



PD Dr. Matthias Fischer

Matthias.Fischer@wiso.uni-erlangen.de

Version 2.0

## Warum R?

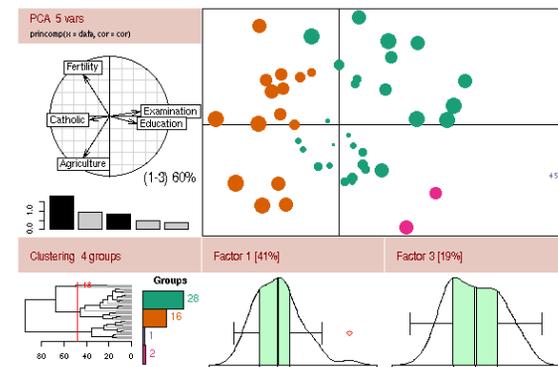
- 1 Dynamisches Tool zur (grafischen) Datenanalyse
- 2 Taschenrechnerfunktion
- 3 Programmiersprache
- 4 Freeware-Version von S-PLUS
- 5 Alternative zu MATLAB, GAUSS, OX, etc.
- 6 Ergänzung zu Excel oder SPSS

Kommerzielle Software	Freeware
S-Plus <a href="http://www.insightful.com/products/splus/">www.insightful.com/products/splus/</a>	R <a href="http://www.r-project.org/">www.r-project.org/</a>
SPSS <a href="http://www.spss.com/">www.spss.com/</a>	PSPP <a href="http://www.gnu.org/software/pspp/pspp.html">www.gnu.org/software/pspp/pspp.html</a>
NCSS <a href="http://www.ncss.com/">www.ncss.com/</a>	
Matlab <a href="http://www.mathworks.de/">www.mathworks.de/</a>	Octave, Scilab <a href="http://www.scilab.org/">www.scilab.org/</a>
Eviews <a href="http://www.eviews.com">www.eviews.com</a>	
Ox <a href="http://www.oxmetrics.net/">www.oxmetrics.net/</a>	
SAS <a href="http://www.sas.com/">www.sas.com/</a>	
STATA <a href="http://www.stata.com/">www.stata.com/</a>	LIMDEP <a href="http://www.limdep.com/">www.limdep.com/</a>
	JMulti <a href="http://www.jmulti.de/">www.jmulti.de/</a>



## R-Homepage

<http://www.r-project.org/>

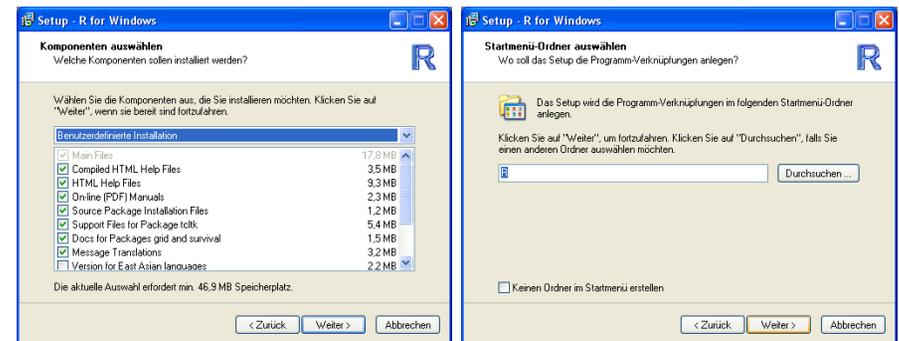
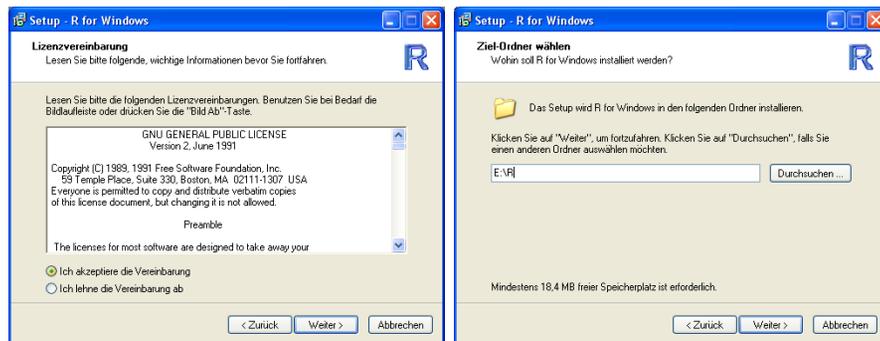
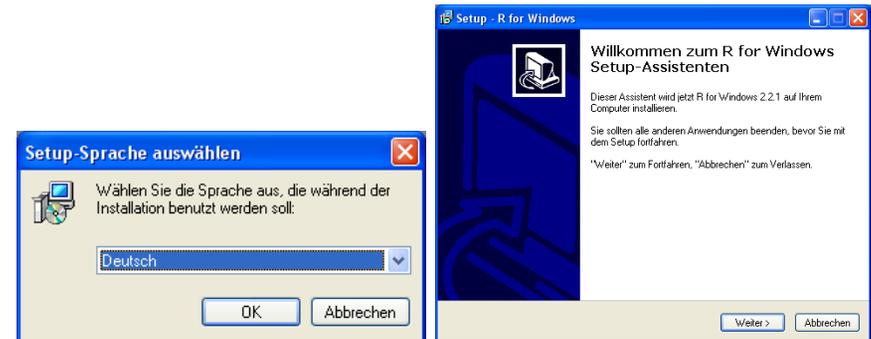


Download, Dokumentation, Newsgroups

## Installation



Start der Datei R-2.2.1-win32.exe





## Literatur



## R-Bücher

- Behr  
*Einführung in die Statistik mit R.*  
Verlag Vahlen, München, 2005.
- Crawley  
*Statistics: An Introduction using R.*  
Wiley, Chichester, 2005.
- Dolíc  
*Statistik mit R.*  
Oldenbourg, München, 2004.
- Dalgaard  
*Introductory Statistics with R Series: Statistics and Computing.*  
Springer, Berlin, 2004.

## R-Bücher

- Everitt  
*An R and S-Plus Companion to Multivariate Analysis.*  
Springer-Verlag, Heidelberg, 2005.
- Everitt/Hothorn  
*A Handbook of Statistical Analyses using R.*  
Chapman & Hall/CRC, Boca Raton, 2006
- Faraway  
*Extending Linear Models with R: Generalized Linear, Mixed Effects and Nonparametric Regression Models.*  
Chapman & Hall/CRC, Boca Raton, 2006.
- Ligges  
*Programmieren mit R.*  
Springer-Verlag, Heidelberg, 2005.

## R-Bücher

-  Maindonald/Braun  
*Data Analysis and Graphics Using R - An Example-Based Approach.*  
Cambridge University Press, Cambridge, 2003.
-  Murrell  
*R Graphics.*  
Chapman & Hall/CRC, Boca Raton, 2005.
-  Pfaff  
*Analysis of Integrated and Cointegrated Time Series with R.*  
Springer, Heidelberg, 2006.
-  Verzani  
*Using R for Introductory Statistics.*  
Chapman & Hall/CRC, Boca Raton, 2005.

## R-Skripten

-  Verzani (2002)  
*simpleR - Using R for Introductory Statistics.*  
<http://cran.r-project.org/doc/contrib/Verzani-SimpleR.pdf>.
-  Paradis (2002)  
*R for Beginners.*  
[http://cran.r-project.org/doc/contrib/Paradis-rdebuts\\_en.pdf](http://cran.r-project.org/doc/contrib/Paradis-rdebuts_en.pdf).
-  Faraway (2002)  
*Practical Regression and Anova using R.*  
<http://cran.r-project.org/doc/contrib/Faraway-PRA.pdf>.
-  Handl (2005)  
*Einführung in die Statistik mit R.*  
<http://www.wiwi.uni-bielefeld.de/~frohn/Lehre/Statistik1/Skript/stat12b.pdf>.

## R-Reference Cards

-  Tom Short (2002)  
*R Reference card.*  
<http://cran.r-project.org/doc/contrib/Short-refcard.pdf>.
-  John Baron (2004)  
*R Reference card.*  
<http://cran.r-project.org/doc/contrib/refcard.pdf>.
-  Diethelm Würtz (2006)  
*R/RMetrics Reference card.*  
<http://www.itp.phys.ethz.ch/econophysics/R/docs/DocRefcard.pdf>.

## Inhaltsverzeichnis

1. **Besonderheiten** & Bibliotheken
2. **R-Commander**
3. Objekte, Klassen, Operatoren und **Funktionen**
4. Import und Export von Daten
5. **Grafiken** in R
6. **Verteilungen** in R

7. Das lineare Regressionsmodell in R
8. Erstellung eigener Funktionen
9. Hilfe
10. R-Metrics
11. Zeitreihenanalyse mit R
12. Multivariate Verfahren mit R
13. Symbolisches Ableiten

## Teil 1: Besonderheiten & Bibliotheken



## Command-Window

Nach dem Programmstart ...

```
RGui - [R Console]
R Datei Bearbeiten Verschiedenes Pakete Windows Hilfe
[Icons]

R : Copyright 2005, The R Foundation for Statistical Computing
Version 2.2.1 (2005-12-20 r36812)
ISBN 3-900051-07-0

R ist freie Software und kommt OHNE JEGLICHE GARANTIE.
Sie sind eingeladen, es unter bestimmten Bedingungen weiter zu verbreiten.
Tippen Sie 'license()' or 'licence()' für Details dazu.

R ist ein Gemeinschaftsprojekt mit vielen Beitragenden.
Tippen Sie 'contributors()' für mehr Information und 'citation()',
um zu erfahren, wie R oder R packages in Publikationen zitiert werden können.

Tippen Sie 'demo()' für einige Demos, 'help()' für on-line Hilfe, oder
'help.start()' für eine HTML Browserschnittstelle zur Hilfe.
Tippen Sie 'q()', um R zu verlassen.

> |
```

## Besonderheiten

- **R** unterscheidet zwischen Groß- und Kleinbuchstaben.
- Variablenamen "c" bzw. "t" sind schon belegt.
- Nachkommastellen werden durch Dezimalpunkt getrennt.
- Eingabebereitschaft wird durch ">" signalisiert.
- Unvollständige Eingabe wird durch "+" signalisiert.
- Objekte werden nach Beenden von **R** nicht automatisch gespeichert.
- Befehlshistorie erhält man mittels "↑"-Taste.

## Bibliotheken

- Ein "gewisser Vorrat" an Funktionen, Daten etc. ist automatisch nach dem Start verfügbar.
- Speziellere Funktionen sind in den unzähligen Zusatzpaketen (sog. "packages") verfügbar (<http://stat.ethz.ch/CRAN/index.html>).
- Das Anzeigen der in der aktiven Sitzung bereits eingebundenen Pakete erfolgt mittels `search()`.
- Potenzielle am Rechner zur Einbindung zur Verfügung stehende Pakete können mittels `library()` angezeigt werden.
- Nicht vorhandene Zusatz-Packages müssen zuerst installiert werden. Danach können sie durch den Befehl `library(package.name)` eingebunden werden, Information über das Paket dann durch `library(help=package.name)`.

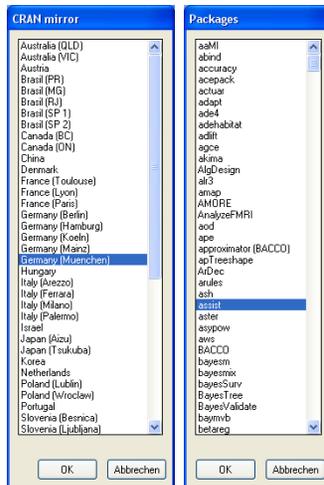
## Bibliotheken

Name	Kurzbeschreibung
evd	Extremwertstatistik
lmtest	Diverse Testverfahren
matrix	Matrizen-Operationen
panel	Analyse von Paneldaten
Rcmdr	Grafische Oberfläche für R
uroot	Einheitswurzeltests
tseries	Zeitreihenanalyse
VaR	Value-at-Risk Berechnung

Table: Ausgewählte R-Bibliotheken

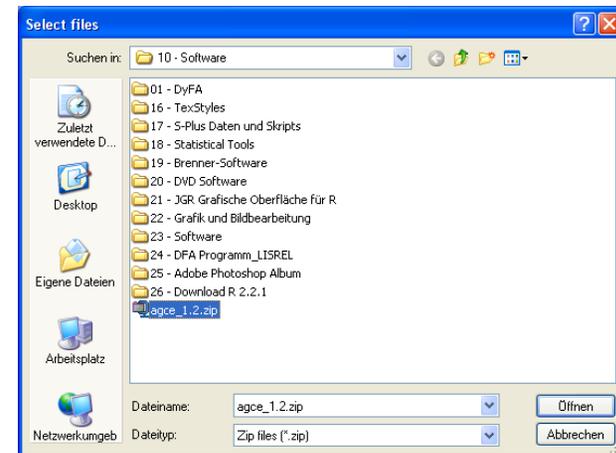
## Installation von Bibliotheken (via Internet)

Menüleiste *Pakete* → *Installiere Paket(e)...*

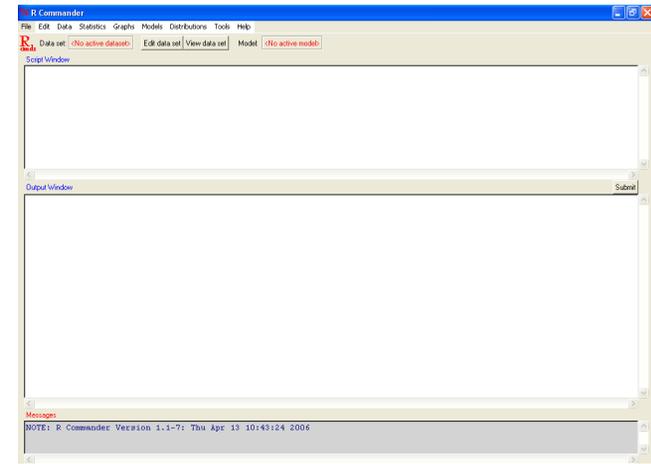


## Installation von Bibliotheken (via CD)

Menüleiste *Pakete* → *Installiere Paket(e) aus lokalen ZIP-Datei...*



## Grafische Benutzeroberfläche von R



## Teil 2: Der R-Commander



## Installation des R-Commanders

- 1 Starten Sie **R**. Installieren Sie die Pakete *Rcmdr* ebenso wie *zoo*, *strucchange*, *sandwich*, *relimp*, *mvtnorm*, *multcomp*, *lmttest*, *effects*, *car* und *abind*.
- 2 Menüleiste "Bearbeiten" → "GUI Einstellungen...".
- 3 Anklicken der Option "SDI".
- 4 Anklicken von "Speichern" (2x) und "OK" (2x).
- 5 Verlassen Sie **R** und starten Sie **R** neu.
- 6 Geben Sie dann `library(Rcmdr)` ein. Die Oberfläche wird dann automatisch geladen.

## Teil 3a: Objekte, Datentypen, Operatoren



## Datentypen in R

- **Numerische Zahlen** ("numeric")  
→ 1.7,  $\pi$  (Pi), +Inf, -Inf ( $\pm\infty$ ), NaN ("not a number")  
`numeric(length = 0), as.numeric(x), is.numeric(x)`
- **Komplexe Zahlen** ("complex")  
→  $1 + 2i$   
`complex(length.out = 0, real = numeric(), imaginary = numeric()),  
as.complex(x), is.complex(x), Re(x), Im(x), Mod(x)`
- **Zeichenketten** ("string")  
→ "Hallo &7"  
`character(length = 0), as.character(x), is.character(x)`
- **Wahrheitswerte** ("logical")  
→ TRUE bzw. T ("wahr"), FALSE/F ("falsch"), NA ("not available")  
`logical(length = 0), as.logical(x), is.logical(x)`

Navigationssymbole

## Objekte in R

- **Vektor**: Anordnung gleichartiger Elemente (z.B. Zahlen)  
`vector(mode = "logical", length = 0)  
as.vector(x, mode = "any") bzw. is.vector(x, mode = "any")`
- **Faktor**: Kategoriale Daten  
`factor(x, levels = sort(unique.default(x), na.last = TRUE), labels = levels)  
is.factor(x) bzw. as.factor(x)`
- **Matrix**: Anordnung gleichartiger Elemente in Zeilen und Spalten  
`matrix(data = NA, nrow = 1, ncol = 1, byrow = FALSE, dimnames = NULL)  
as.matrix(x) bzw. is.matrix(x)`
- **Array**: "Matrix für höhere Dimensionen"  
`array(data = NA, dim = length(data), dimnames = NULL)  
as.array(x) bzw. is.array(x)`
- **Liste**: Anordnung evtl. unterschiedlicher Objekte  
`list(...)` bzw. `as.list(x, ...)` bzw. `is.list(x)`

Navigationssymbole

## Objekte in R

- **Datenframe**: Anordnung unterschiedlicher Elemente gleicher Länge  
`data.frame(...)` bzw. `is.data.frame` bzw. `as.data.frame`
- **Zeitreihe**: zeitliche Anordnung von Beobachtungen einer Variablen (in Vektor) bzw. mehrerer Variablen (in Matrix) sowie Informationen  
`ts(data = NA, start = 1, end = numeric(0), frequency = 1)  
as.ts(x, ...) bzw. is.ts(x)`
- **Ausdrücke** ("expression"): Sinnvolle Ansammlung von Zeichen  
`expression(...)` bzw. `as.expression(x, ...)` bzw. `is.expression(x)`
- **Funktion** ("function"): Sinnvolle Ansammlung von Zeichen  
`function(x, ...)` bzw. `as.function(x, ...)` bzw. `is.function(x)`
- **Formel** ("formula"): Sinnvolle Ansammlung von Zeichen  
`formula(x, ...)` bzw. `as.formula(x, ...)`

Navigationssymbole

## Attribute von Objekten in R

→ "spezifizieren die Art der Daten näher"

- **Datentyp**: `mode(x)`
- **Länge**: `length(x)`
- **Klasse**: `class(x)`

Navigationssymbole

## Variablen in R

- Der Begriff einer Variablen in **R** ist sehr flexibel gefasst: Man versteht darunter allgemein ein mit einem Namen ("Variablennamen") ansprechbares Objekt.
- Variablen werden durch ihre erste Wertzuweisung mittels "`<-`" oder "`=`" initialisiert: `a<-3` bzw. `a=3`.
- **R** unterscheidet zwischen Klein- und Großbuchstaben bei Variablennamen: `aaa=3` ungleich `AAA=3`.
- Achtung: **R** überschreibt Variablen ohne Nachfrage.

## Objektverwaltung in R

- Anzeigen aller Objekte im aktuellen Workspace mittels `objects()` bzw. synonym `ls()`.
- Anzeigen aller Objekte eines Pakets XX mittels `objects("package:XX")`.
- Anzeigen aller Objekte im aktuellen Workspace mittels die ein "b" im Namen enthalten mittels `ls(pat="b")`.
- Löschen eines Objekte X mittels `rm(X)` bzw. `remove(X)`.
- Löschen aller Objekte im aktuellen Workspace mittels `rm(list=ls())`.

## [Mathematische] Operatoren in R

- Verknüpfung von Zahlen, Konstanten, Vektoren bzw. Matrizen.
- Ergebnis ist wieder Zahl, Konstante, Vektor bzw. Matrix.
- Ist Verknüpfung nicht möglich, wird NaN zurückgeliefert.

Operator	Operation	Beispiel
+	Addition	1+3
-	Subtraktion	1-3
*	Multiplikation	1*3
^	Potenzierung	1^3
/	Division	4/3
%%	Divisionsrest	4%%3
%/%	Ganzzahlige Division (ohne Rest)	4%/%3

Table: (R Operator.R)

## Vergleichsoperatoren in R

- (Komponentenweise) Vergleich zweier Objekte.
- Ergebnis ist ein (Vektor) boolescher Wert(e), d.h. TRUE oder FALSE.

Operator	Operation	Beispiel (a=2, b=1)
>	größer als	3>5 → FALSE
<	kleiner als	4<4 → FALSE
==	genau gleich	a==4 → FALSE
!=	ungleich	b!=5 → TRUE
>=	größer gleich	a>=b → TRUE
<=	größer gleich	a<=3 → TRUE

Table: (R Operator.R)

## Logische bzw. Boolesche Operatoren in R

- Ergebnis ist ein (Vektor) boolescher Wert(e), d.h. TRUE oder FALSE.
- 0 hat den Wert FALSE, alle anderen Zahlen den Wert TRUE.

Operator	Operation	Beispiel (a=2, b=0)
!	nicht	!b → TRUE
&	und	a&b → FALSE
	oder	a b → TRUE
xor	oder (ausschließlich)	xor(a,b) → TRUE
isTRUE	Prüfung auf WAHR	isTRUE(a) → FALSE

Table: (R Operator.R)

## Teil 3b: Elementare Funktionen



## Elementare mathematische Funktionen

Funktion	Beschreibung
sqrt(x)	Wurzelfunktion $f(x) = \sqrt{x}$
abs(x)	Betragsfunktion $f(x) =  x $
exp(x)	Exponentialfunktion $f(x) = e^x$
log(x, base = exp(1))	Logarithmus
log10(x)/log2(x)	Logarithmus
sin(x)/cos(x)/tan(x)	Sinus/Kosinus/Tangens (SKT)
asin(x)/acos(x)/atan(x)	Arcus SKT
sinh(x)/cosh(x)/tanh(x)	SKT Hyperbolicus
asinh(x)/acosh(x)/atanh(x)	Arcus SKT Hyperbolicus
gamma(x)/lgamma(x)	(logarithmierte) Gamma-Funktion
psigamma(x, deriv=0)	Psigamma-Funktion
digamma(x)	Digamma-Funktion
beta(a,b)/lbeta(a,b)	(logarithmierte) Beta-Funktion
besselI(x, nu)/besselJ(x, nu)	Modifizierte Bessel-Funktion
besselK(x, nu)/besselY(x, nu)	Modifizierte Bessel-Funktion

Table: Mathematische Funktionen (R Mathe.R).

## Elementare Funktionen

R-Funktion	Beschreibung
choose(n, k)	Binomialkoeffizient $\binom{n}{k}$
lchoose(n, k)	$\log \binom{n}{k}$
factorial(x)	Fakultät $n!$
lfactorial(x)	$\log n!$

Table: Mathematische Funktionen (R Mathe.R).

R-Funktion	Beschreibung
round(x, digits=0)	Mathematische Rundung
signif(x, digits=0)	Mathematische Rundung
zapsmall(x)	Mathematische Rundung
ceiling(x)	Nächstgrößere ganze Zahl
floor(x)	Nächstkleinere ganze Zahl
trunc(x)	Abschneiden der Nachkommastellen

Table: Funktionen zur Rundung von Zahlen (R Rundung.R).

## Erzeugung von Vektoren

- Mittels des **Combine-Befehls** `c`:  
`x=c(1,4,3,5)` oder `x=c(1,2,3,4)`
- Mittels des **:-Operators**:  
`x=1:4` oder `x=10:1`
- Mittels des **sequence-Befehls**:  
`seq(-2,4,2)` oder `seq(-2,4,,2)`
- Mittels des **rep-Befehls**:  
`rep(1,10)` oder `rep(c(1,3),c(2,4))`
- Durch **Kombination der obigen Befehle**:  
`c(1:10,rep(2,3),seq(0,1,,200))`

## Selektion von Elementen eines Vektors

R-Befehl	Beschreibung
<code>x[3]</code>	Element 3
<code>x[c(1,3)]</code>	Element 1 und 3
<code>x[3:5]</code>	drittes bis fünftes Element
<code>x[-3]</code>	Vektor ohne drittes Element
<code>x[-c(1,3)]</code>	Vektor ohne erste und drittes Element
<code>x[x&gt;3]</code>	Alle Elemente größer als 3
<code>x[x&lt;1 x&gt;3]</code>	Alle Elemente größer als 3 oder kleiner als 1
<code>x[x&lt;1&amp;x&gt;3]</code>	Alle Elemente größer als 3 und kleiner als 1
R-Funktion	Beschreibung
<code>any(x&gt;0)</code>	TRUE, falls ein Element positiv, sonst FALSE
<code>all(x&gt;0)</code>	TRUE, falls alle Element positiv, sonst FALSE
<code>which(x&gt;0)</code>	Indizes der Elemente, die größer als 0 sind
<code>append(x,values,after)</code>	Verketten von Vektoren

Table: Selektion einzelner Elemente (R Vektor.R).

## Funktionen für Vektoren I

R-Funktion	Beschreibung
<code>length(x)</code>	Länge des Vektors
<code>sum(x)</code>	Summe der Elemente
<code>mean(x)/median(x)</code>	Mittelwert/Median der Elemente
<code>var(x)</code>	Varianz der Elemente mit $1/(n-1)$
<code>cumsum(x)</code>	Laufende Summe
<code>prod(x)</code>	Produkt der Elemente
<code>cumprod(x)</code>	Laufendes Produkt
<code>max(x)</code>	Größtes Element
<code>cummax(x)</code>	Laufendes Maximum
<code>min(x)/cummin(x)</code>	Minimum/Laufendes Minimum
<code>range(x)</code>	Minimum und Maximum
<code>sort(x,decreasing = FALSE)</code>	Sortierung der Elemente
<code>rank(x)</code>	Rangbildung
<code>rev(x)</code>	Vektor in umgekehrter Reihenfolge
<code>unique(x)</code>	Vektor "ohne Ties"

Table: Funktionen zur Manipulation von Vektoren (R Vektor.R).

## Funktionen für Vektoren II

R-Funktion	Beschreibung
<code>diff(x)</code>	Vektor der ersten Differenzen
<code>sgn(x)</code>	Vektor der Vorzeichen
<code>which.max(x)</code>	Index des Maximums
<code>which.min(x)</code>	Index des Minimums
<code>is.vector(x)</code>	TRUE falls Objekt x ein Vektor
<code>na.omit(x)</code>	Vektor ohne NA's
<code>na.fail(x)</code>	Fehlermeldung falls NA in Vektor

Table: Funktionen zur Manipulation von Vektoren (R Vektor.R).

## Erzeugung von Matrizen

- Erzeugung einer Matrix mit dem **R**-Befehl `matrix`  
`elemente=c(1,1,1,2,3,4,6,7,3)` # Eingabe als Vektor!  
`matrix(elemente,ncol=3,nrow=3,byrow=FALSE)`  
 Spaltenweise Anordnung der Elemente in drei Spalten und drei Zeilen.
- Diagonal-Matrizen:** `diag(a)` bzw. `diag(c(1,2,3))`
- Operatoren für Matrizen

Operator	Operation
<code>+</code> , <code>-</code>	Elementweise Addition/Subtraktion
<code>*</code> , <code>/</code>	Elementweise Multiplikation/Division
<code>%*%</code>	Matrix-Multiplikation
<code>%x%</code>	Kroneckerprodukt, vgl. auch <code>kron</code>
<code>%o%</code>	Äußeres Produkt, vgl. auch <code>outer</code>
<code>t(x)</code>	Transponierung
<code>solve(x)</code>	Invertierung

Table: (R Matrix.R)

## Selektion von Elementen einer Matrix

R-Befehl	Beschreibung
<code>A[1,3]</code>	Element in Zeile 1 und Spalte 3
<code>A[1,]</code>	erste Zeile
<code>A[,2]</code>	zweite Spalte
<code>A[c(1,4),]</code>	erste und vierte Zeile
<code>A[-c(1,4),]</code>	Matrix ohne erste und vierte Zeile
R-Funktion	Beschreibung
<code>crossprod(A,B)</code>	Berechne $A'A$ bzw. $A'B$
<code>colSums(A)</code>	Spaltensumme der Matrix A
<code>rowSums(A)</code>	Zeilensumme der Matrix A
<code>colnames(A)</code>	Spaltennamen übergeben
<code>rownames(A)</code>	Zeilenamen übergeben
<code>rbind(A,B)/cbind(A,B)</code>	Verbinde Matrizen "zeilenweise" / "spaltenweise"
<code>lower.tri/upper.tri(A)</code>	TRUE/FALSE falls im unteren/oberen Bereich
<code>apply(A,1,mean)</code>	Mittelwert über alle Zeilen
<code>apply(A,2,mean)</code>	Mittelwert über alle Spalten

Table: Selektion einzelner Elemente (R Matrix.R)

## Funktionen für Matrizen

R-Funktion	Beschreibung
<code>ncol(x)</code>	Anzahl der Spalten
<code>nrow(x)</code>	Anzahl der Zeilen
<code>length(x)</code>	Anzahl der Elemente
<code>dim(x)</code>	Dimension
<code>diag(x)</code>	Hauptdiagonale
<code>diag(a)</code>	Einheitsmatrix ( $a \in \mathbb{N}$ )
<code>solve(x)</code>	Inverse Matrix
<code>chol2inv(x)</code>	Inverse Matrix (mittels Cholesky)
<code>det(x)</code>	Determinante der Matrix
<code>eigen(x)</code>	Eigenwerte/vektoren der Matrix
<code>t(x)</code>	transponierte Matrix
<code>qr(x)</code>	QR-Zerlegung
<code>svd(x)</code>	Singulärwert-Zerlegung
<code>chol(x)</code>	Cholesky-Zerlegung

Table: Funktionen zur Manipulation von Matrizen (R Matrix.R)

## Funktionen für Zeichenketten (Strings)

Funktion	Beschreibung
<code>cat</code>	Ausgabe von Objekten
<code>paste</code>	Fusion von Zeichenketten
<code>nchar</code>	Anzahl der Zeichen
<code>charmatch</code>	Matchen von Zeichenketten
<code>substr(x, start, stop)</code>	Teile einer Zeichenkette

Table: Funktionen für Zeichenketten (R String.R)

## Teil 4: Datenmanagement: Eingabe, Austausch, Import & Export



- Geringe Datenmengen können über Tastatur mittels `scan` eingegeben werden, z.B. `datenneu=scan()`.

Ein Leerzeichen oder das Betätigen der *Return*-Taste markieren einen neuen Datenpunkt. Der "Eingabemodus" wird durch zweimaliges Betätigen der *Return*-Taste verlassen.

- Ist ein Datensatz bzw. sind mehrere Datensätze bereits definiert, so können diese bequem mittels `data.entry` modifiziert bzw. erweitert werden.

```
> x=1:10  
> y=2:11  
> data.entry(x,y)
```

## Dateneingabe über Tastatur

### Der Daten-Editor

	x	y	var3	var4	var5	var6	var7
1	1	2					
2	2	3					
3	3	4					
4	4	5					
5	5	6					
6	6	7					
7	7	8					
8	8	9					
9	9	10					
10	10	11					
11							
12							
13							
14							
15							
16							
17							
18							
19							

## Einlesen von Daten aus externen Dateien

- Datenimport mittels `scan` (für gleiche Datentypen!)

```
scan(file = "", what = double(0), sep = "", dec = ".", skip = 0)
```

Es können u.a. folgende Spezifikation festgelegt werden: Pfad inkl. Dateinamen (*file*), Dateityp (*what*), Trennzeichen zwischen den Datenpunkten (*sep*), Trennzeichen für Dezimalstellen (*dec*) und Anzahl der Zeilen, die übersprungen werden sollen (*skip*), z.B. weil Überschriften enthalten sind.

→ erzeugt einen Vektor in **R**.

- *Beispiel:* `bmw=scan("d:/rkurs/bmw.txt",skip=1).`

## Einlesen von Daten aus externen Dateien

- Datenimport mittels `read.table` (für unterschiedliche Datentypen!)

```
read.table(file,header = FALSE,sep = ",",dec = ".")
```

Es können u.a. folgende Spezifikation festgelegt werden: Pfad inkl. Dateinamen (*file*), Trennzeichen zwischen den Datenpunkten (*sep*), Trennzeichen für Dezimalstellen (*dec*) und das Vorhandensein einer Überschrift (*header*). Weitere Optionen: Spaltenüberschriften (*col.names*) bzw. Zeilenüberschriften (*row.names*).

→ erzeugt einen `Data.frame` in **R**.

- *Beispiel*: `kdat=read.table("d:/rkurs/kontodaten.dat",TRUE)`.

## Einlesen von Daten aus externen Dateien

- **Modifikationen** des `read.table`-Befehls (library "*foreign*")

- 1 `read.csv`: Einlesen von CSV-Dateien (amerikanische Version)
- 2 `read.csv2`: Einlesen von CSV-Dateien (europäische Version)
- 3 `read.spss`: Einlesen von SPSS-Dateien
- 4 `read.dta`: Einlesen von STATA-Dateien
- 5 `read.ssd`: Einlesen von SAS-Dateien
- 6 `read.octave`: Einlesen von Octave/Matlab-Dateien

Analoge Funktionen stehen zum Auslesen zur Verfügung.

## Auslesen von Daten in externen Dateien

- mittels `write.table`
- mittels `cat`
- mittels `write`
- mittels `sink`
- mittels `save`

## Austausch von R-Objekten

- **Austausch einzelner R-Objekte** mittels `dump` und `source`.

```
> x=1:10; y=c("Marco","Ruth")
> dump(c("x","y"),file="d:/rkurs/austausch.R")
> rm(x,y) # Loescht die Objekte x und y
> source("d:/rkurs/austausch.R")
> ls()
```

- **Austausch aller R-Objekte** mittels `save.image` und `load`.

```
> save.image(file="d:/rkurs/14042006.RData")
> rm(list=ls()) # Loescht alle Objekte im Verzeichnis
> load(file="d:/rkurs/14042006.RData")
> ls()
```

## Einbinden von Datensätzen aus R-Bibliotheken

- Ein "gewisser Vorrat" an Datensätzen ist automatisch nach dem Start verfügbar. Die können mit `data()` angezeigt werden.
- Datensätze, die in einem Paket `pname` zur Verfügung stehen, können mit `data(package="pname")` erfragt werden.
- Ein spezieller Datensatz `dname` kann mittels `data("dname")` eingebunden werden.
- Weitere nützliche Befehle sind außerdem:  
`names(dname)`  
`attach(dname)`  
`detach(dname)`

## Teil 5: Grafiken



## Der Plot-Befehl

- R stellt umfangreiche Grafikmöglichkeiten zur Verfügung, starte z.B. `demo(graphics)`.
- Der wichtigste Befehl zur Erzeugung von Grafiken ist `plot`.  

```
> x=seq(-4,4,,200)
> y=x^2
> windows() # Nicht unbedingt notwendig
> plot(x,y)
```
- Grafiken werden gemäß fest definierter Grafikparameter erzeugt. Diese können mittels `par` verändert werden.
- Weitere Packages: `grid`, `lattice`, `xgr`

## Der Plot-Befehl

→ Ausgewählte Optionen des Plot-Befehls: `plot(x,y,option)`.

option	Beschreibung	Beispiel
add	Überlagerung der letzten Plots	add=TRUE
axes	Anzeigen der Achsen	axes=FALSE
col	Farbe	col=2
type	Art des Plots	col="l"/"p"/"b"/"o"/"h"/"s"
lwd	Linienstärke	lwd=2
main	Überschrift	main="Histogramm"
sub	Unterschrift	sub="Werte"
xlab	Titel für x-Achse	xlab="x"
ylab	Titel für y-Achse	ylab="f(x)"
xlim	Ober-/Untergrenze für x-Achse	xlim=c(0,2)
ylim	Ober-/Untergrenze für y-Achse	ylim=range(x)
pch	Symbol für Punktgröße	pch=1.2
cex	Symbolgröße	cex=1.2

Table: Optionen des Plot-Befehls (R Grafik.R)

## Grafik-Parameter

→ Ausgewählte Grafikparameter: `par(parameter)`.

parameter	Beschreibung	Beispiel
adj	Textanordnung (linksbündig=0)	adj=0
bg	Farbe des Hintergrunds	bg="yellow"
bty	Kontrolle der Umrandung des Plots	bty="o"
cex.xx	Text und Symbolgröße	cex.lab=1.3
col.xx	Farbe der Schrift etc.	col.axis=2
font.xx	Schriftart	font.main=2
las	Ausrichtung der Achsenbeschriftung	las=1
lty	Kontrolle des Linientyps	lty=1
lwd	Kontrolle der Linienstärke	lwd=2
mar	Abstände zwischen Rand und Plot	las=c(5,4,4,2)
mfrow	Aufteilung in Subfenster	mfrow=c(2,3)
pch	Plot-Symbol	pch=8
tck	Länge der Tickmarks	tck=-0.5

Table: Grafik-Parameter (R Grafik.R)

## Plot-Befehl

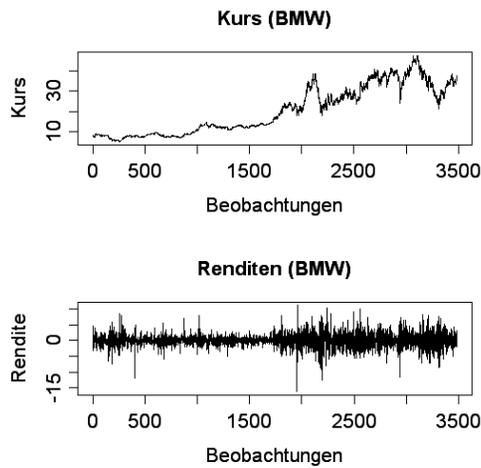


Figure: Plot (R Grafik.R)

## High-level-Grafikfunktionen

→ zur Darstellung **eindimensionaler Daten**.

R-Befehl	Beschreibung
plot	Indexplot, Stabdiagramm, empirische CDF
curve	Erzeugung mathematischer Kurven
boxplot	Boxplot
barplot	Säulendiagramm
dotchart	Cleveland-Dot-Chart
hist	Histogramm
pie	Kreisdiagramm
qqnorm	Quantil-Plot für Normalverteilung
qqplot	Quantil-Quantil-Plot
stripchart	Eindimensionales Streudiagramm
stem	Stem&Leaf-Diagramm

Table: Befehle zur Erstellung von Grafiken (R Grafik.R)

## Histogramm

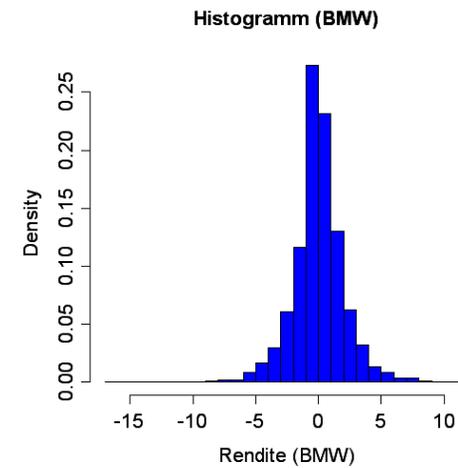


Figure: Histogramm (R Grafik.R)

## Boxplot

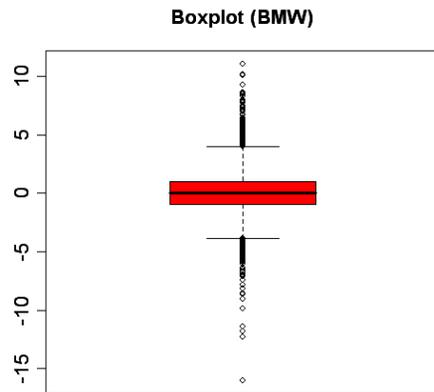


Figure: Boxplot (R Grafik.R)

## Tortendiagramm

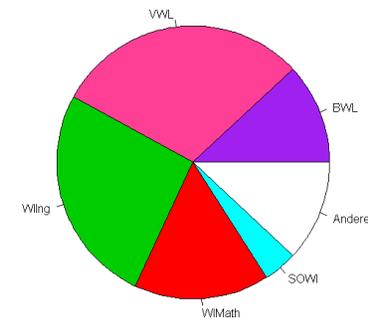


Figure: Tortendiagramm (R Grafik.R)

## Balkendiagramm

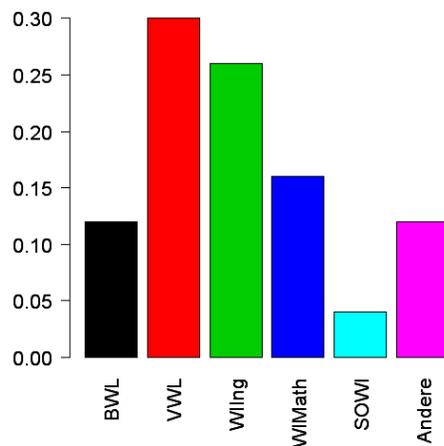


Figure: Barplot (R Grafik.R)

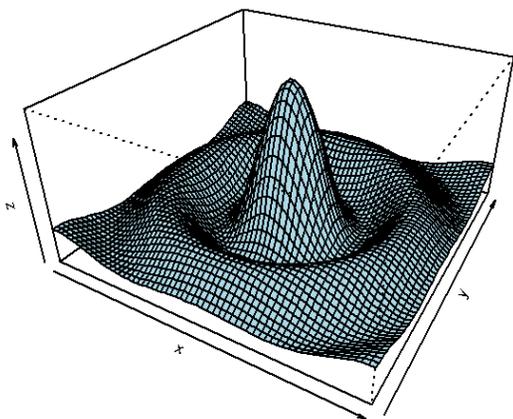
## High-level-Grafikfunktionen

→ zur Darstellung **höherdimensionaler Daten**.

R-Befehl	Beschreibung
plot	Streudiagramme bzw. Linienzüge
matplot	Darstellung der Spalten einer Matrix gegen die anderen Spalten
stars	Sternendiagramm multivariater Daten
persp	3-D Plot
contour	Plot der Höhenlinien
coplot	Bedingte Streudiagramme
image	Image Plots
pairs	Streudiagramm-Matrix

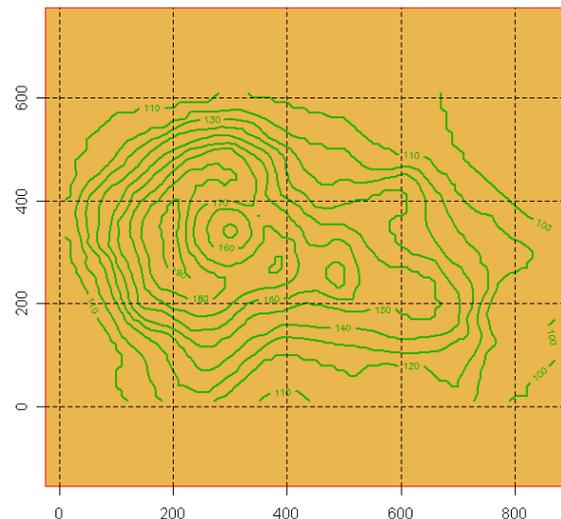
Table: Befehle zur Erstellung von Grafiken (R Grafik.R)

## 3-D Plot



## Contour-Plot

Topografische Landkarte des Maunga Whau



## Low-level-Grafikfunktionen

→ erzeugen keine Grafiken sondern verändern vorhandene Grafiken nur.

R-Befehl	Beschreibung
<code>abline</code>	Hinzufügen einer Gerade in vorhandene Grafik
<code>legend</code>	Hinzufügen einer Legende in vorhandene Grafik ein
<code>lines</code>	Hinzufügen einer Linie in vorhandene Grafik
<code>points</code>	Hinzufügen einer Punkte in vorhandene Grafik
<code>polygon</code>	Hinzufügen eines Polygonzuges, auch ausgefüllt
<code>rug</code>	Hinzufügen eines ergänzenden Stabdiagramms
<code>segment</code>	Hinzufügen von Linien-Segmenten zwischen Punktenpaaren
<code>text</code>	Hinzufügen von Text an beliebigen Stellen in einer Grafiken
<code>mtext</code>	Hinzufügen von Text im Randbereich
<code>title</code>	Hinzufügen einer Überschrift

Table: Befehle zur Modifikation von Grafiken (R Grafik.R)

## Ergänzungen zu Grafiken

- Befehle `device`, `dev.list`, `dev.cur`, `dev.off`
- Partitionierung einer Grafik mit `split.screen`, `erase.screen`, `screen`
- Partitionierung einer Grafik mit `layout`, `layout.show`
  - > `m=matrix(1:4,2,2)`
  - > `layout(m,width=c(1,3),heights=c(3,1))`
  - > `layout.show(4)`
  - > `test=rnorm(100)`
  - > `hist(test,col=2)`
  - > `hist(test,col=3)`
  - > `hist(test,col=4)`
  - > `hist(test,col=5)`

## Partitionierung einer Grafik

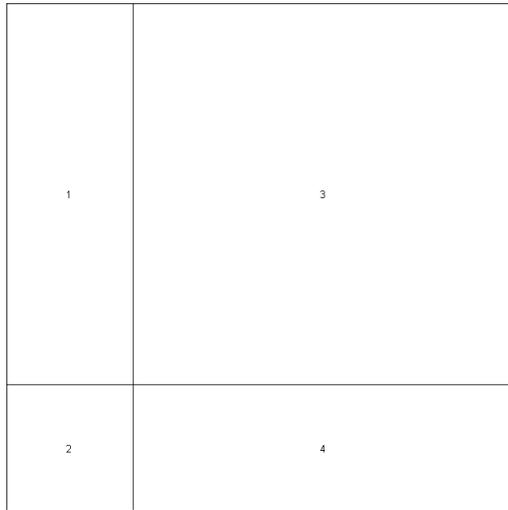


Figure: Partitionierung (R Grafik.R)

## Partitionierung einer Grafik

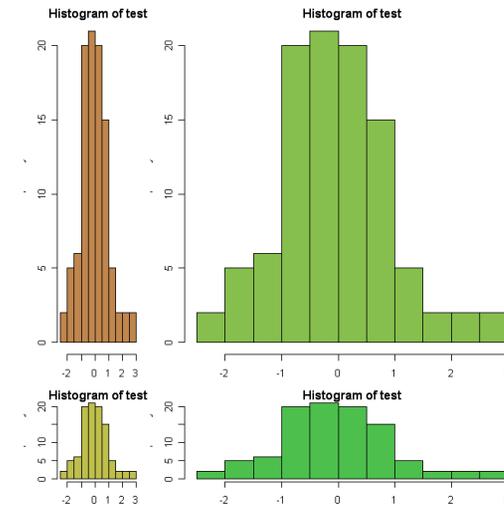


Figure: Partitionierung (R Grafik.R)

## Teil 6: Verteilungen & Zufallszahlen



## Ziehung von Stichproben mit `sample`

- Ziehung von Stichproben mittels `sample`, Syntax:

```
sample(x, size, replace = FALSE, prob = NULL)
```

**Erklärung:** `x` enthält die Objekte, aus denen Stichproben gezogen werden sollen, `size` gibt die Anzahl der Stichproben an, `replace=TRUE` entspricht einer Ziehung mit Zurücklegen und mittels `prob` kann eine Gewichtung der Objekte festgelegt werden.

- *Beispiel:* `sample(1:3, 10, TRUE)`.
- Weitere Beispiele im R-Skript `R Verteilung.R`.

## Verteilungen

- R bieten Funktionen für **Dichten** (density function), **Verteilungsfunktionen** (probability function) und der **Quantile** (quantile function) von diversen Verteilungen sowie zur Erzeugung von **Zufallszahlen** (random number) aus diversen Verteilungen.
- Jede Funktion hat zwei Bestandteile  
**Funktionskürzel** (d,p,q,r) + **Verteilungskürzel** (norm,t,f,...)
- *Beispiele:*

Funktion	Beschreibung
<code>dnorm(1,0,2)</code>	Normaldichte ( $\mu = 0, \sigma = 2$ ) an der Stelle $x = 0$
<code>rnorm(10)</code>	10 Zufallszahlen aus Standard-Normalverteilung
<code>pt(0.5,1)</code>	t(1)-Verteilungsfunktion an Stelle $x = 0.5$
<code>qbinom(0.9,10,0.5)</code>	90%-Quantile (Binomialvert.) mit $n = 10, p = 0.5$

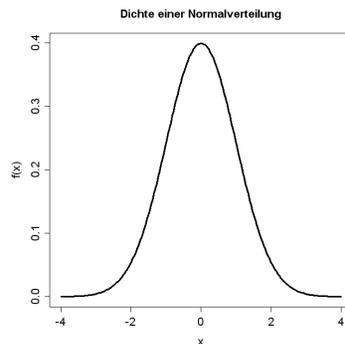
Table: Funktionen zu Verteilungen (R **Verteilung.R**)

## Verteilungen in R (Auszug)

Kürzel	Verteilung	Paket
<code>exp</code>	Exponential-Verteilung	
<code>norm</code>	Normalverteilung	
<code>t</code>	Student-t-Verteilung	
<code>f</code>	F-Verteilung	
<code>chisq</code>	$\chi^2$ -Verteilung	
<code>logis</code>	Logistische Verteilung	
<code>binom</code>	Binomialverteilung	
<code>pois</code>	Poissonverteilung	
<code>lnorm</code>	Lognormalverteilung	
<code>symstb</code>	Symmetrische Stabile Verteilung	<i>fBasics</i>
<code>stable</code>	Schiefe Stabile Verteilung	<i>fBasics</i>
<code>gh</code>	Verallg. hyperbolische Verteilungen	<i>fBasics</i>
<code>hyp</code>	Hyperbolische Verteilung	<i>fBasics</i>
<code>nig</code>	Normal inverse Gauss Verteilung	<i>fBasics</i>

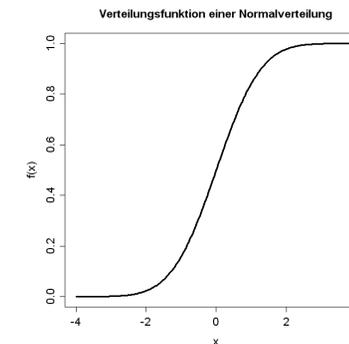
Table: Ausgewählte Verteilungen in R

## Verteilungen in R

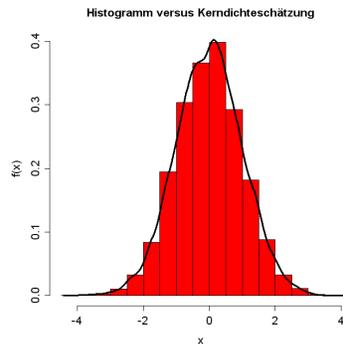


```
# Dichtefunktionsplot einer Standard-Normalverteilung
x=seq(-4,4,,200)
par(cex.axis=1.3,cex.lab=1.3,cex.main=1.3)
titel="Dichte einer Normalverteilung"
plot(x,dnorm(x),type="l",main=titel,xlab="x",ylab="f(x)",lwd=3)
```

## Verteilungen in R



```
# Plot der Verteilungsfunktion einer Standard-Normalverteilung
x=seq(-4,4,,200)
par(cex.axis=1.3,cex.lab=1.3,cex.main=1.3)
titel="Verteilungsfunktion einer Normalverteilung"
plot(x,pnorm(x),type="l",main=titel,xlab="x",ylab="f(x)",lwd=3)
```



```
# KDE/Histogramm fuer normalverteilte Zufallszahlen
par(cex.axis=1.3,cex.lab=1.3,cex.main=1.3)
zufallszahlen=rnorm(10000)
titel="Histogramm versus Kerndichteschaetzung"
hist(zufallszahlen,prob=T,col=2,xlab="x",ylab="f(x)",main=titel)
lines(density(zufallszahlen),lwd=3)
```

- Schätzung der Dichtefunktion mittels `density`

```
density(x,bw = "nrd0",adjust = 1,kernel = "gaussian")
```

- Schätzung der Verteilungsfunktion `ecdf`

```
ecdf(x)
```

- Schätzung der Quantilsfunktion `quantile`

```
quantile(x,probs = seq(0,1,0.25))
```

- Beispiele im R-Skript  **Verteilung.R**.

## Teil 7: Lineare Regression



## Einfache lineare Regression

Geschätzt werden soll das nachfolgende einfache, lineare Regressionsmodell

$$Y_t = \alpha + \beta X_t + U_t, \quad t = 1, \dots, T.$$

- Lineare Regression mittels `lm` ("linear model") und `summary`
- Lineare Regression mittels `lsfit` "veraltet"

*Beispiel:*

```
> data(cars);attach(cars);plot(speed,dist)
> lm.ergeb=lm(speed~dist)
> summary(lm.ergeb)
```

## Einfache lineare Regression

- Zusammenfassung der Regressionsergebnisse: `summary(reg.ergeb)`
- Geschätzte Koeffizienten mittels `coeff(reg.ergeb)`
- Prognose mittels `predict(reg.ergeb)`
- Grafische Analyse mittels `plot(reg.ergeb)`
- Ausreißeranalyse mittels `influence.measure`
- Ausreißeridentifikation mittels `identify`

## Exkurs: Testverfahren

Funktion	Operation	Package
Testverfahren auf Normalverteilung		
<code>jarque.bera.test</code>	Jarque-Bera Test (1980)	<code>tseries</code>
<code>shapiro.test</code>	Shapiro/Wilk/Royston Test (1982)	<code>stats</code>
<code>ks.test</code>	Kolmogorov-Smirnov Test (1950)	<code>stats</code>
<code>chisq.test</code>	$\chi^2$ -Anpassungstest	<code>stats</code>
<code>t3plot</code>	$T_3$ -Plot von Gosh (1996)	Homepage
<code>qqnorm</code>	Quantil-Quantil-Plot	<code>stats</code>
Testverfahren auf Autokorrelation		
<code>dwtest</code>	Durbin-Watson Test (1950)	<code>lmtest</code>
<code>bgtest</code>	Breusch-Godfrey Test (1978)	<code>lmtest</code>

Table: Ausgewählte Testverfahren

## Exkurs: Testverfahren

Funktion	Operation	Package
Testverfahren auf Linearität		
<code>harvtest</code>	Harvey-Collier Test (1977)	<code>lmtest</code>
<code>resettest</code>	RESET Test von Ramsey (1969)	<code>lmtest</code>
Testverfahren auf Heteroskedastie		
<code>bptest</code>	Breusch-Pagan Test (1978)	<code>lmtest</code>
<code>qgtest</code>	Goldfeld-Quandt Test (1965)	<code>lmtest</code>
<code>hmctest</code>	Harrison-McCabe Test (1979)	<code>lmtest</code>
Testverfahren auf Unabhängigkeit		
<code>runsTest</code>	Run Test	<code>fBasics</code>

Table: Ausgewählte Testverfahren

## Teil 8: Entwicklung eigener Funktionen



R Coding Conventions: [www.maths.lth.se/help/R/RCC/](http://www.maths.lth.se/help/R/RCC/)

## Schreiben eigener Funktionen

- Nicht alle gewünschten Funktionen stehen in **R** standardmäßig zur Verfügung. Es können jedoch sehr einfach eigene/individuelle Funktionen (=Objekte) erstellt werden.
- Kleinere Funktionen können direkt im Command-Window erstellt werden, z.B. `summe=function(a,b){a+b}`. Anschließend stehen Sie dem Benutzer zur Verfügung.
- Umfangreichere Funktion sollten in einem Editor (z.B. WinEdt) erstellt werden. Der erstmaligen Aufruf einer Funktion geschieht mit `fix(wuerfelwurf)`. Danach Funktion abspeichern

```
wuerfelwurf - R Editor
Datei Bearbeiten Pakete Hilfe
function (n)
{
# Funktion wuerfelwurf
# Diese simuliert einen n-fachen Würfelwurf
ergebnis=sample(1:6,n,replace=T)
return(ergebnis)
}
```

## Schreiben eigener Funktionen

- Der **grundlegende Aufbau einer R-Funktion**  
`function_name = function (function_argument){  
function_body  
function_return_value  
}`
- Mit dem `function_name` wird die Funktion angesprochen.
- Das Keyword `function` signalisiert **R**, dass es sich um ein Objekt der Klasse "function" handelt.
- Generell können einer Funktion bei Aufruf Informationen im sog. `function_argument` übergeben werden.
- Der `function_body` steht stellvertretend für die **R**-Befehle, die abgearbeitet werden.
- Am Ende wird der `function_return_value` zurückgegeben.

## Kontrollstrukturen: For next-Schleife

```
for (variable in sequence){  
R-Befehle  
}  
next
```

```
wuerfelwurf2 - R Editor
Datei Bearbeiten Pakete Hilfe
function (n)
{
# Funktion wuerfelwurf
# Diese simuliert einen n-fachen Würfelwurf

for(i in 1:n) {
x=sample(1:6,1,replace=T)
cat("Wurf ",i,":", x,"\n")
}
return()
}
```

## Kontrollstrukturen: While-Schleife

```
while (expression){  
R-Befehle  
}  
next
```

```
wuerfelwurf2 - R Editor
Datei Bearbeiten Pakete Hilfe
function ()
{
# Funktion wuerfelwurf2
# Würfelwurf bis zur ersten Sechs

i=0;
x=1;

while (x!=6) {
i=i+1
x=sample(1:6,1,replace=T)
cat("Wurf ",i,":", x,"\n")
}
return()
}
```

```
if (condition){ R-Befehle }  
else{R-Befehle }  
end
```

```
R wuerfelwurf3 - R Editor  
Datei Bearbeiten Pakete Hilfe  
function ()  
{  
# Funktion wuerfelwurf3  
# Würfelspiel  
  
x=sample(1:6,1)  
if(x==6){  
  cat("Gewonnen\n")  
}  
else{  
  cat("Leider verloren\n")  
}  
}
```

## Teil 9: Hilfe in R



## Hilfe in R

- Mit dem Befehl `help.start()` bzw. mittels **Hilfe/HTML Hilfe** aus der Menüleiste wird die Hilfefunktion gestartet.
- Der Aufruf `help(such.begriff)` bzw. `?such.begriff` liefert Hilfe/Dokumentation zu dem (exakten!) Begriff `such.begriff`.
- Mit dem Befehl `apropos(such.begriff)` werden alle Objekte gelistet, die auf den Begriff bzw. auf die Funktion `such.begriff` zurückgreifen.
- Mit dem Befehl `help(library=package.name)` werden alle Objekte der Bibliothek `package.name` angezeigt.