben zu trennen.

Wenn man mehrere R-Befehle hintereinander ausführen möchte, ist es nützlich, diese Befehle in einem sogenannten Skriptfile, einer Datei mit Endung `.R`, abzuspeichern. Skriptfiles können im Editor der Entwicklungsumgebung (oder in jedem anderen Editor) bearbeitet werden und über den *Run*-Knopf oder den Befehl `source("dateiname.R")` ausgeführt werden.

Neben dem interaktiven Modus gibt es auch noch den *Batch*-Modus. Hier können Befehlsfolgen, die in einer Datei gespeichert wurden, automatisch und ohne das manuelle Starten des R-Systems ausgeführt werden. Dies ist beispielsweise nützlich, wenn jeden Tag die gleichen Berichte zu den aktuellen Daten erzeugt werden sollen. Gibt man in der Windows-Eingabeaufforderung[2] oder in einer Linux-Shell die Befehlszeile

```
R CMD BATCH Dateiname.R
```

ein, werden die Befehle, die in `Dateiname.R` stehen, im *Batch*-Modus ausgeführt.

1.2 Hilfesystem

Für den Umgang mit den verschiedenen R-Befehlen bzw. R-Funktionen ist das Hilfesystem, genauer die Hilfefunktion `help()` (oder `?`), sehr hilfreich. Durch `help(Funktionsname)` erhält man eine Erklärung der Syntax und der Wirkungsweise von Funktionsargumenten und Parametern einer R-Funktion. Weiter werden typische Anwendungsbeispiele der Funktionen aufgezeigt, die mit `example(Funktionsname)` auf einen Schlag ausgeführt werden können. Mit `??String` werden alle Hilfeseiten nach dem `String` durchsucht, was nützlich ist, wenn man nicht genau weiß, wie man den gesuchten Befehl schreibt.
Im *R-Studio* gibt es ein eigenes Hilfefenster, in dem die Hilfeseiten geöffnet werden und in dem man wie in einem Web-Browser navigieren kann.

1.3 Zusatzpakete

Über Zusatzpakete, auch *packages* genannt, kann die Funktionalität von R erweitert werden. Obwohl bereits im Grundsystem von R eine Vielzahl von Funktionen zum Datenmanagement, zur Grafik-Erzeugung und zur statistischen Analyse zur Verfügung stehen, kommt es immer wieder vor, dass man zur Bearbeitung einer Aufgabe spezielle Tools benötigt. Das kann die Anbindung an eine spezielle Datenbank sein, eine Bibliothek zur Erstellung besonders schöner Grafiken oder ein spezielles statistisches Verfahren. Meist gelangt man über eine Suche im Internet an den Namen des gewünschten Pakets, das zunächst einmalig installiert werden muss. Um das Paket und damit die darin enthaltenen R-Funktionen während einer R-Sitzung zur Verfügung zu haben, muss das Paket innerhalb jeder R-Sitzung gela-

[2] Die Windows-Eingabeaufforderung startet man mit dem Befehl `cmd.exe`.

den werden.[3] Nach der Bereitstellung eines Pakets erhält man über `?(Paket-Name)` eine Beschreibung des Pakets und seiner Funktionen.
Ohne größere Schwierigkeiten können auch eigene Pakete geschrieben und an andere verteilt werden. Wie das im Detail geht, steht in der Anleitung *Writing R Extensions*, die man unter *Tutorials* auf `https://cran.r-project.org` findet.

2 Anmerkungen zum Skript von Prof. Wellisch

2.1 Anwenden von Funktionen auf Spalten von Matrizen oder Datentabellen (Ergänzungen zu den Abschn. 2.7.5 und 2.7.6)

Mit den Funktionen `apply` (für Matrizen) und `sapply` (für Datentabellen) können beliebige Funktionen (auch selbst geschriebene) auf jede Spalte einer Matrix oder einer Datentabellen ausgeführt werden. Als Ergebnis erhält man einen Vektor, dessen Länge der Spaltenzahl entspricht. Damit lassen sich häufig Schleifen vermeiden und elegante und effiziente Befehlszeilen schreiben.

Beispiele:

1. *In der Matrix A stehen je Spalte die Schulnoten von Klaus, Anna und Josef in den Fächern Deutsch (erste Zeile), Mathematik (zweite Zeile) und HSU (dritte Zeile). Nun soll für jeden Schüler der Durchschnitt dieser drei Noten gebildet werden.*

```
A <- matrix( c(1,2,3,3,2,4,3,4,4), nrow=3 )
# (a) Schleifenvariante
durchschnitt <- c() # Deklaration als Vektor (notwendig!)
for (i in 1:3)
  durchschnitt[i] <- mean( A[,i] )
# (b) eleganter und kürzer mit apply (keine Deklaration nötig!)
durchschnitt <-apply( A, 2, mean ) # 2 = Spalte
```

2. *In der Datentabelle* `dat.stud` *soll für jedes Merkmal geprüft werden, für wie viele Beobachtungen keine Werte vorhanden sind.*

```
dat.stud <- data.frame( geschlecht = c("m","w","w",NA,"m"),
                        matnr = c(443, NA, NA, 446, 447),
                        fach = c("WMA","WIF","WMA","IAB","INF") )
```

[3] Am bequemsten installiert und lädt man die Pakete über den Reiter *Packages* in R-Studio. Alternativ kann man die Befehle `install.packages(Paket-Name)` (einmalig) und `library(Paket-Name)` (in jeder Sitzung) in der Konsole verwenden.

```
# (a) Schleifenvariante
anz.NAs <- c() # Deklaration als Vektor
anz.merkmale <- length(dat.stud) # Anzahl der Merkmale
for (i in 1 : anz.merkmale )
  anz.NAs[i] <- sum( is.na(dat.stud[,i]) )
anz.NAs

## [1] 1 2 0

# (b) eleganter mit sapply (weder Deklaration noch Bestimmung
#                           der Anz. der Merkmale nötig!)
anz.NAs <- sapply( dat.stud, function(x){ sum( is.na(x) ) } )
anz.NAs # Ausgabe mit den Bezeichnungen der Merkmale!

## geschlecht      matnr       fach
##          1          2          0
```

2.2 Selektionen (Ergänzung zu Abschnitt 2.8)

Alternativ zum `subset`-Kommando kann man auch direkt per Indexselektion Zeilen und Spalten aus einer Datentabelle auswählen. So kann man die letzte Selektion des Abschnitts 2.8 auch wie folgt durchführen:[4]

```
attach(Daten2)
Daten4 <- Daten2[Geschlecht=="m" & Alter < 28,c("Gewicht","Alter")]
# oder
Daten4 <- Daten2[Geschlecht=="m" & Alter < 28,2:3]
```

[4]Es besteht ein kleiner Unterschied zwischen der Selektion mit dem `subset`-Kommando und der Indexselektion: Wenn der Wert des Selektionsmerkmals fehlt (also `NA` ist), wird die entsprechende Zeile beim `subset`-Befehl nicht ausgewählt; bei der Indexselektion wird die Zeile ausgewählt und alle Werte auf `NA` gesetzt. So liefert bei der `dat.stud`-Datentabelle aus dem letzten Abschnitt `subset(dat.stud, geschlecht == m")` die Ausgabe:

```
   geschlecht  matnr  fach
1  m           443    WMA
5  m           447    INF
```

während die Indexselektion `attach(dat.stud); dat.stud[geschlecht == m",]` die Ausgabe:

```
    geschlecht  matnr  fach
1   m           443    WMA
NA  <NA>        NA     <NA>
5   m           447    INF
```

liefert.