Kompaktwissen zu R

1. Hilfe

Allgemein	
Manuale und Archive von Mailinglisten	R-Homepage: www.R-Project.org
Beispiele hilfreicher Internetseiten (mit weiteren Verweisen zu Einführungen und Tutorien)	http://de.wikibooks.org/wiki/GNU_R http://www.statmethods.net/index.html
Bücher	z.B. Ligges (2005): Programmieren mit R
In R	
Starten des Hilfesystems im Browser	help.start()
Hilfe zu einer Funktion	?Funktionsname help(Funktionsname)
nach Schlagwort suchen	help.search(„Schlagwort")
ähnliche Funktionen suchen	apropos(„Funktionsname")

2. „Grammatik“

„Satzbau“

Zuweisung	`<-` `=` (funktioniert, ist aber nicht zu empfehlen)
Objekt	Alles ist ein Objekt (Daten, Funktionen, Ergebnisse,…)
Objekt definieren	Objektname sollte mit Buchstaben beginnen, darf aber auch Zahlen und Punkte/Unterstriche enthalten
	Groß- und Kleinschreibung wird unterschieden
Funktion	Funktionsname mit Argumenten in runden Klammern, zB: `table( Variable1, Variable2)`
Kommentar	#
R-Konsole	Befehle dürfen sich über mehrere Zeilen erstrecken, Zeilenbeginn ist dann mit + statt > markiert.
Pfadangaben	statt üblichem / muss // oder \ verwendet werden

Datenstrukturen

Je nach Datenstruktur können bestimmte Funktionen angewendet werden. Bei unpassender Datenstruktur gibt es Fehlermeldungen oder falsche Ergebnisse.
Häufigster Fehlergrund: `factor` statt `numeric.`
Teilweise können Daten direkt in andere Datenstruktur überführt werden (zB `as.numeric`).

Vektor	`vector`
logical	`TRUE` oder `FALSE`
numeric [integer]	Ganzzahlen
numeric [double]	reele Zahlen in doppelter Maschinengenauigkeit
complex	Komplexe Zahlen
character	Zeichenketten (strings)
Matrix	`matrix` (mehrere Spalten bzw Zeilen von Vektoren)
Array	`array` (“dreidimensinale Matrix”)
Dataframe	`data.frame` (Matrix, bei der jede Spalte anderes Format haben darf)
Liste	`list` (jeder Eintrag ist eigenes Objekt beliebigen Formates)
Abfragen der Klasse	`class()`

3. „Vokabeln“

Logik

Vergleiche	`==, !=, <, >, <=, >=`
Konstanten	`TRUE, FALSE`

Operatoren	`! (Negation)`
	`&, | bzw. &&, || (und, oder)`
	`any() (ist irgendein Element TRUE?)`
	`all() (sind alle Elemente TRUE?)`
	`xor() (ausschließendes oder)`

Vektoren

Erzeugen eines Vektors	
Zusammenfügen von Elementen	`c()`
Folgen	`seq(Anfang, Ende, by=Abstand)` `seq(along = Objekt)`
	`Anfang:Ende`
Wiederholungen	`rep(Objekt, Anzahl)` `rep(Objekt, each = Anzahl)`
Indizieren von Vektoren	
n-tes Element aufrufen	`x[n]`
Vektor (vec) mehrerer Elemente aufrufen	`x[vec]`
Element(e) entfernen	`x[-vec]`
Benanntes Element aufrufen	`x[„Name“]`
Elemente über TRUE/FALSE-Vektor (muss dieselbe Länge wie der Vektor haben) aufrufen	`x[logikvektor]`

Grundlegendes Rechnen

Grundlegende Operatoren	`+, -, *, /, ^`
Extremwerte, Betrag	`min(), max(), abs()`
Summe, Produkt	`sum(), prod()`
Runden	`round(), floor(), ceiling()`
Wurzel	`sqrt()`
Logarithmen	`log(), log10(), log2()`
Exponentialfunktion	`exp()`
trigonometrische Funktionen	`sin(), cos(), tan()`

Unendlichkeit	`Inf, -Inf`
nicht definiert	`NaN (not a number)`
fehlende Werte	`NA (not available)`
leere Menge	`NULL`

Nützliche Funktionen

Welche Elemente erfüllen eine gegebene Bedingung?	`which()`
Länge eines Vektors	`length()`
transponieren	`t()`
Sortieren	`sort()`
Ränge bilden (ohne/ mit Entfernen von Bindungen)	`rank(), order()`
Überprüfen von exakter Gleichheit zweier Objekte	`identical()`
Überprüfen von Gleichheit auf Maschinen-Rechengenauigkeit	`all.equal()`
Entfernen mehrfach vorkommender Elemente aus einem Vektor	`unique()`
Sind Werte mehrfach vorhanden?	`duplicated()`
Untermenge abfragen (zB Alter *a* in 18 bis 65)	`a %in% b`
kumulative Summe/ Produkt	`cumsum(), cumprod()`
Kontinuierliche Variable kategorisieren	`library(car)` `findInterval()`

Matrizen

Erzeugen einer Matrix	`matrix(data = Datenvektor,`
	`nrow = Zeilenanzahl,`
	`ncol = Spaltenanzahl, byrow = FALSE)`
	(„byrow = TRUE“ baut die Matrix zeilenweise statt spaltenweise auf)

Indizierung
der n-ten Zeile
der m-ten Spalte
eines Elementes
	
	`M[n, ]`
	`M[ ,m]`
	`M[n,m]`

Benannte Spalte aufrufen	`M$Name`
Matrixmultiplikation	`%*%`
Abfragen und Setzen der Hauptdiagonalen	`diag()`
Zeilen-, Spaltennamen	`dimnames(), rownames(), colnames()`
Anzahl Zeilen, Spalten	`nrow(), ncol()`
Invertieren einer Matrix	`solve()`

Listen

Erzeugen einer Liste	`list(Element1, Element2, …)`
Erzeugen einer benannten Liste	`list(E1 = Element1, E2 = Element2, …)`
Aufrufen des n-ten Elementes	`L[[n]]`
Aufrufen eines benannten Elementes	`L$Name, L[[„Name“]]`

Data Frames

Erzeugen eines Data Frames	`data.frame(Name1 =Vektor1, Name2 =Vektor2,…)`
Indizierung	wie bei Matrix
Untermengen extrahieren	`subset(Dataframe, Bedingung)`
Datensätze aufteilen und zusammenfügen	`split(), merge()`

Daten einlesen

Textdatei (.txt)	`read.table()`
Excel-Datei	Einfachster Weg: Excel-Datei als .csv speichern, dann `read.csv(), read.csv2()`
SPSS-Datei	`library(foreign)` `read.spss()`
STATA-Datei	`library(foreign)` `read.dta()`
SAS-Datei	(auf Rechner ohne SAS:) SAS System Viewer (kostenloser Download) zum Öffnen des SAS Datensatzes und Speichern als .csv, dann `read.csv(), read.csv2()`

Auch Datenbanken (OBDC, MySQL, …) sind möglich.

Daten speichern

Textdatei (.txt)	`write.table()`
Excel-Datei	`write.csv(), write.csv2()`
RData	`save(data, file=„Name.RData")`

Datumsangaben

SPSS-Datum nach R umkodieren	`library(date); library(chron)`
	`SPSS_Datum <-`
	`  as.date( as.chron(ISOdate(1582, 10, 14)`
	`  + SPSS_Datum) )`

Pakete

Paket installieren	`install.packages("Name")` oder: http://www.cran.r-project.org/ → Packages → Windows Binary (.zip) in library-Order des R-Programmes entpacken
Paket aufrufen	`library(Paketname)`

Statistische Maßzahlen

Lagemaße	
Arithmetisches Mittel	`mean()`
Median	`median()`
Häufigkeitstabelle	`table()` `prop.table(Kreuztabelle)`
Streuungsmaße	
Varianz	`var()`
Standardabweichung	`sd()`
Spannweite (d.h. Minimum, Maximum)	`range(), min(), max()`
Zusammenhangsmaße	
Korrelation	`cor()`
Kovarianz	`cov()`
Zusammenfassung	`summary()`

Bei fehlenden Werten ist das Ergebnis NA. Zum Ausschluss dieser Werte das Argument na.rm=TRUE angeben.

Regressionsmodelle und Tests

Lineare Regression	`lm( Outcome ~ Prädiktor )`
Verallgemeinerte lineare Modelle (logistische, Poisson, …)	`glm(, family=binomial)` `glm(, family=poisson)`
Zusammenfassung	`summary()`
Diagnoseplots	`plot()`

t-Test	`t.test()`
Chi-Quadrat-Test	`chisq.test()`

Grafiken

Allgemeine Plot-Funktion	`plot()`
Boxplot	`boxplot()`
Histogramm	`hist()`
Säulendiagramm, Stabdiagramm	`barplot()`
Quantile-Quantile-Plot	`qqnorm()`
Einige Argumente für eine neue Grafik:	
Bereich für x- und y-Achse	`xlim, ylim`
Punkt-Symbol (**p**lotting **ch**aracter), Farbe, Linientyp	`pch, col, lty`
labels der x- und y-Achse	`xlab, ylab`
Überschrift, „Unterschrift“	`main, sub`
Hinzufügen von…	
… einer Geraden	`lines()` `abline()`
… Punkten	`points()`
… einer Legende	`legend()`
… Text	`mtext()`