

Elementare Befehle

Kommandos

<code>ls()</code> oder <code>objects()</code>	Auflistung der Objekte im Workspace
<code>rm(object)</code>	<code>object</code> wird aus dem Workspace entfernt
<code>help(funktion)</code> oder <code>?funktion</code>	Zeigt die Hilfeseite einer Funktion
<code>example(funktion)</code>	Zeigt Beispielaufufe einer Funktion

Variablennamen

Variablen werden aus Buchstaben (Groß- und Kleinschreibung beachten!), Ziffern und dem Punkt „.“ gebildet. Sie dürfen nicht mit Ziffern beginnen und sollten nicht mit dem Punkt beginnen.

Zuweisungen

<code><-</code>	Übergabe eines Werts an eine Variable
<code>-></code>	Übergabe eines Werts an eine Variable nach rechts

Operatoren

Arithmetische Operationen

<code>+</code>	Addition
<code>-</code>	Subtraktion
<code>*</code>	Multiplikation
<code>/</code>	Division
<code>^</code>	Potenzierung
<code>%%</code>	Division mit Rest
<code>%/%</code>	Rest einer Division

Logische Operationen

<code>==</code>	gleich
<code>!=</code>	ungleich
<code><</code>	kleiner
<code>></code>	größer
<code><=</code>	kleiner gleich
<code>>=</code>	größer gleich
<code>&</code>	logisches UND
<code> </code>	logisches ODER
<code>!</code>	logisches NICHT

Vektoren und Datenstrukturen

Generierung von Vektoren

<code>numeric(10)</code>	ein Vektor mit 10 Nullen
<code>character(10)</code>	10 × " "
<code>logical(10)</code>	10 × FALSE
<code>seq(0,1,0.1)</code>	generiert den Vektor 0 0.1 ... 0.9 1
<code>1:10</code>	wie <code>seq(1,10,1)</code>
<code>c(1,2,3)</code>	generiert den Vektor 1 2 3
<code>rep(c(2,7),2)</code>	erzeugt den Vektor 2 7 2 7

Umwandlung

<code>as.character(x)</code>	wandelt in Zeichenketten-Variable um
<code>as.numeric(x)</code>	wandelt in numerische Variable um
<code>factor(x)</code>	konvertiert in einen nominalskalierten Vektor

Datentabellen

<code>data.frame(x,y)</code>	erzeugt Datentabelle mit den Spalten <code>x</code> und <code>y</code>
<code>data.frame(age=x,height=y)</code>	erzeugt Datensatz mit zwei benannten Vektoren <code>x</code> und <code>y</code>
<code>attach(data)</code>	<code>data</code> wird in den Suchpfad kopiert
<code>detach(data)</code>	<code>data</code> wird wieder aus dem Suchpfad gelöscht

Einlesen von Datensätzen

<code>read.table("Pfadname")</code>	liest externen Datensatz ein
-------------------------------------	------------------------------

Gebräuchliche Argumente der read.table-Funktion

<code>header=TRUE</code>	die erste Zeile wird als Zeile der Variablennamen aufgefasst
<code>sep=","</code>	Daten werden durch Kommata getrennt
<code>dec=","</code>	Komma ist Dezimalkomma (üblich: Punkt)
<code>na.strings="."</code>	Punkte für fehlende Einträge

Varianten von read.table

<code>read.csv("Pfadname")</code>	liest durch Kommata getrennte Spalten
<code>read.csv2("Pfadname")</code>	liest durch Semikola getrennte Spalten, Komma als Dezimalkomma
<code>read.delim("Pfadname")</code>	liest Tab-getrennte Spalten
<code>read.delim2("Pfadname")</code>	liest Tab-getrennte Spalten, Komma als Dezimalkomma

Indizierung und Komponentensteuerung

Vektoren

<code>x[i]</code>	gibt die i -te Komponente des Vektors <code>x</code> aus
<code>x[1:5]</code>	gibt die ersten 5 Komponenten von <code>x</code> aus
<code>x[c(2,3,5)]</code>	gibt die 2., 3. und 5. Komponente des Vektors <code>x</code> aus
<code>x[y<=30]</code>	gibt den Vektor derjenigen Komponenten x_i aus, für die $y_i \leq 30$ ist
<code>which[y<=30]</code>	gibt die Positionen derjenigen Komponenten y_i aus, für die $y_i \leq 30$ ist
<code>x[sex="male"]</code>	Selektion anhand eines qualitativen Merkmals

Matrizen und Datentabellen

<code>dfr\$body</code>	gibt den Vektor <code>body</code> der Datentabelle <code>dfr</code> aus
<code>dfr[dfr\$body<=70]</code>	gibt alle Zeilen der Datentabelle <code>dfr</code> aus, die in der Spalte <code>body</code> einen Wert ≤ 70 haben
<code>subset(dfr,body<=70)</code>	wie oben, in vielen Situationen einfacher
<code>m[4,]</code>	gibt die 4. Zeile der Matrix/Datentabelle <code>m</code> aus
<code>m[,3]</code>	gibt die 3. Spalte von <code>m</code> aus

Numerische und logische Funktionen

Elementare Funktionen

<code>abs(x)</code>	Betrag von x
<code>choose(n,k)</code>	Binomialkoeffizient $\binom{n}{k}$
<code>factorial(n)</code>	Fakultät von n
<code>log(x)</code>	natürlicher Logarithmus von x
<code>log10(x)</code>	10er Logarithmus von x
<code>exp(x)</code>	e^x
<code>sin(x)</code>	Sinus von x
<code>cos(x)</code>	Kosinus von x
<code>tan(x)</code>	Tangens von x
<code>asin(x)</code>	Arkussinus von x
<code>acos(x)</code>	Arkuskosinus von x
<code>atan(x)</code>	Arkustangens von x

Funktionen für Vektoren

<code>sort(x)</code>	Aufsteigende Sortierung des Vektors x
<code>length(x)</code>	Anzahl der Komponenten des Vektors x
<code>any(x,cond)</code>	Gibt TRUE zurück, wenn eine Komponente die Bedingung <code>cond</code> erfüllt
<code>all(x,cond)</code>	Gibt TRUE zurück, wenn alle Komponenten die Bedingung <code>cond</code> erfüllen
<code>min(x)</code>	Minimum der Komponenten des Vektors x
<code>min(x₁, ..., x_n)</code>	Min. der Komponenten der Vektoren x_1, \dots, x_n
<code>max(x)</code>	Maximum der Komponenten des Vektors x
<code>range(x)</code>	wie <code>c(min(x), max(x))</code>
<code>pmin(x₁, ..., x_n)</code>	wie <code>c(min(c(x₁[1], ..., x_n[1])), ...)</code>
<code>pmax(x)</code>	analog zu <code>pmin</code>
<code>sum(x)</code>	Summe der Komponenten des Vektors x
<code>prod(x)</code>	Produkt der Komponenten des Vektors x
<code>cumsum(x)</code>	Vektor der kumulierten Summen der Komponenten von x
<code>cumprod(x)</code>	Vektor der kumulierten Produkte der Komponenten von x

Funktionen für Matrizen

<code>t(m)</code>	Transponiert die Matrix m
<code>m %*% n</code>	Matrizenmultiplikation von m und n
<code>det(m)</code>	Determinante von m
<code>colSums(m)</code>	Spaltensummen von m
<code>rowSums(m)</code>	Zeilensummen von m
<code>colMeans(m)</code>	Mittelwerte der Spalten von m
<code>rowMeans(m)</code>	Mittelwerte der Zeilen von m

Funktionen aus der Statistik

<code>mean(x)</code>	Durchschnitt von x : $\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i$
<code>var(x)</code>	Stichprobenvarianz von x : $\frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})^2$
<code>sd(x)</code>	Standardabweichung, wie <code>sqrt(var(x))</code>
<code>cor(x,y)</code>	Korrelationskoeffizientenmatrix von x und y
<code>quantile(x,p)</code>	p-Quantil von x
<code>median(x)</code>	Median von x , wie <code>quantile(x,0.5)</code>
<code>summary(x)</code>	Zusammenstellung wichtiger Statistiken von x
<code>cut(x,breaks)</code>	teilt den Wertebereich von x in Intervalle mit den in <code>breaks</code> angegebenen Intervallgrenzen und kodiert die Werte von x entsprechend der Klasse, in die sie fallen
<code>table(x)</code>	Kontingenztafel von x
<code>prop.table(x,margin)</code>	bedingte Häufigkeiten gegeben die Zeilen, falls <code>margin=1</code> bedingen unter Spalten mit <code>margin=2</code>
<code>order(x)</code>	Ordnungsstatistik angewandt auf x
<code>rank(x)</code>	gibt die Rangwerte des numerischen Vektors x wieder; bei Bindungen oder Ties werden mittlere Ränge berechnet
<code>lm(y~x)</code>	bestimmt Regressionsparameter der lin. Regression der abhängigen Variablen y und der unabh. x

Wahrscheinlichkeitsverteilungen

Normalverteilungen

<code>dnorm(x)</code>	Dichte der Standardnormalverteilung in x
<code>pnorm(x)</code>	Verteilungsfunktion der Standardnormalverteilung in x
<code>qnorm(p)</code>	p -Quantil der Standardnormalverteilung, d. h., <code>qnorm(p)</code> ist das x mit $\mathcal{N}(0,1)((-\infty, x]) = p$
<code>rnorm(n)</code>	erzeugt n (Pseudo-) Zufallszahlen gemäß der $\mathcal{N}(0,1)$ -Vtlg.

Andere Verteilungen

<code>dbinom(x,n,p)</code>	Binomialverteilung mit Parametern n und p
<code>pgeom(x,p)</code>	Geometrische-Verteilung mit Parameter p
<code>phyper(x,m,n,k)</code>	Hypergeometrische-Verteilung
<code>pnbinom(x,size,prob)</code>	negative Binomialverteilung
<code>ppois(x,lambda)</code>	Poisson(λ)-Verteilung
<code>punif(x,a,b)</code>	Gleichverteilung auf (a,b)
<code>pexp(x,rate)</code>	Exponentialverteilung
<code>pbeta(x,a,b)</code>	$\beta(a,b)$ -Verteilung
<code>pchisq(x,df)</code>	χ^2 -Verteilung
<code>pf(x,df1,df2)</code>	F -Verteilung
<code>pgamma(x,a,b)</code>	$\Gamma(a,b)$ -Verteilung
<code>pweibull(x,a,b)</code>	Weibull-Verteilung mit Parametern a und b
<code>pcauchy(x,a,b)</code>	Cauchy-Verteilung mit Parametern a und b
<code>plnorm(x,mean,sd)</code>	Lognormalverteilung
<code>plogis(x,location,scale)</code>	Logistische Verteilung
<code>pnorm(x,mean,sd)</code>	Normalverteilung
<code>psignrank(x,n)</code>	Verteilung der Wilcoxon'schen Vorzeichen-Statistik
<code>pt(x,df)</code>	Student'sche t -Vtlg. (<code>df</code> für <i>degrees of freedom</i>)
<code>pwilcox(x,m,n)</code>	Vtlg. der Wilcoxon'schen Rangsummenstatistik für Samples der Größe m und n

Ersetzt man das **p** in den Befehlen durch **d** oder **q**, so erhält man die Dichte oder ein Quantil der entsprechenden Verteilung. Ersetzt man das **p** durch ein **r**, so erhält man eine Randomisierung (auch: *Sample*) gemäß der jeweiligen Verteilung.

Tests

Testen in R

<code>binom.test</code>	Exakter Binomialtest
<code>chisq.test</code>	Chiquadrat-Anpassungstest
<code>ks.test</code>	Kolmogoroff-Smirnoff-Test
<code>shapiro.test</code>	Shapiro-Wilk-Normalverteilungstest
<code>t.test</code>	Student'scher <i>t</i> -Test und Welch'scher <i>t</i> -Test
<code>wilcox.test</code>	Wilcoxon-Vorzeichen-Rangtest und Wilcoxon-Rangsummentest

Graphiken

Standard-Plots

<code>barplot()</code>	Säulendiagramm
<code>boxplot()</code>	Boxplot
<code>curve(f(x),a,b)</code>	zeichnet den Graphen der (stetigen) Funktion <i>f</i> auf <i>[a,b]</i>
<code>hist()</code>	Histogramm
<code>interaction.plot()</code>	Profildiagramm
<code>mosaicplot()</code>	Mosaikplot
<code>plot()</code>	Standardplot
<code>plot(ecdf(x))</code>	Empirische Verteilungsfunktion des Vektors <i>x</i>
<code>pie()</code>	Kreisdiagramm
<code>qqplot()</code>	QQ-Plot
<code>qqnorm()</code>	NQ-Plot

Zusätzliche Plot-Elemente

<code>abline(a,b)</code>	fügt Gerade mit Achsenabschnitt <i>a</i> und Steigung <i>b</i> in bestehenden Plot ein
<code>abline(h=u)</code>	wie oben; jedoch Horizontale in <i>u</i>
<code>abline(v=u)</code>	wie oben; jedoch Vertikale in <i>u</i>
<code>box()</code>	erstellt einen Rahmen um die Graphik
<code>legend()</code>	Legende
<code>lines()</code>	zusätzliche Linienelemente für einen Plot
<code>points()</code>	zusätzliche Punkte für einen Plot
<code>title()</code>	erstellt Titel (und Untertitel) für den letzten Plot

Graphikparameter

<code>expression</code>	mit <code>xlab</code> und <code>ylab</code> , erlaubt das Einsetzen von mathematischen Termen (z. B. mit Index)
<code>lty, lwd</code>	Linientyp bzw. Linienstärke
<code>mfrow, mfcoll</code>	mehrere Graphiken in einem Bild (multiframe)
<code>pch</code>	Einstellung des plotting characters
<code>xlim=c(0,9)</code>	die Graphik zeigt den <i>x</i> -Achsenabschnitt von 0 bis 9
<code>ylim=c(0,9)</code>	wie <code>xlim</code> für die <i>y</i> -Achse

Speichern von Graphiken

<code>pdf("Pfadname")</code>	Speichert die Graphik des folgenden <code>plot</code> -Befehls in eine <i>pdf</i> Datei
<code>jpeg("Pfadname")</code>	Speichert die Graphik des folgenden <code>plot</code> -Befehls in eine <i>jpg</i> Datei
<code>png("Pfadname")</code>	Speichert die Graphik des folgenden <code>plot</code> -Befehls in eine <i>png</i> Datei
<code>bmp("Pfadname")</code>	Speichert die Graphik des folgenden <code>plot</code> -Befehls in eine <i>bmp</i> Datei
<code>tiff("Pfadname")</code>	Speichert die Graphik des folgenden <code>plot</code> -Befehls in eine <i>tiff</i> Datei

Im Anschluss des `plot`-Befehls muss `dev.off()` aufgerufen werden.

Beispiel: `pdf("Pfadname"); plot(ecdf(rnorm(15))); dev.off();`

Programmierung

<code>if(cond){commands}</code>	Ausführung der Befehle <code>commands</code> , wenn der Wert der Formel <code>cond</code> TRUE ist
<code>else{commands}</code>	alternative Arbeitsanweisungen (im Anschluss an einen <code>if</code> -Befehl)
<code>for(i in x){commands}</code>	Schleife über die Werte des Vektors <code>x</code>
<code>while(cond){commands}</code>	Wiederholung der Befehle <code>commands</code> solange der Wert der Formel <code>cond</code> TRUE ist
<code>repeat {commands}</code>	(erstmal) unbegrenzte Wiederholung der Befehle <code>commands</code>
<code>if(cond) break</code>	wenn der Wert der Formel <code>cond</code> TRUE ist, wird die aktuelle <code>repeat</code> -Schleife abgebrochen