Kurzeinführung in R
(Bernd Huwe, SoSem 2006)

Inhaltsverzeichnis

1 Erstkontakt

1.1 Starten von R

Doppelklick auf das R-Icon startet die Windowsoberfläche

1.2 Verlassen von R

- quit()
- q()
  Es folgt die Abfrage, ob der Workspace gesichert werden soll.
- oder: q("yes")

Bem.:

- R arbeitet mir Objekten und Funktionen bzw. Befehlen. Objekte enthalten Daten und Methoden. Methoden sind Funktionen innerhalb von Objekten. Objekte sind Instanzen von Klassen. Klassen sind Matrizen für Objekte. Sie bestehen aus der Datenstruktur und den Methoden. Klassen sind in objektorientierten Programmiersprachen wie C++, Delphi, Java, u.a. entwickelt.
- q() ist eine Funktion, erkenntlich an den Klammern.
- q() ist eine Kurzschreibweise von quit(). In R müssen nur soviele Buchstaben eines Namens eingegeben werden bis die Zeichenfolge vollständig ist.

1.3 Erläutern der Windows-Oberfläche

Eingabemodus:

- R-Texteditor: ***Datei – Öffne Skript oder Datei – Neues Skrip***t
  Skripte enthalten R-Kommandos (*.R)
- Dateneditor: ***Bearbeiten – Dateneditor***
  bearbeitet Matrix oder data frame
- R-Skripte einlesen: ***Datei – Lese R Code ein***
  der R-Code wird ausgeführt und die Ergebnisse ggf. in der R-Console angezeigt.
  mit ls() erhält man eine Liste der erzeugten Objekte
  mit obj ↵ wird der Inhalt des Objekts angezeigt
  mit str(obj) ↵ wird die Struktur des Objekts angezeigt
  mit rm(obj) wird ein Objekt gezielt aus dem Workspace entfernt
  mit rm(list=ls()) werden alle Objekte aus dem Workspace entfernt
- Arbeitsverzeichnis ermitteln und setzen
  getwd() ermittelt das Arbeitsverzeichnis
  setwd(Verzeichnisname) setzt das Arbeitsverzeichnis
  Alternative: ***Datei – Verzeichnis wechseln***
- Pakete installieren, aktualisiern
  Pakete sind thematisch gebündelte und aufeinander abgestimmte, oft umfangreiche Programmsammlungen
  library(Paketname) lädt ein Paket
  Installation über die Oberfläche: ***Pakete – lade Paket; Pakete – Aktualisiere Paket; Pakete – Installiere Pakete aus lokalen zip-Dateien***
- Pakete laden:
  library(Paketname)
  Paket – Lade Paket
- ***Hilfe – Menü***: Handbücher, find, apropos, html-Help, FAQ, ...

Graphikmodus:

- Speichern von Graphiken: ***Datei – Speichern als*** ...
  es kann zwischen verschiedenen Ausgabeformaten gewählt werden
  Ein button kopiert die Graphik als Metafile in die Zwischenablage
- Vergößern, Verkleinern: Ziehen am Graphikfenster; Abgespeichert wird die aktuelle Form und Größe
- Historyfunktion: wenn eingeschaltet, kann mit page up and down durch die erzeugten Graphiken geblättert werden.

1.4 Bezug von R

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

1.5 Informationen zu R-Graphiken

http://addictedtor.free.fr/graphiques/thumbs.php?sort=keywords

http://addictedtor.free.fr/graphiques/

1.6 Weitere Hilsfmittel

- JGR: Jaguar – eine Java-basierte Benutzeroberfläche für R (wird ständig weiterentwickelt)
- R-Commander: eine in R integrierte Benutzeroberfäche, erfordert die Installation und das Laden des Rcmdr-packages. Hat Schnittstellen zu einer Reihe wichtiger Statistikroutinen.
- Tinn-R: ein sehr guter MDI-Editor mit Syntax-highlighting für viele Programmier- und Skriptsprachen, z.B. für R.

alle Dateien sind in dem Verzeichnis "Kurzeinführung in R" enthalten.

1.7 Bücher zu R

fett: an der Abteilung vorhanden.

Bernhard Pfaff. Analysis of Integrated and Cointegrated Time Series with R. Use R. Springer, 2006. ISBN 0-387-98784-3.

Brian Everitt and Torsten Hothorn. A Handbook of Statistical Analyses Using R. Chapman & Hall/CRC, Boca Raton, FL, 2006. ISBN 1-584-88539-4.

Handl, Andreas. Multivariate Analysenmethoden – Theorie und Praxis multivariater Verfahren unter besonderer Berücksichtigung von S-Plus. Springer-Verlag, Heidelberg, 2002. ISBN 3-540-43386-4.

Jana Jureckova and Jan Picek. Robust Statistical Methods with R. Chapman & Hall/CRC, Boca Raton, FL, 2006. ISBN 1-584-88454-1.

John Fox. An R and S-Plus Companion to Applied Regression. Sage Publications, Thousand Oaks, CA, USA, 2002. ISBN 0761922792.

John Maindonald and John Braun. Data Analysis and Graphics Using R. Cambridge University Press, Cambridge, 2003. ISBN 0-521-81336-0.

John Verzani. Using R for Introductory Statistics. Chapman & Hall/CRC, Boca Raton, FL, 2005. ISBN 1-584-88450-9.

Julian J. Faraway. Linear Models with R. Chapman & Hall/CRC, Boca Raton, FL, 2004. ISBN 1-584-88425-8.

Michael J. Crawley. Statistics: An Introduction using R. Wiley, 2005. ISBN 0-470-02297-3.

Paul Murrell. R Graphics. Chapman & Hall/CRC, Boca Raton, FL, 2005. ISBN 1-584-88486-X.

Peter Dalgaard. Introductory Statistics with R. Springer, 2002. ISBN 0-387-95475-9.

Uwe Ligges. Programmieren mit R. Springer-Verlag, Heidelberg, 2005. ISBN 3-540-20727-9, in German.

William N. Venables and Brian D. Ripley. Modern Applied Statistics with S. Fourth Edition. Springer, New York, 2002. ISBN 0-387-95457-0.

William N. Venables and Brian D. Ripley. S Programming. Springer, New York, 2000. ISBN 0-387-98966-8.

1.8 Hilfe

- help.start(): startet ausführliche html-Hilfe: packages, introduction, Suchfunktion, thematisch gruppierte Funktionenübersicht

- ?procname: gibt Hilfe zur Prozedur procname
- help(procname) wie oben
- help.search(string): durchsucht das Hilfesystem nach Einträgen (siehe ?help.search)
  z.B help.search("linear models")
- apropos(objname|regexp): liefert einen Textvektor mit allen Objekten in R (im Suchpfad), in denen ein Objekt objname oder ein regulärer Ausdruck (Textpatter, z.B. abc?x*.*) vorkommt.
- Die R-homepage: http://cran.r-project.org/ mit FAQ's und R-New

1.9 Dokumentationen

siehe R homepage und Literaturliste

- R-intro.pdf
- Rlecturenotes.pdf, hierzu gehört:
  Rscripts.zip
  Datasets.zip
- SimpleR.pdf, heirzu gehört das package:
  Simple_0.5.zip
- usingR.pdf
- refman.pdf: Referenzmanual mit allen R-Hilfetexten (ohne contributed packages)

alle Dateien sind in dem Verzeichnis "Kurzeinführung in R" enthalten.

1.10 Beispiele

Die R-Dokumentation zu den packages ist insgesamt sehr einheitlich aufgebaut und enthält eine schablonisierte aber umfassende Beschreibung der Funktionen der packages (Name, usage, Parameterlisten mit Erläuterung, Autor, ev. links, Literatur und Beispiele). Die Beispiele können mit cut and paste in die R-Konsole kopiert werden. Vorher ist aber das jeweilige package mit library(packagename) zu laden. Oft ist auch eine Demosektion vorhanden:

- demo(): Startet Demos; siehe ?demo für Details.
  z.B. demo()
  demo(graphics)
  demo(package = .packages(all.available = TRUE))
  demo(lm.glm, package="stats")
- example(): Startet Beispiele zu einem package; siehe ?example für Details.
  z.B. example("smooth", package="stats", lib.loc=.Library)

1.11 Eine Einführungssitzung

(Beispiele aus Venables and Ripley (2002), modifiziert)

...\Kurzeinführung in R\Beispielskripte\Beispiel_1.R

```
# alle Objekte löschen:
rm(list=ls())
# libaries MASS und DAAG laden
```

```
library(MASS)
library(DAAG)
# Graphikausgabe unterteilen (hier nur eine Gaphik)
par(mfrow=c(1,1))
# Erzeugen normalverteilter Zufallszahlen
x <- rnorm(10000)
y <- rnorm(10000)
# Darstellung als Histogramm
truehist(c(x,y+3),nbins=50)
pause() # mit return geht's weiter
```

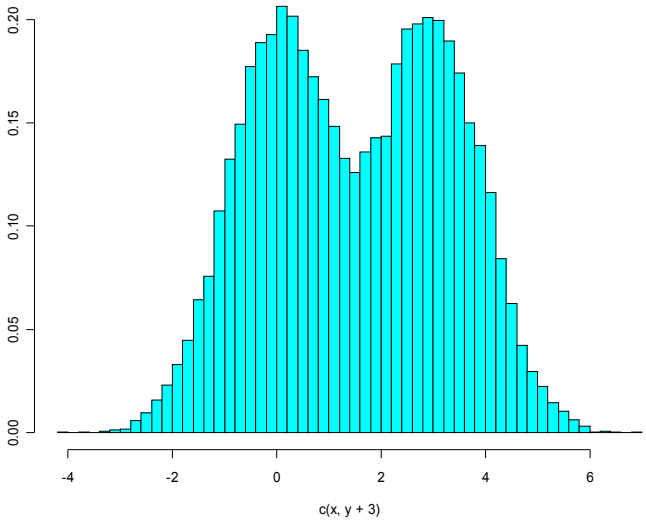

Abbildung 1: Histogramm zu c(x,y+3) mit 50 Intervallen

```
# Erzeugen einer geglätteten zweidimensionalen Dichtefunktion der Daten
dd <- kde2d(x,y,n=40)
# Darstellung als Contourplot mit verschiedenen Farbschemata
filled.contour(dd,color = terrain.colors,nlevels=50)
pause()
filled.contour(dd,color = rainbow,nlevels=50)
#pause()
filled.contour(dd,color = heat.colors,nlevels=50)
#pause()
filled.contour(dd,color = cm.colors,nlevels=50)
#pause()
filled.contour(dd,color = topo.colors,nlevels=50, plot.axes={
axis(1); axis(2); points(dd) })
filled.contour(dd,color = topo.colors,nlevels=50, plot.axes=
{axis(1,at=seq(-3.5,3.5,by=1));axis(2,at=seq(-3.5,3.5,by=1))})
title(main = "2D density plot", font.main = 4)
pause()
# Andere Art der Datstellung
image(dd,col=topo.colors(100), axes=FALSE) # Graphik ohne Achsen
contour(dd,nlevels=10,add=T) # Hinzufügen von Contourlinien
```

```
# Achsenbeschriftungen
axis(1, at = seq(-3.5, 3.5, by = 1),add=T)
axis(2, at = seq(-3.5, 3.5, by = 1),add=T)
title(main = "2D density plot", font.main = 4)
# Box um Graphik zeichnen:
box()
```

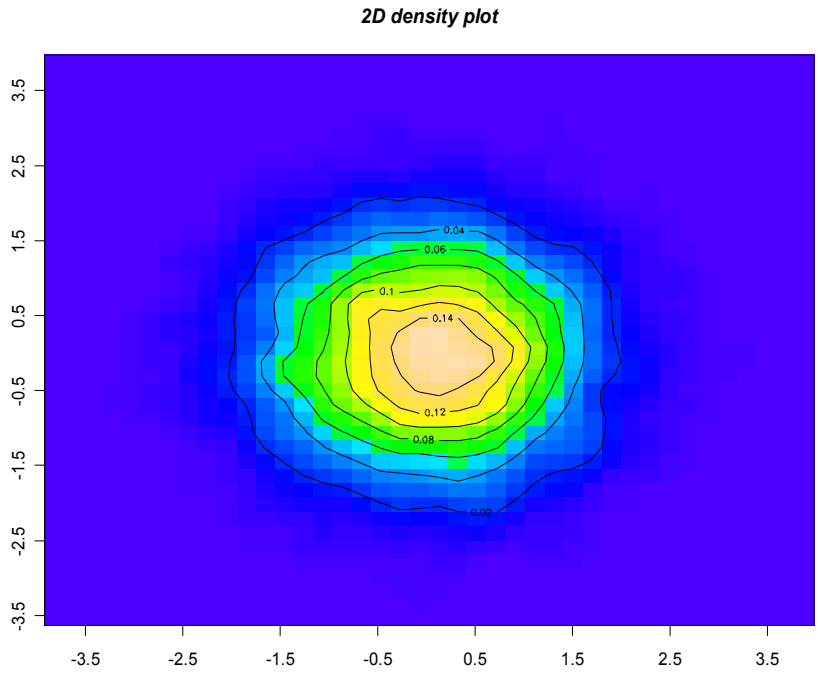

Abbildung 2: 2D-Dichteplot von x und y

...\Kurzeinführung in R\Beispielskripte\Beispiel_2.R

```
# alle Objekte löschen:
rm(list=ls())
# libaries MASS und DAAG laden
library(MASS)
library(DAAG)
# Graphikausgabe unterteilen (hier nur eine Gaphik)
par(mfrow=c(1,1))
# Stufenweise einen data-frame erzeugen
x <- seq(1,20,0.5) # x enthält Zahlensequenz
w <- 1+x/2 # (x,w) ist eine Gerade
y <- x+w*rnorm(x) # y ist eine verrauschte Gerade
dum <- data.frame(x,y,w) # Anschauen mit "dum ↵"
rm(x,y,w)
attach(dum) # Variable in dum können direkt angesprochen werden
# Lineare Regression
fm <- lm(y~x, data=dum)
print(summary(fm))
pause()
```

```
# Gewichtete Regression mit w
fm1 <- lm(y~x, data=dum,weight=1/w^2)
print(summary(fm1))
pause()
# Glatte Regression
lrf <- loess(y~x, dum)
print(summary(lrf))
pause()
# Plot der Regressionsgeraden bzw. -kurven
print("******** Plot der Regressionsgeraden bzw. -kurven
***********")
plot(x,y,asp=1)
lines(spline(x,fitted(lrf)),col=2,lwd=2)
points(x,fitted(lrf),pch=15)
abline(0,1,lty=3,col=3,lwd=2)
abline(fm,col=4,lwd=3)
abline(fm1,col=5,lty=4,lwd=3)
pause()
```

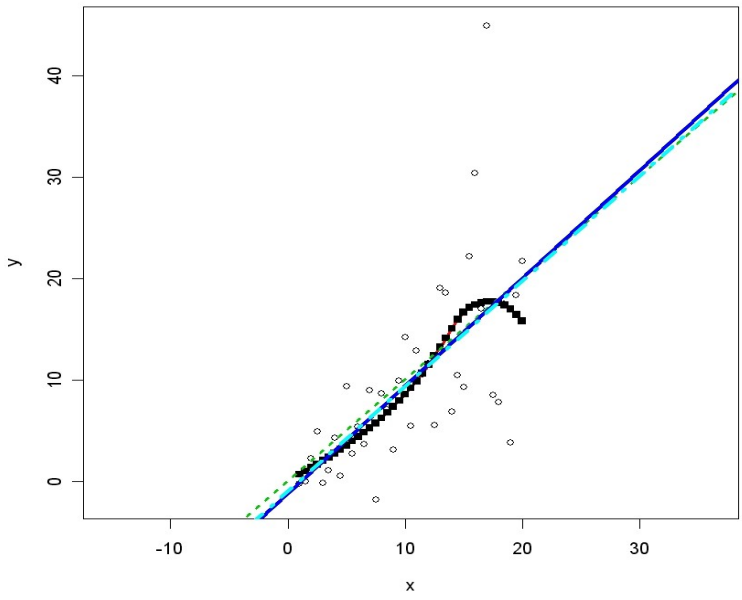

Abbildung 3: Verschiedene Regressionen: linear, gewichtet, gleitend

```
# Graphikausgabe unterteilen (hier neun Gaphiken)
par(mfrow=c(3,3))
# Residuen der drei Methoden
plot(fitted(fm),resid(fm),xlab="Fitted Values",
ylab="Residuals",main="Standard")
plot(fitted(fm1),resid(fm1),xlab="Fitted Values",
ylab="Residuals",main="Weighted")
plot(fitted(lrf),resid(lrf),xlab="Fitted Values",
ylab="Residuals",main="Smooth")
# Gefitted gegen gemessen
plot(y,fitted(fm),xlab="y",
```

```
ylab="fitted",main="measured vs. fitted, standard",asp=1)
plot(y,fitted(fm1),xlab="y",
ylab="fitted",main="measured vs. fitted, weighted",asp=1)
plot(y,fitted(lrf),xlab="y",
ylab="fitted",main="measured vs. fitted, smooth",asp=1)
# Quantilplots: emp. Quantile gegen theoretische Quantile
qqnorm(resid(fm),main="Q-Q standard"); qqline(resid(fm))
qqnorm(resid(fm1),main="Q-Q weighted"); qqline(resid(fm1))
qqnorm(resid(lrf),main="Q-Q smooth"); qqline(resid(lrf))
```

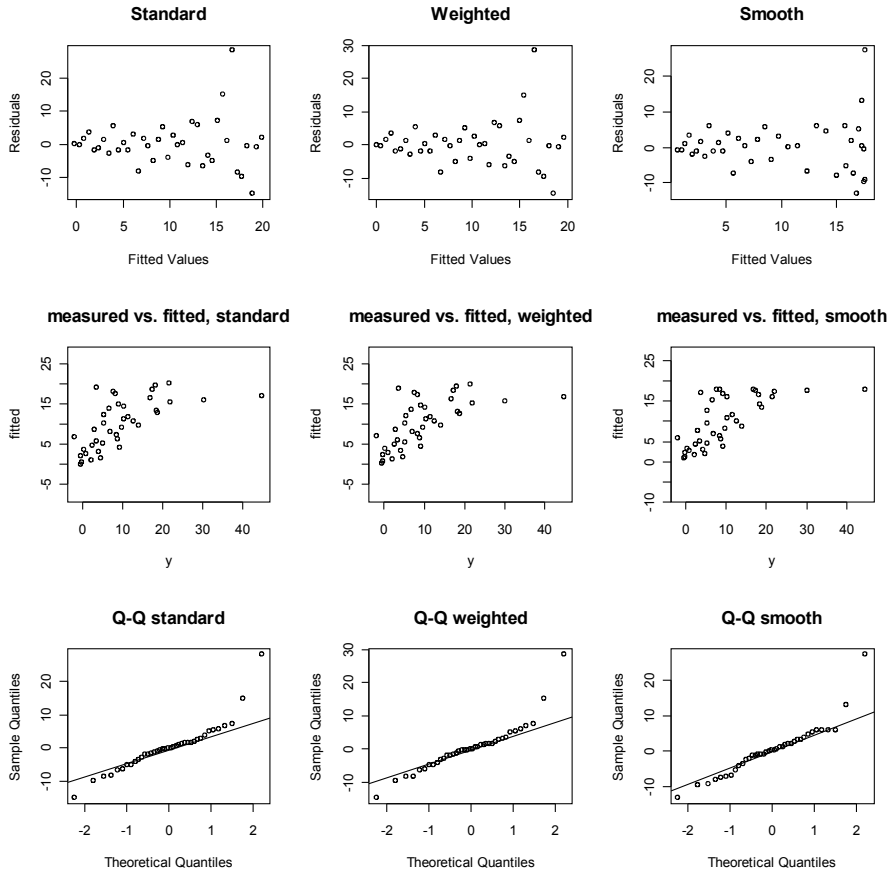

Abbildung 4: Regression: Qualitätscheck

```
# Korrelation gemessen-gefitted
print("normal R:"); print(cor(y,fitted(fm)))
print("weighted R:"); print(cor(y,fitted(fm1)))
print("smoothed R:"); print(cor(y,fitted(lrf)))
detach(dum) # hebt attach(dum) wieder auf
```

...\Kurzeinführung in R\Beispielskripte\Beispiel_3.R

```
# Löschen aller Objekte
rm(list=ls())
# Laden der libraries MASS und DAAG
library(MASS)
library(DAAG)
pairs(hills)
```

```
pause()
# interaktive Graphik
attach(hills)
plot(dist,time, main = "click on circles")
identify(dist,time,row.names(hills))
abline(lm(time~dist)) # Regression nach least squares
abline(rlm(time~dist),lty=3,col=4,lwd=2) # robust regression line
abline(lqs(time~dist),lty=3,col=4,lwd=2) # resistant regression
line
detach(hills)
# Regression mit selbst gezeichneten Punkten
plot(c(0,1),c(0,1),type="n", main = "set points")
xy <- locator(type="p")
abline(lm(y~x,xy),lwd=2,lty=1,col="blue") # Regression nach least
squares
abline(rlm(y~x,xy),lwd=2,lty=2,col="red") # robust regression line
abline(lqs(y~x,xy),lwd=2,lty=3,col="green") # resistant regression
line
```

...\Kurzeinführung in R\Beispielskripte\Beispiel_4.R

```
# Löschen aller Objekte
rm(list=ls())
# Laden der libraries DAAG und MASS
library(DAAG)
library(MASS)
# data frame "michaelson" attached
attach(michelson)
sp <- search()
# Boxplots von Speed als Fu von Expt
plot(Expt,Speed,main="Speed of light data",xlab="Experiment No.")
print("-------------------- Zweiweg-Anova ------------------")
pause()
# Zweiweg Anova
fm <- aov(Speed~Run+Expt)
print(summary(fm))
print("--------------- Entfernen eines Faktors -------------")
pause()
# Entfernen eines Faktors
fm0 <- update(fm, .~ . - Run)
print(summary(fm0))
pause()
detach(michelson)
```

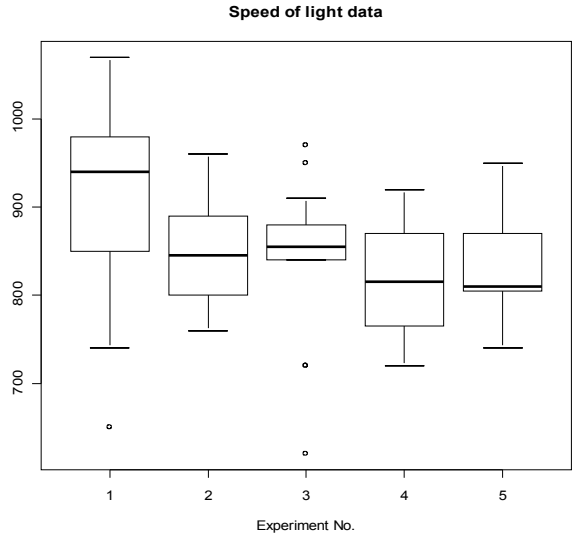

Abbildung 5: Boxplot der Lichtgeschwindigkeit als Funktion der Experimentnummer

2 Erste Schritte

2.1 R als Taschenrechner

```
1 + 2*3 # Bestätigen mit return 
2 * 5^2 - 10*5
(5+3)*(tan(2)+cos(4.23))
4 * sin( pi / 2 )
0/0 # gibt NaN
```

2.2 Operatoren

R kennt die üblichen Operatoren (aus R Development Core Team, R Language Definition, Version 2.1.1 (2005-06-20) DRAFT)

-	Minus, can be unary or binary
+	Plus, can be unary or binary
!	Unary not
~	Tilde, used for model formulae, can be either unary or binary
?	Help
:	Sequence, binary (in model formulae: interaction)
*	Multiplication, binary
/	Division, binary
^	Exponentiation, binary
%x%	Special binary operators, x can be replaced by any valid name
%%	Modulus, binary
%/%	Integer divide, binary

`%*%`	Matrix product, binary
`%o%`	Outer product, binary
`%x%`	Kronecker product, binary
`%in%`	Matching operator, binary (in model formulae: nesting)
`<`	Less than, binary
`>`	Greater than, binary
`==`	Equal to, binary
`>=`	Greater than or equal to, binary
`<=`	Less than or equal to, binary
`&`	And, binary, vectorized
`&&`	And, binary, not vectorized
`|`	Or, binary, vectorized
`||`	Or, binary, not vectorized
`<-`	Left assignment, binary
`->`	Right assignment, binary
`$`	List subset, binary

2.3 Konstanten

`pi`	die Zahl π
`Inf, -Inf`	plus minus undendlich
`NaN`	Not a Number
`NA`	Not Available (fehlende Werte)
`NULL`	leere Menge

2.4 Eingebaute Fuktionen

- `ls():` listet alle Objekte auf
- `str(obj):` gibt die Struktur eines Objekts an
- `rm(obj):` entfernt ein Objekt

+++

- `print():` Gibt ein R-Object aus
- `cat()` Druckt mehrere Objekte, eins nach dem anderen
- `length()` Ergibt die Anzahl der Elemente eines Vektors oder einer Liste

+++

- `mean()` Mittelwert
- `median()` Median
- `range()` Spannweite
- `var()` Varianz
- `sd()` Standardabweichung
- `cov()` Covarianz

- `cor()` Correlation

+++

- `sort()` Sortiert die Elemente eines Vektors ohne NAs
- `cumsum()` kumulative Summe
- `cumprod()` kumulatives Produkt
- `rev()` kehrt die Ordnung eines Vektors um

+++

- `cos(x), sin(x), tan(x), acos(x), asin(x), atan(x),`
- `log(x, base = ...), logb(x, base = ...), log10(x), log2(x), exp(x)`
- `abs()`

2.5 Daten-Ein- und Ausgabe

- `getwd()`: liefert das aktuelle Arbeitsverzeichnis
  z.B.
  ```
  getwd()
  [1] "C:/Dokumente und Einstellungen/Bernd Huwe/Eigene
  Dateien/Programme/ R_doku/ Kurzeinführung in
  R/Beispielskripte"
  ```
- `setwd()`: setzt das aktuelle Arbeitsverzeichnis
  z.B.: `setwd("C:/Dokumente und Einstellungen/Bernd Huwe/Eigene Dateien/Programme /R_doku/Kurzeinführung in R/Beispielskripte")`
- `read.table()`: liest aus einer Textdatei einen Datensatz mit mehreren Spalten. Die Spalten können verschiedene Typen enthalten.
  z.B.: `tbl <- read.table("testdatei.txt", header = TRUE)`
- `write.table()`: schreibt einen data frame in eine Textdatei
  z.B. write.table(a, file="a.dat")
- `source()`: liest eine Datei mit R-befehlen
  z.B.: `source("c:/temp/ifile.R")`
- `sink()`: leitet die Bildschirmausgabe in eine Datei um
  ```
  sink("sink-examp.txt")
  sink() # zurück zur Console
  ```
- `save(filename)`: schreibt R-Objekte in den angebenen file.
  ```
  x <- runif(20); y <- list(a = 1, b = TRUE, c = "oops");
  save(x, y, file = "xy.Rdata")
  save(list = ls(all=TRUE), file = ".RData") # passiert bei q("yes")
  ```
- `save.image(filename)`: speichert den aktuellen Workspace in der angegebenen Datei.
- `load(filename)`: liest R-Objekte aus dem angebenen file.
  ```
  save(list = ls(all=TRUE), file= "all.Rdata");
  ```

```
load("all.Rdata")
```

- `.Ristory`
  bei Beenden mit q("yes") werden Befehle in der Datei ".Rhistory" gespeichert
- `.Rdata`
  bei Beenden mit q("yes") wird der Workspace in der Datei ".Rdata" gespeichert

2.6 Elementare Handgriffe

- Kommentare: #
- einfache Berechnungen
```
5+5
sin(0.1*1:100)
```
- Zuweisungen
```
a <- 4.5
12.3 -> b
y <- x <- 5
```
- Vektoren
```
a <- 1:10
b <- (4:8)*0.1
trf <- c(4,8,1:6,9)
abc <- c("a","b","c","?")
value <- c(rep(c(3,4),6)
rp <- rep(3:5,1:3)
ro <- rep(4:8,c(4,3,6)
comb <- c(a,b,value,3:8)
sqtzu <- seq(3,-2,by=-0.5)
```
- Matrizen
```
Z <- matrix(c(4,7,3,8,9,2),ncol = 3); Z↵
Y <- matrix(c(4,7,3,8,9,2),nrow = 3, byrow = TRUE); Y↵
A <- matrix(c(3,8,9,2),ncol = 2)
Ainv <- solve(A)
Ainv %*% A
A^(-1)
1/A
```
- outer(x,y)
```
x <- 1:5
y <- 6:10
z <- outer(x,y)
z
      [,1] [,2] [,3] [,4] [,5]
[1,]    6    7    8    9   10
[2,]   12   14   16   18   20
[3,]   18   21   24   27   30
[4,]   24   28   32   36   40
[5,]   30   35   40   45   50
outer(x,y,"+")
outer(x,y,"*")
```

algemein: `outer(x,y,f)` mit f Funktion

- Eigenwerte
```
eigen(z)
v <- eigen(z)
v
str(v)
z%*%v$vectors[,1]
v$values[1]*v$vectors[,1]
```
- Arrays: Verallgemeinerung von Matrizen auf beliebige Dimensionen
- `mean(x), median(x), var(x), summary(a)`: ausprobieren mit `x <- rnorm(100)`!
- data.frames
  sind Datensätze in Spalten; die Spalten können unterschiedliche Datentypen enthalten
  Indizierung ähnlich wie Matrizen
- Vektorisierte Berechnungen
```
x <- c(3,2,5,6,7,2,3,5:10)
x
sin(x)
```
- Faktoren
  diskrete Merkmale; kann mit `factor()` erzeugt werden
  intern als Nummer extern als Namen dargestellt
- Listen: Sammlung beliebiger Objekte (z.B. Skalare, Text, Plots, Ergebnisse einer Regression)
  Erzeugen mit `list()`:
  z.B.
```
L1 <- list(c(1,2,5,4), matrix(1:4,2), c("Hallo", " Welt"), 1)
L1 ↵
```
  Zugriff auf Elemente von Liste L1 mit `L1[[i]]`
- Fehlende Werte
  meist `NA`

2.7 Konstrukte

- Bedingungen:
```
if(Bedingung)
 {Anweisungen}
else
 {Anweisungen}
```

  Beispiel:
```
x <- 5
if(x < 99) print("x < 99")
```
- Schleifen:
  3 Typen
```
repeat {Anweisungen}
while(Bedingung) {Anweisungen}
```

```
for(in in M) {Anweisungen}
```

Beispiele:

```
x <- c(3,6,4,8,0)
for(i in x) print(i^2) # i ist Mengenelement
oder
for(in in seq(along=x)) print(x[i^2]) # i ist hier Index

i <- 0; s <- 0
while(i<=8) {s <- s + i; i <- i+1; print(c(i,s))}

i <- 0
repeat{
     i <- i+1
     if(i<3) next # springt zum Anfang der Schleife
     print(i)
     if(i == 3) break # stoppt die Schleife
}
```

2.8 Zeichenketten

Beispiele:

```
x <- 8.25
cat("das Objekt x hat den Wert: ", x, "\n", sep = "\t")

paste("Datei", 1:3, ".txt", sep = "")
```

2.9 Eigene Funktionen

- Funktionsdefinition

```
MeineFunkion <- function(Argumente) {
     Anweisungen
}
```

- Funktionsaufruf

```
MeineFunktion(Argumente)
```

...\Kurzeinführung in R\Beispielskripte\Beispiel_5.R

```
sumup <- function(i=0,s=0){
     while(i<=8) {s <- s + i; i <- i+1; print(c(i,s))}
}
sumup()
sumup(3,4)
```

3 Elementarstatistik

3.1 Verteilungen

- R kennt die wichtigsten statistischen Verteilungen: Nachschlagen in Tabellen ist somit überflüssig.

Tabelle 1: Verfügbare Verteilungen in R

Distribution	R name	additional arguments
beta	beta	shape1, shape2, ncp
binomial	binom	size, prob
Cauchy	cauchy	location, scale
chi-squared	chisq	df, ncp
exponential	exp	rate
F	f	df1, df2, ncp
gamma	gamma	shape, scale
geometric	geom	prob
hypergeometric	hyper	m, n, k
log-normal	lnorm	meanlog, sdlog
logistic	logis	location, scale
negative binomial	nbinom	size, prob
normal	norm	mean, sd
Poisson	pois	lambda
Student's t	t	df, ncp
uniform	unif	min, max
Weibull	weibull	shape, scale
Wilcoxon	wilcox	m, n

- Aufruf der Verteilungsfunktionen
  für jede Verteilung gibt es 4 Funktion die durch den vorgestellten Bucstaben unterschieden werden:
  d (density) Dichtefunktion
  ```
  dnorm(x, mean=0, sd=1, log = FALSE)
  ```
  p (probability) Verteilungsfunktion
  ```
  pnorm(q, mean=0, sd=1, lower.tail = TRUE, log.p = FALSE)
  ```
  q (quantiles) Quantile
  ```
  qnorm(p, mean=0, sd=1, lower.tail = TRUE, log.p = FALSE)
  ```
  r (random)
  ```
  rnorm(n, mean=0, sd=1)
  ```

  ```
  Beispiele:

  x <- seq(-4,4,0.1)
  plot(x,dnorm(x),type="l",lwd=2)
  ```

...\Kurzeinführung in R\Beispielskripte\Beispiel_6.R

Funktion zur Erzeugung von Zufallszahlen, Histogrammen und zugehöriger theoretischer Verteilung:

```
sim <- function(ncol=2,nrow=2,ndat=40){
     par(mfrow=c(nrow,ncol))
     M <- 1:(ncol*nrow)
     #set.seed(54321)
     for(i in M){
          x <- rnorm(ndat)          # simulation von Zufallszahlen
          h <- hist(x,plot=F)       # Zeichnen des Histogramms
          yrange <- range(0, h$density, dnorm(0)) # sucht Min
                                    und Max aus angeg. Werten
          hist(x, freq=F, ylim=yrange)
          curve(dnorm(x), add=T)    # Normalverteilung
     }
}
sim(4,4,30)
```

Bem.: für multivariate Verteilungen gibt es auf CRAN das Package mvtnorm (multivariate Normal- und t-Verteilung)

3.2 Stichproben

Das Ziehen von Stichproben aus einer endlichen Menge (z.B. Zuordnung von Behandlungen zu Flächen) geschieht mit der Funktion sample(). Nachstehend hierzu ein Beispiel zum Ziehen mit und ohne Zurücklegen

...\Kurzeinführung in R\Beispielskripte\Beispiel_7.R

```
set.seed(62345)
smpl <- function(MLoop=5,NPop=15,NSample=5)
{
  if(NSample >= NPop)
  {
    print("NPop < NSample: ==> break")
  } else
  {
    IM <- 1:MLoop
    print("--- ohne Zurücklegen ---")
    for(i in IM) {
        x <- sample(1:NPop,NSample,replace=FALSE);print(x)
    }
    print("--- mit Zurücklegen ---")
    for(i in IM) {
        x <- sample(1:NPop,NSample,replace=TRUE);print(x)}
  }
}
```

3.3 Deskriptive Statistik

Deskriptive Statistic dient dem Überblick über die Datenstruktur. Hierzu gehören die Funktionen mean(), median() und var(). Einen guten Überblick über einen data frame liefert summary(). Visuell sind scatterplots, boxplots, violinplots gute Hilfsmittel.

...\Kurzeinführung in R\Beispielskripte\Beispiel_8.R

```
rm(list=ls())
library(DAAG)
par(mfrow=c(1,1))
# Einlesen des data frame d
d <- read.table("Q2_B_all_mod.dat",header=TRUE)
# Ausschluss etlicher Spalten
dat <- d[,-c(1,2,3,4,11,13,16,17,19)]
# Ausschluss der Zeilen mit fehlenden Werten (NA's)
cc <- complete.cases(dat)
dat <- dat[cc,] # enthält keine fehlenden Werte mehr
# Tabellarische Übersicht über die Daten
print(summary(dat))
pause()
# erzeugt Matrix von Scatterplots
plot(dat)
```

...\Kurzeinführung in R\Beispielskripte\Beispiel_9.R

das nachstehende Skript berechnet für den selben Datensatz eine Korrelationstabelle:

```
rm(list=ls())
library(DAAG)
par(mfrow=c(1,1))
# Einlesen des data frame d
d <- read.table("Q2_B_all_mod.dat",header=TRUE)
# Ausschluss etlicher Spalten
dat <- d[,-c(1,2,3,4,11,13,16,17,19)]
# Ausschluss der Zeilen mit fehlenden Werten (NA's)
cc <- complete.cases(dat)
dat <- dat[cc,]
# erzeugt Matrix von Scatterplots
plot(dat)
pause()
# Berechnen der Korrelationsmatrix
a<-matrix(1:100,10,10)
for(i in 1:10){
     for(j in 1:10){
          a[i,j]<-cor(dat[,i],dat[,j],method="pearson")
}}
print(a)
```

3.4 Vertrauensintervalle

Vertrauensintervalle geben obere und untere Grenzen eines Intervalls an, das mit einer gewissen Irrtumswahrscheinlichkeit einen bestimmten Parameter der Grundgesamtheit umfasst.

3.4.1 Der z-test

Annahme: σ ist bekannt. Dann ist die Größe

$$Z = \frac{\bar{X} - \mu}{\sigma / \sqrt{n}}$$

N(0,1) – verteilt. Aus der Normalverteilung können dann die 0.05 und 0.95 – Quantile abgegriffen werden und hieraus die Vertrauensintervalle berechnet werden.

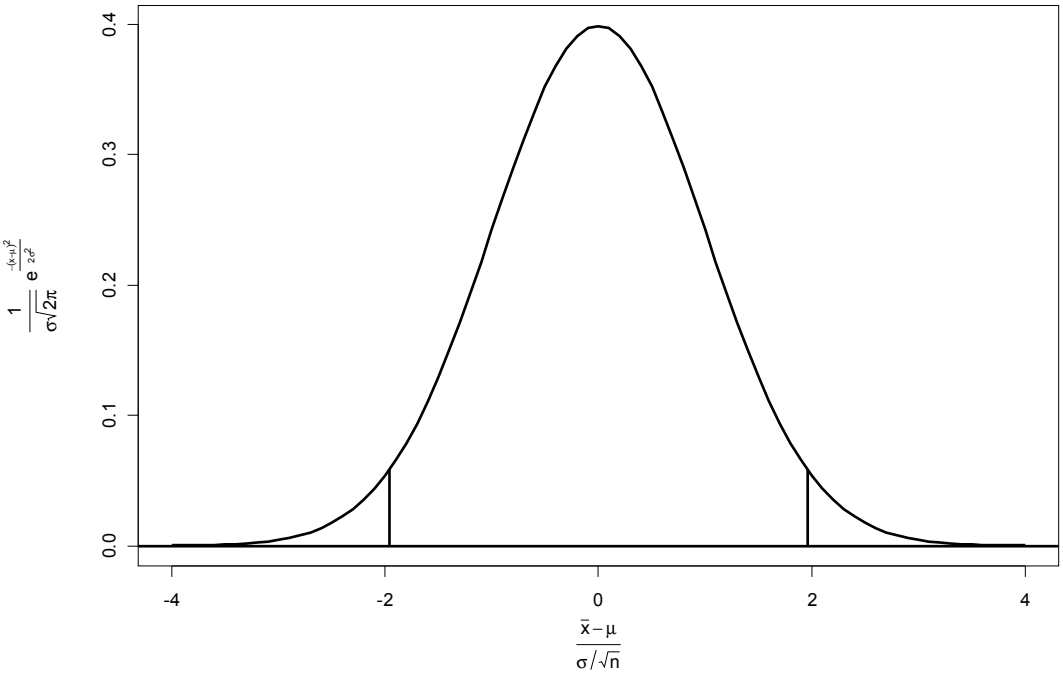

Abbildung 6: Bestimmung eines Konfidenzintervalls für eine Irrtumswahrscheinlichkeit von 95%

...\Kurzeinführung in R\Beispielskripte\Beispiel_10.R

Die mit nachstehendem Skript erzeugte Graphik zeigt die N(0,1)-Normalverteilung mit Konfidenzgrenzen für eine Irrtumswahrscheinlichkeit von 95%:

```
rm(list=ls())
library(grDevices)
# Ränder einstellen
par(mar=c(6,8,1,1),mgp=c(4,1,0))
# Zeichnen der Normalverteilung
# mit Formelbeschriftung der Achsen (siehe ?plotmath für Deatails)
x <- seq(-4,4,0.1)
plot(x,dnorm(x),type="l",lwd=2,
ylab=expression(frac(1,sigma*sqrt(2*pi))~~e^{frac(-(x-
mu)^2,2*sigma^2)})
,xlab=expression(frac({bar(x)-mu},{sigma/sqrt(n)})))
abline(0,0,lwd=2)
# Einzeichnen der Konfidenzgrenzen
px <- c(qnorm(0.025),qnorm(0.975))
py <- dnorm(px)
points(px,py,lwd=2,type="h")
```

...\Kurzeinführung in R\Beispielskripte\Beispiel_11.R

Die Konfidenzgrenzen für Stichprobe x und einer vorgegebene Irrtumswahrscheinlichkeit können

mit folgender Funktion berechnet werden:

```
rm(list=ls())
conlim <- function(x = rnorm(10),alpha=0.05){
  n <- length(x)
  xmean <- mean(x)
  xstd <- sqrt(var(x))
  lowalpha <- alpha/2
  upalpha <- 1- lowalpha
  px <- c(qnorm(lowalpha),qnorm(upalpha))
  clu <- c(xmean+px[1]*xstd/sqrt(n),xmean+px[2]*xstd/sqrt(n))
  return(clu)
}
```

3.4.2 Eine kleine Simulationsstudie

Das Verhalten der Konfidenzintervalle soll hier durch eine kleine Simulationsstudie verdeutlicht werden. Es lohnt sich durchaus, im Skript die Parameter NSample und alpha zu variieren und das Ergebnis zu analysieren.

...\Kurzeinführung in R\Beispielskripte\Beispiel_12.R

```
rm(list=ls())
# Funktion conlim zur Berechnung der Konfidenzlimits
conlim <- function(x = rnorm(10),alpha=0.05){
  n <- length(x)
  xmean <- mean(x)
  xstd <- sqrt(var(x))
  lowalpha <- alpha/2
  upalpha <- 1- lowalpha
  px <- c(qnorm(lowalpha),qnorm(upalpha))
  clu <- c(xmean+px[1]*xstd/sqrt(n),xmean+px[2]*xstd/sqrt(n))
  return(clu)
}
# Schleife mit NExp zufälligen, normalverteilten Daten
# NSample: Stichprobenumfang, Mue: Mittelwert usw.
NExp    <- 40; NSample <- 10; Mue <- 5; SDev <- 3; alpha <- 0.05
dlow <- rep(0,NExp)
dupp <- rep(0,NExp)
for(i in 1: NExp){
  x <- rnorm(NSample,Mue,SDev)
  dlow[i] <- conlim(x, alpha)[1] # da Rückgabe von conlim 2
Elemente enthält
  dupp[i] <- conlim(x, alpha)[2]
}
cl.frame <- cbind(dlow,dupp)
# Plot ohne Daten
XRange <- range(cl.frame)
plot(cl.frame,xlim=XRange,ylim=c(0,NExp),type="n",xlab="Confidence
limits",ylab="Experiment")
# Plot der einzelnen Konfidenzintervalle
delta <- NExp/(NExp-1)
```

```
y <- 0
for(i in 1:NExp){
  px <- c(dlow[i],dupp[i])
  py <- c(y,y)
  lines(px,py,lwd=2,type="l")
  y <- y + delta
}
# Plot des "wahren" Erwartungswerts
abline(v=Mue,lwd=3,col="blue")
```

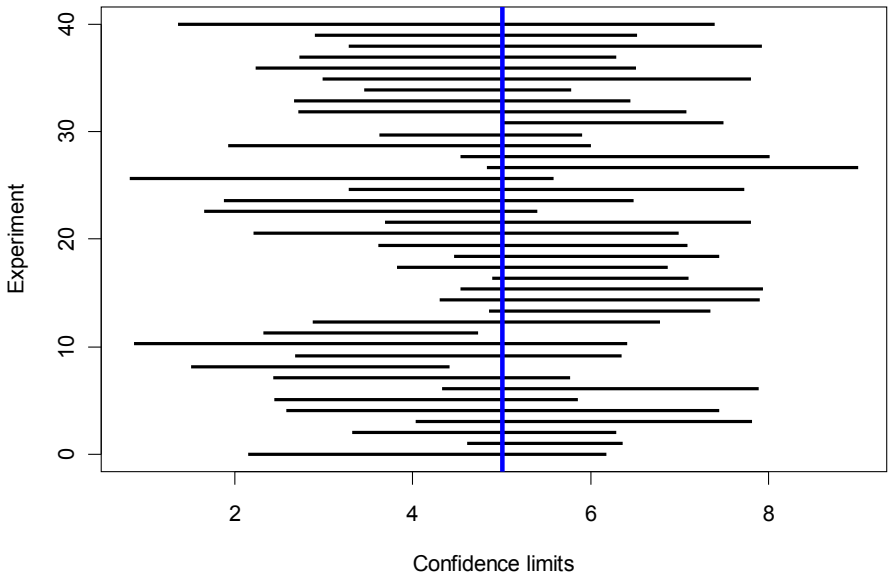

Abbildung 7: Vertrauensintervalle von 40 zufällig gezogenen Stichproben vom Umfang 10; die senkrechte Linie kennzeichnet den wahren Erwartungswert der Grundgesamtheit.

3.4.3 Der t-Test

Ist die Varianz nicht bekannt, so folgt

$$T = \frac{\bar{X} - \mu}{s/\sqrt{n}}$$

einer t-Verteilung mit n-1 Freiheitsgraden. Ansonsten ist das Vorgehen wie in 3.4.1 beschrieben.

...\Kurzeinführung in R\Beispielskripte\Beispiel_13.R

```
conlim.t <- function(x = rnorm(10),alpha=0.05){
  n <- length(x)
  xmean <- mean(x)
  xstd <- sqrt(var(x))
  lowalpha <- alpha/2
  upalpha <- 1- lowalpha
  px <- c(qt(lowalpha,n-1),qt(upalpha,n-1))
  clu <- c(xmean+px[1]*xstd/sqrt(n),xmean+px[2]*xstd/sqrt(n))
```

```
  return(clu)
}
```

Vergleichen Sie die Funktion conlim und conlim.t, indem Sie sie auf die gleiche Stichprobe anwenden: welches Vertrauensintervall ist enger?

3.4.4 Konfidenz-Intervall für den Median

Hier kommt der nicht-parametrische Wilcoxon-Test ur Anwendung:

```
x <- c(3,4,5,3,7,7,8,9,2,5,6)
wilcox.test(x,conf.int=TRUE)
```

Ergebnis:

```
        Wilcoxon signed rank test with continuity correction

data:  x
V = 66, p-value = 0.003805
alternative hypothesis: true mu is not equal to 0
95 percent confidence interval:
 3.500003 6.999934
sample estimates:
(pseudo)median
      5.499972

Warning messages:
1: cannot compute exact p-value with ties in:
wilcox.test.default(x, conf.int = TRUE)
2: cannot compute exact confidence interval with ties in:
wilcox.test.default(x, conf.int = TRUE)
```

3.5 Ein- und Zwei-Stichprobentests

Die nachstehend aufgelisteten Beispiele stammen aus dem Buch von Dalgaard (2002) und wurden für die Einführung überarbeitet (vervollständigt und kommentiert).

t-Test für eine Stichprobe

- Daten $N(\mu,\sigma)$–verteilt
- σ aus Daten geschätzt
- H_0: $\mu = \mu_0$

```
# Chapter 4.1 one sample t-test
#------------------------------------------
# Laden der Datenpackagew ISwR unD DAAG aus R
rm(list=ls())
library(DAAG)
print("")
print(" one sample t-test:")
print("-----------------------------------------------------------")
```

```
# Dateneingabe
daily.intake <-
c(5260,5470,5640,6180,6390,6515,6805,7515,7515,8230,8770)
print(mean(daily.intake))
print(sd(daily.intake))
print(quantile(daily.intake))
# t-tests für verschiedene Situationen
print(t.test(daily.intake,mu=7770))
pause()
print(t.test(daily.intake,mu=8000),alternative="less")
pause()
print(t.test(daily.intake,mu=5000),alternative="greater")
pause()
#
print(" ")
print("----------------------  Ende ---------------------------")
print(" ")
# Ende
```

Wilcoxon Vorzeichen-Rang-Test für eine Stichprobe

Daten verteilungsfrei

- $H_0: \mu = \mu_0$

```
# Chapter 4.2 Wilcoxon signed rank test
#-----------------------------------------
# Laden der Datenpackagew ISwR unD DAAG aus R
rm(list=ls())
library(DAAG)
print("")
print(" Wilcoxon signed rank test:")
print("------------------------------------------------------------")
#
daily.intake <-
c(5260,5470,5640,6180,6390,6515,6805,7515,7515,8230,8770)
print(mean(daily.intake))
print(sd(daily.intake))
print(quantile(daily.intake))
#
print(wilcox.test(daily.intake,mu=7770))
pause()
#
print(" ")
print("------------------------- Ende ---------------------------")
print(" ")
# Ende
```

t-Test für zwei-Stichproben

- Daten N(μ,σ)–verteilt
- σ_1, σ_2 aus Daten geschätzt
- H_0: $\mu_1 = \mu_2$

```
# Chapter 4.3 Two-sample t-test
#-----------------------------------------
# Laden der Datenpackagew ISwR unD DAAG aus R
rm(list=ls())
library(ISwR)
library(DAAG)
print("")
print(" Two-sample t-test:")
print("-------------------------------------------------------------")
#
data(energy)
attach(energy)
print(energy)
#
print(t.test(expend~stature))
pause()
print(t.test(expend~stature,var.equal=T))
pause()
#
print(" ")
print("------------------------ Ende --------------------------")
print(" ")
# Ende
```

F-Test zum Vergleich von Varianzen

- Daten normalverteilt
- σ_1, σ_2 aus Daten geschätzt
- H_0: $\sigma_1/\sigma_2 = 1$

```
# Chapter 4.3 comparison ov variances
#-----------------------------------------
# Laden der Datenpackagew ISwR unD DAAG aus R
rm(list=ls())
library(ISwR)
library(DAAG)
print("")
print(" comparison of variances:")
print("-------------------------------------------------------------"
)
#
```

```
data(energy)
attach(energy)
print(energy)
#
print(var.test(expend~stature)) # gleicher Aufruf wie t.test
pause()
#
print(" ")
print("------------------------ Ende ----------------------------")
print(" ")
# Ende
```

Wilcoxon-Test für den Vergleich zweier Stichproben (Rangsummen)

Daten verteilungsfrei

- H_0: $\mu_1 = \mu_2$

```
# Chapter 4.5 Two-sample Wilcoxon test
#-----------------------------------------
# Laden der Datenpackagew ISwR unD DAAG aus R
rm(list=ls())
library(ISwR)
library(DAAG)
print("")
print(" Two-sample Wilcoxon test:")
print("------------------------------------------------------------")
#
data(energy)
attach(energy)
print(energy)
#
print(wilcox.test(expend~stature))
pause()
#
print(" ")
print("------------------------ Ende ----------------------------")
print(" ")
# Ende
```

t-Test für gepaarte Stichproben

- Daten normalverteilt, gepaart (d.h. je zwei Werte aus beiden Stichproben gehören zusammen
- σ aus Daten geschätzt
- H_0: $\mu_1 = \mu_2$

```
# Chapter 4.6 The paired t-test
#-----------------------------------------
# Laden der Datenpackagew ISwR unD DAAG aus R
rm(list=ls())
library(ISwR)
library(DAAG)
print("")
print(" The paired t-test:")
print("-------------------------------------------------------------")
#
data(intake)
attach(intake)
print(intake)
print(post-pre)
print(t.test(pre,post,paired=T))
#
pause()
#
print(" ")
print("---------------------- Ende ----------------------------")
print(" ")
# Ende
```

Wilcoxon Vorzeichen-Rang-Test für gepaarte Stichproben

- Daten verteilungsfrei, gepaart (d.h. je zwei Werte aus beiden Stichproben gehören zusammen
- $H_0: \mu_1 = \mu_2$

```
# Chapter 4.6 The matched paires Wilcoxon test
#-------------------------------------------------
# Laden der Datenpackagew ISwR unD DAAG aus R
rm(list=ls())
library(ISwR)
library(DAAG)
print("")
print(" The matched paires Wilcoxon test:")
print("-------------------------------------------------------------")
#
data(intake)
attach(intake)
print(intake)
print(post-pre)
print(wilcox.test(pre,post,paired=T))
#
pause()
#
print(" ")
print("---------------------- Ende ----------------------------")
```

```
print(" ")
# Ende
```

3.6 Regression und Korrelation

Auch die nachstehend aufgelisteten Beispiele stammen aus dem Buch von Dalgaard (2002) und wurden für die Einführung überarbeitet (vervollständigt und kommentiert).

Einfache lineare Regression

Lineare Regressionen werden mit dem Befehl lm() durchgeführt. Hierbei kommt in zunehmendem Umfang eine spezifische Formelschreibweise zur Anwendung, die auch bei anderen Statistikpaketen zu finden ist (z.B. a ~ b + c +b*d). Wichtig ist immer die Anwendung des summary-Befehls, der fast bei allen Prozeduren Zusatzinformationen liefert.

```
# Chapter 5.1 simple linear regression
#-----------------------------------------
# Laden der Datenpackagew ISwR unD DAAG aus R
rm(list=ls())
library(ISwR)
library(DAAG)
print("")
print(" Simple linear regression:")
print("-------------------------------------------------------------")
#
data(thuesen)
attach(thuesen)
print(str(thuesen))
a <- lm(short.velocity~blood.glucose)
print(a)
#
pause()
b <- summary(a)
print(b)
#
pause()
plot(blood.glucose,short.velocity)
abline(a)
print(" ")
print("--------------------- Ende ------------------------------")
print(" ")
# Ende
```

Residuen und gefittete Daten

Residuen und der Vergleich mit gefitteten Daten sind die abschließenden Kriterien für die Beurteilung der Qualität der Regression

```
# Chapter 5.2 residuals and fitted values
#-----------------------------------------
# Laden der Datenpackagew ISwR unD DAAG aus R
rm(list=ls())
library(ISwR)
library(DAAG)
print("")
print(" residuals and fitted values:")
print("-------------------------------------------------------------")
#
```

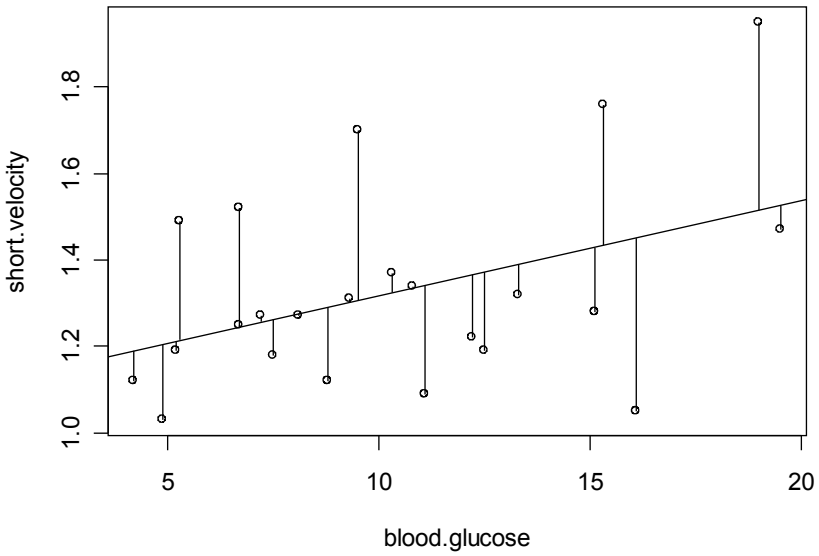

Abbildung 8: Anpassung der Regressionsgeraden an die Messdaten

```
options(na.action=na.exclude)
data(thuesen)
attach(thuesen)
print(str(thuesen))
#
a <- lm(short.velocity~blood.glucose)
print("fitted values:")
print(fitted(a))
print("residuals:")
print(resid(a))
#
plot(blood.glucose,short.velocity)
abline(a)
segments(blood.glucose,fitted(a),blood.glucose,short.velocity)
pause()
plot(fitted(a),resid(a))
pause()
qqnorm(resid(a))
pause()
```

```
par(mfrow=c(2,2),mfcol=c(2,2))
plot(a)
par(mfrow=c(1,1),mfcol=c(1,1))
print(" ")
print("---------------------- Ende ------------------------------")
print(" ")
# Ende
```

Konfidenzbänder

Unterschieden wird hier zwischen der Unsicherheit der Regressionslinie (enge Bänder) und der Unsicherheit der Vorhersagen mit der Regressionsbeziehung (weite Bänder)

```
# Chapter 5.3 prediction and conficence bands
#-----------------------------------------
# Laden der Datenpackagew ISwR unD DAAG aus R
rm(list=ls())
library(ISwR)
library(DAAG)
print("")
print(" prediction and conficence bands:")
print("-----------------------------------------------------------")
```

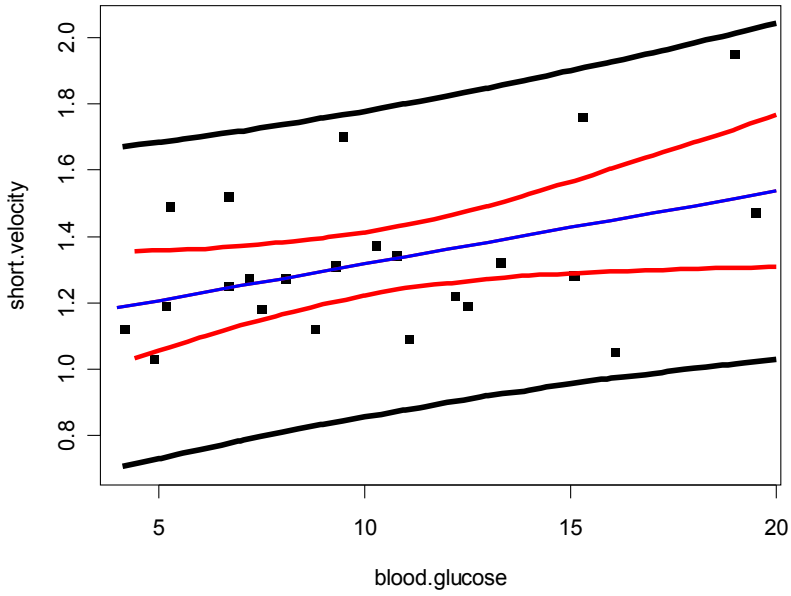

Abbildung 9: Konfidenzbänder einer linearen Regression.

```
#
options(na.action=na.exclude)
data(thuesen)
attach(thuesen)
print(str(thuesen))
#
```

```
a <- lm(short.velocity~blood.glucose)
print("fitted values:")
print(fitted(a))
print("residuals:")
print(resid(a))
#
print("prediction:")
b <- predict(a,int="c")
print(b)
pause()
#
pred.frame <- data.frame(blood.glucose=4:20)
pp <- predict(a,int="p",newdata=pred.frame) # "p": prediction
pc <- predict(a,int="c",newdata=pred.frame) # "c": conficence
plot(blood.glucose,short.velocity,ylim=range(short.velocity,pp,na.
rm=T),pch=15)
pred.gluc <- pred.frame$blood.glucose
matlines(pred.gluc,pc,lty=c(1,2,2),col="red",lwd=c(2,3,3))
matlines(pred.gluc,pp,lty=c(1,10,3),col=c("blue","green","black"),
lwd=c(2,5,15))
#
print(" ")
print("---------------------- Ende -----------------------------")
print(" ")
# Ende
```

Korrelation

In folgenden Beispielen werden Korrelationen mit verschiedenen Ansätzen berechnet, die sich v.a. hinsichtlich der statistischen Voraussetzungen unterscheiden (Pearson: normalverteilt, andere nicht).

```
# Chapter 5.4 Correlation
#-----------------------------------------
# Laden der Datenpackagew ISwR unD DAAG aus R
rm(list=ls())
library(ISwR)
library(DAAG)
print("")
print(" correlation:")
print("-------------------------------------------------------------")
#
#options(na.action=na.exclude)
data(thuesen)
attach(thuesen)
print(str(thuesen))
# Pearson correlation
a <- cor(blood.glucose,short.velocity,use="complete.obs")
print("Pearson correlation:")
print(a)
```

```
pause()
a <- cor(thuesen,use="complete.obs")
print("Pearson correlation:")
print(a)
pause()
a <- cor.test(blood.glucose,short.velocity,use="complete.obs")
print("Pearson correlation:")
print(a)
pause()
# Spearmans rho:
a <- cor.test(blood.glucose,short.velocity,method="spearman")
print("Spearmans rho (rank correlation):")
print(a)
pause()
# Kendall's tau:
a <- cor.test(blood.glucose,short.velocity,method="kendall")
print("Kendall's tau (rank correlation):")
print(a)
pause()
print(" ")
print("---------------------- Ende -----------------------------")
print(" ")
# Ende
```

3.7 Berechnung des Stichprobenumfangs

Die Berechnung des Stichprobenumfangs sollte selbverständlicher Bestandteil der Versuchsplanung von wissenschaftlichen Experimenten sein. wei Varianten werden hier unterschieden:

a) Berechnung aus Vertrauensintervallen und Irrtumswahrscheinlichkeiten

b) dsgl. + Berücksichtigung des Fehlers 2. Art

Erläuterung:

Fehler 1. Art: Die Null-Hypothese trifft zu, aber der Test verwirft sie.

Fehler 2. Art: Die Null-Hypothese ist falsch, aber der Test nimmt sie an.

Abb. 10 zeigt eine Veranschaulichung des Fehlers 2. Art. Die vertikale Linie ist die obere Grenze eines zweiseitigen Tests auf dem 5%-Niveau. Die Abbildung wurde erstellt mit:

```
curve(pt(x,25,ncp=3),from=0, to=6)
abline(v=qt(.975,25))
```

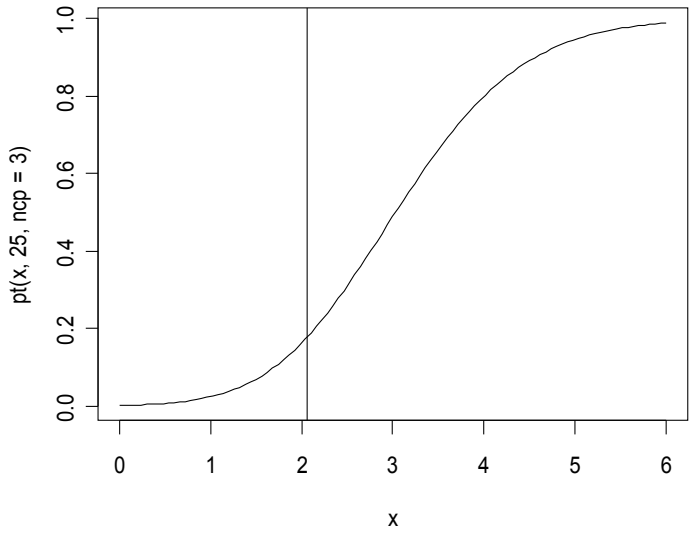

Abbildung 10: Veranschaulichung des Fehlers 2. Art. Die vertikale Linie ist die obere Grenze eines zweiseitigen Tests auf dem 5%-Niveau.

Berechnung aus Vertrauensintervallen und Irrtumswahrscheinlichkeiten:

Geg.: Z aus N(0,1)-Verteilung, $|\bar{X}-\mu|$, σ, α

Ges.: n

Lös.: aus $Z_{1-\alpha/2}=\frac{\bar{X}-\mu}{\sigma/\sqrt{n}}$ folgt $n=\left[\frac{\sigma\cdot Z_{1-\alpha/2}}{|\bar{X}-\mu|}\right]^2$

Berechnung aus Vertrauensintervall und Power:

Power = 1- β mit β: Fehler 2. Art (s.o.)

Ziel: Bestimmung des Stichprobenumfangs so, dass eine vorgegebene Differenz zweier Stichproben bei gegebener Standardabweichung, gegebenem Signifikanzniveau, und gegebener power durch einen t-Test entdeckt werden kann.

Beispiel:

```
power.t.test(delta = 0.5, sd=2, sig.level=0.01, power=0.9)
```

mit type="paired" und type="one.sample" erhält man entsprechende Tests für eine Stichprobe bzw. gepaarte Stichproben.

```
power.t.test(delta = 10, sd=14, sig.level=0.05, power=0.85,
type="paired")
```

4 Graphik

4.1 Graphiktypen in R

Übersicht

R kennt die wichtigsten Graphiktypen. Grundlage für die meisten Graphiken ist das graphics-package und das lattice-package. Hier kann nur überblicksartig auf einige Aspekte der Graphiken eingegangen werden. Das meiste lernt man auch hier aus Beipielen. Verschiedene dieser Graphikfunktionen wurden und werden im Laufe dieser Einführung schon benutzt und können als Vorlage dienen.

Graphiktypen des graphics-packages:

barplot(), boxplot(), contour(), coplot(), curve(), dotchart(), hist(), image(), mosaicplot(), pairs(), persp(), qqplot()

hinzu kommen eine Reihe von low level – Routinen:

abline(), arrows(), axis(), grid(), legend(), lines(), mtext(), plot.new(), plot.window(), points(), polygon(), pretty(), segments(), text(), title()

Graphiktypen des lattice-packages:

barchart(), bwplot(), cloud(), contourplot(), densityplot(), dotplot(), histogram(), levelplot(), piechart(), qq(), splom(), wireframe(), xyplot() und weitere.

Links

- mit ?function erhält man Hilfeinformationen;
- mit example(funame) werden Beispiele präsetiert
- demo(graphics) gibt einen Überblick über das Spektrum des Graphiksystems
- ganz nett ist auch:
  ```
  library(rgl)
  demo(rgl)
  ```
- http://addictedtor.free.fr/graphiques/thumbs.php?sort=keywords
  http://addictedtor.free.fr/graphiques/
  für Anregungen
- C:\...\R_doku\Kurzeinführung in R\R-Graphics\rgraphics.html
  enthält ebenfalls eine Sammlung von Graphikbeispielen mit R-Skripten aus dem Buch von Murrell (2005). Die Beispiele können legal aus dem Internet heruntergeladen werden.

4.2 Festlegung von Graphikeigenschaften

- Anordnung von Graphiken
  par(mfrow=c(3,2)) führt zur Aufteilung des Graphikfensters in 6 Graphiken
- Achsenbeschriftungen
  xlab="txt", ylab="txt"
- Achsenbegrenzungen

xlim, ylim

- Fonts und Linien
  type = "l", "p", "b", "n"
  lty = n
  lwd =n
  pch = n
  cex = zahl: Größe eines Punktes oder Buchstabens
- Ränder
  mar = c(2,3,2,4)
- usw.

Viele Parameter können mit par() festgelegt werden: siehe ?par

4.3 Speichern von Graphiken

- Oberfläche: bei aktivem Graphikfenster: Datei/Speichern unter
- Im Skript: Device kann gewählt werden: Bildschirm oder Datei (z.B. pdf, jpg, png, bmp, ...
  z.B.

```
pdf("testgraphik.pdf")
plot(1:10)
dev.off()
```

4.4 Einige Beispiele

- **Histogramme**

```
x <- rnorm(200)
hist(x)
```

- **Dichtefunktionen**

  das folgende Skript erzeugt Histogramme und Dichtefunktionen in 6 Varianten

```
x <- rnorm(100,6,2)
par(mfrow=c(2,3))
hist(x,main="hist(x)")
hist(x,15,prob=T,main="prob=T")
lines(density(x),lwd=2,lty=1)
hist(x,15,prob=T,main="prob=T,bw=0.1")
lines(density(x,bw=0.1),col="blue",lwd=2,lty=1)
hist(x,15,prob=T,main="prob=T,bw=1")
lines(density(x,bw=1),col="blue",lwd=2,lty=1)
hist(x,15,prob=T,main="prob=T,bw=2")
lines(density(x,bw=2),col="blue",lwd=2,lty=1)
hist(x,15,prob=T,main="prob=T,bw=\"SJ\"")
lines(density(x),bw="SJ",col="blue",lwd=2,lty=1)
```

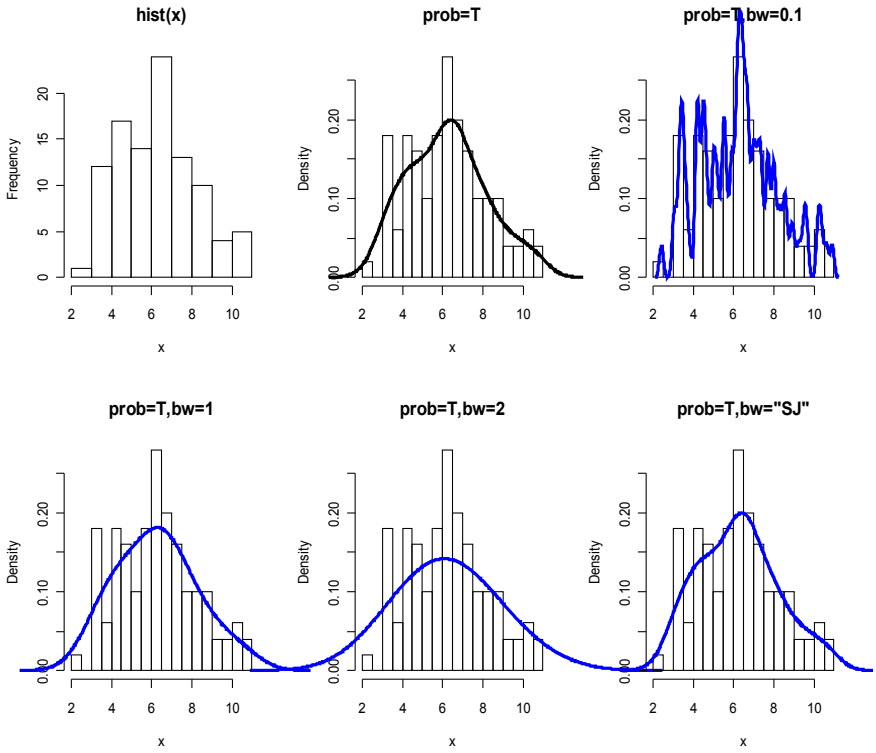

Abbildung 11: Histogramme und angepasste Dichtefunktionen

- **Boxplots**
  Boxplots gibt es in den verschiedensten Varianten. Hier ist eine mit Katzenohren.

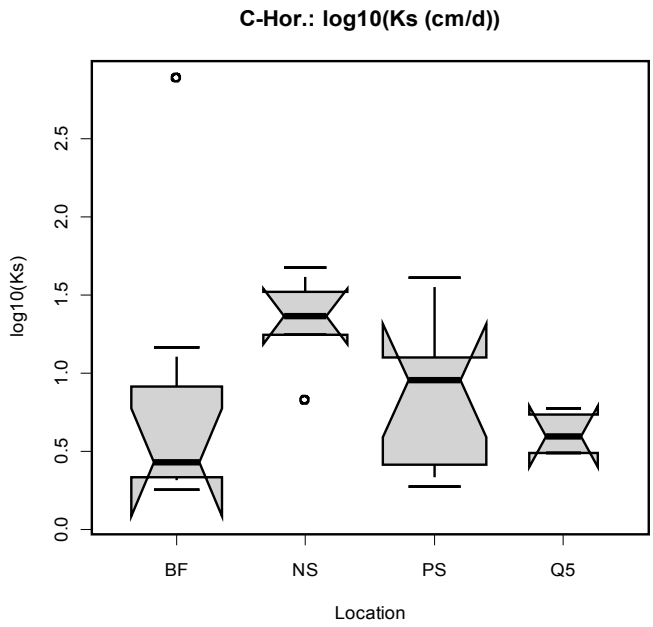

Abbildung 12: Boxplot mit notches; angezeigt werden Median, Vertrauensintervall für den Median, 1. und 3. Quartil sowie Min und Max.

```
library(DAAG)
par(mfrow=c(1,1))
datip <- read.table("KsBoxplot.dat",header=TRUE)
prts <- c(3,4,5)
dat <- datip[,prts]
cc <- complete.cases(dat)
datcc <- dat[cc,]
datok <- datcc
datok[,1] <- 1440*datok[,1]
oldpar<-par(lwd=2,cex=1.2)
d1 <- datok[datok$Hor=="A",]
boxplot(log10(Ks2)~Loc,d1,xlab="Location",ylab="log10(Ks)",ma
in="A-Hor.: log10(Ks (cm/d))", notch = TRUE, add = FALSE, col
= "lightgray",varwidth=T)
d2 <- datok[datok$Hor=="B",]
boxplot(log10(Ks2)~Loc,d2,xlab="Location",ylab="log10(Ks)",ma
in="B-Hor.: log10(Ks (cm/d))", notch = TRUE, add = FALSE, col
= "lightgray",varwidth=T)
d3 <- datok[datok$Hor=="C",]
boxplot(log10(Ks2)~Loc,d3,xlab="Location",ylab="log10(Ks)",ma
in="C-Hor.: log10(Ks (cm/d))", notch = TRUE, add = FALSE, col
= "lightgray",varwidth=T)
```

- **Scatterplots**
  wurden schon zur Genüge gezeigt.
- **Contourplots**
  Eine Serie von Contourplots mit verschiedenen Farbschemata erzeugt das folgende Skript. Beachten Sie auch die Verwendung von outer().

```
rm(list=ls())
library(DAAG)
x <- seq(-5,15,0.4)
y <- seq(1,8,0.2)
f <- function(x,y) x + y^2
z <- outer(x,y,f)
filled.contour(x,y,z,color.palette=rainbow,nlevels=50)
pause()
filled.contour(x,y,z,color.palette=terrain.colors,nlevels=50)
pause()
filled.contour(x,y,z,color.palette=heat.colors,nlevels=50)
pause()
filled.contour(x,y,z,color.palette=cm.colors,nlevels=50)
pause()
filled.contour(x,y,z,color.palette=topo.colors,nlevels=50)
```

- **Perspective-plots**
  Überblick mit `library(scatterplot3d); example(scatterplot3d)` lohnt sich!
  Ein Beispiel aus Ligges(2004):

```
rm(list=ls())
```

```
library(scatterplot3d)
data(trees)
s3d <- scatterplot3d(trees, type="h",angle=55.,scale.y=0.7,
       pch = 16, main = "trees",lwd=2)
my.lm <- with(trees,lm(Volume~Girth+Height))
s3d$plane3d(my.lm, lty.box="solid",col="red")
```

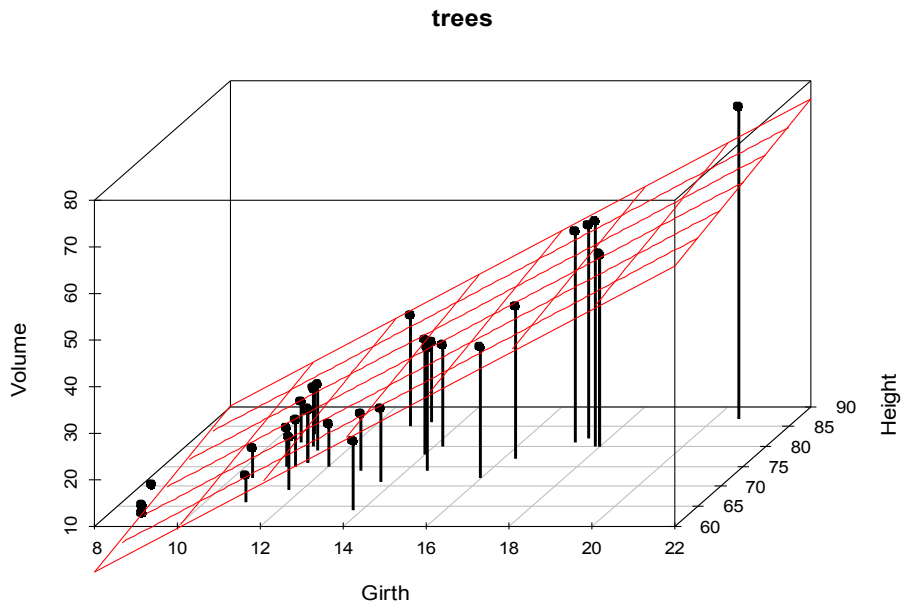

Abbildung 13: 3D-Scatterplot mit eingezeichneter Regressionsebene

- **<u>Dreieckplots</u>**
  Das Script zu untenstehendem "ternary" plot findet sich in der Datei Beispiel_17.r. Die Ausführung erfordert das Laden des packages MASS.

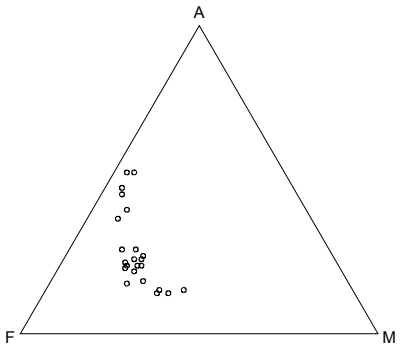

Abbildung 14: Dreiecksdiagramm zum Beispiel zur Darstellung von Bodenarten

5 Beispielskripten

Zum Schluss noch eine unsystematische Sammlung von Beispielskripten, die möglicherweise nach entsprechender Analyse und Abwandlung für das ein oder andere Problem nützlich sein könnte.

Auf die Angabe der (z.T. umfänglichen) Skripte wurde verzichtet. Angegeben wird jeweils nur der Dateiname des R-Skripts. Die zugehörigen Datendateien befinden sich im selben Verzeichnis. Der Dateiname ist im jeweiligen R-Skript zu finden. Wichtig ist die richtige Einstellung des Arbeitsverzeichnisses, da sonst die Datendatei nicht gefunden wird.

- **Deskriptive Statistik**
  Das Beispiel in Datei "Statistik_Q2_B.R" berechnet eine Scatterplot-Matrix eines Datensatzes aus einem Kleineinzugsgebiet. Es werden Korrelationen und eine Regression berechnet sowie Gütekriterien (z.B. Cook's distance) graphisch dargestellt.
- **Hauptkomponentenanalyse**
  Die Datei "Hauptkomponenten_Q2.R" berechnet eine Hauptkomponentenanalyse. Ausgegeben werden die Hauptkomponenten, die Linearkombinationen für die einzelnen Variablen sowie die Beiträge der Komponenten zur Gesamtvarianz. Graphisch wird u.a. die Auftrennung der Stichprobe durch die ersten beiden Hauptkomponenten dargestellt.
- **Clusteranalyse**
  Die Datei "Cluster_Q1Q2.R" berechnet eine Clusteranalyse anhand von Textur- und Mächtigkeitsdaten. Eingestellt ist die Euklidische Distanz und die Centroid-Methode. Ausgegeben wird ein hierarchischer Clusterungsbaum.
- **Multiple Regression**
  ist als Möglichkeit in dem Beispiel für deskriptive Statistik enthalten.
- **Empirische Variogramme**
  In Datei "Gstat_Q2_A_KsRos.R" werden empirische direktionale Variogramme für eine Schätzung der gesättigten Leitfähigkeit berechnet. Ferner werden räumliche Trends analysiert. Dargestellt werden die sog. Variogram-Cloud sowie die div. emp. Variogramme.
- **Modellanpassung**
  In "Gstat_Q2_A_KsRos_fit.R" wird ein unidirektionales empirisches Variogramm berechnet und ein sog. Gaussmodell angefittet.
- **Inverse Distanzwichtung, Kriging, geostatistische Simulation**
  In "Gstat_Q2_A_KsRos_sim_V2.R" wird die ganze geostatistische Palette durchgerechnet und graphisch dargestellt: Variogrammanalyse, Varogrammanpassung, Inverse Distanzwichtung, Kriging, Berechnung der Schätzvarianzen, unbedingte und bedingte Simulation
- **Varianzanalyse und multiple Vergleiche (Tukey-HSD)**
  In "Beispiel_18.r" wird eine zweifaktorielle Varianzanalyse berechnet und anschließend ein multipler Mittelwertvergleich mit Tukey-HSD durchgeführt.

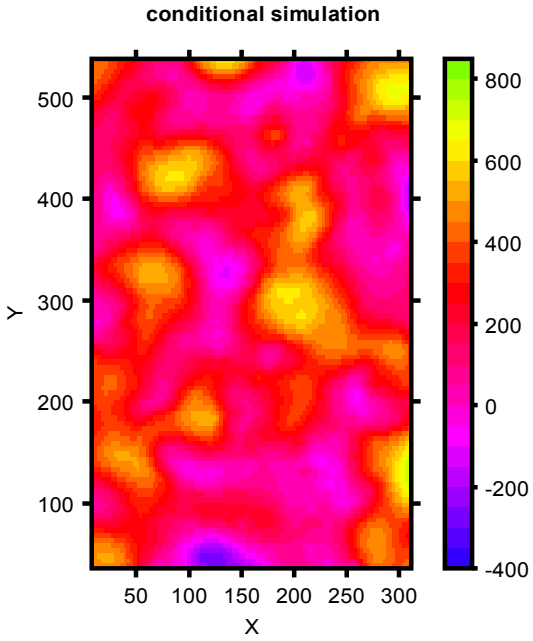

Abbildung 15: Bedingte räumliche Simulation von geschätzten hydraulischen Leitfähigkeitsdaten in einem Kleineinzugsgebiet.

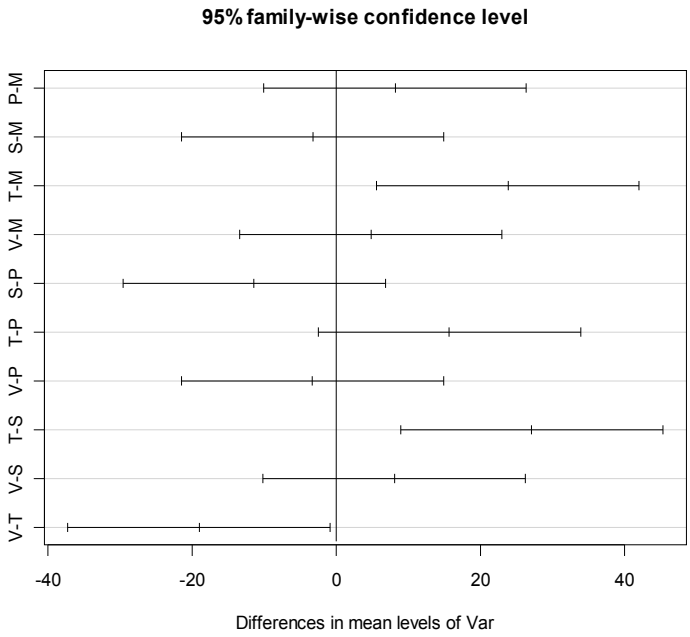

Abbildung 16: Multipler Mittelwertvergleich mit Tukey-HSD.

6 Literatur

Nachstehende Literaturliste wurde der R-homepage entnommen und ist ziemlich aktuell.

[1] Richard A. Becker, John M. Chambers, and Allan R. Wilks. The New S Language. Chapman & Hall, London, 1988.

This book is often called the `"Blue Book", and introduced what is now known as S version 2.

[2] John M. Chambers and Trevor J. Hastie. Statistical Models in S. Chapman & Hall, London,

1992.

This is also called the `"White Book", and introduced S version 3, which added structures to facilitate statistical modeling in S.

[3] John M. Chambers. Programming with Data. Springer, New York, 1998. ISBN 0-387-98503-4.

This ``Green Book" describes version 4 of S, a major revision of S designed by John Chambers to improve its usefulness at every stage of the programming process.

[4] William N. Venables and Brian D. Ripley. Modern Applied Statistics with S. Fourth Edition. Springer, New York, 2002. ISBN 0-387-95457-0.

A highly recommended book on how to do statistical data analysis using R or S-Plus. In the first chapters it gives an introduction to the S language. Then it covers a wide range of statistical methodology, including linear and generalized linear models, non-linear and smooth regression, tree-based methods, random and mixed effects, exploratory multivariate analysis, classification, survival analysis, time series analysis, spatial statistics, and optimization. The `on-line complements' available at the books homepage provide updates of the book, as well as further details of technical material.

[5] William N. Venables and Brian D. Ripley. S Programming. Springer, New York, 2000. ISBN 0-387-98966-8.

This provides an in-depth guide to writing software in the S language which forms the basis of both the commercial S-Plus and the Open Source R data analysis software systems.

[6] Deborah Nolan and Terry Speed. Stat Labs: Mathematical Statistics Through Applications. Springer Texts in Statistics. Springer, 2000. ISBN 0-387-98974-9.

Integrates theory of statistics with the practice of statistics through a collection of case studies (``labs"), and uses R to analyze the data.

[7] Jose C. Pinheiro and Douglas M. Bates. Mixed-Effects Models in S and S-Plus. Springer, 2000. ISBN 0-387-98957-0.

A comprehensive guide to the use of the `nlme' package for linear and nonlinear mixed-effects models.

[8] Frank E. Harrell. Regression Modeling Strategies, with Applications to Linear Models, Survival Analysis and Logistic Regression. Springer, 2001. ISBN 0-387-95232-2.

There are many books that are excellent sources of knowledge about individual statistical tools (survival models, general linear models, etc.), but the art of data analysis is about choosing and using multiple tools. In the words of Chatfield ``...students typically know the technical details of regression for example, but not necessarily when and how to apply it. This argues the need for a better balance in the literature and in statistical teaching between techniques and problem solving strategies." Whether analyzing risk factors, adjusting for biases in observational studies, or developing predictive models, there are common problems that few regression texts address. For example, there are missing data in the majority of datasets one is likely to encounter (other than those used in textbooks!) but most regression texts do not include methods for dealing with such data effectively, and texts on missing data do not cover regression modeling.

[9] Manuel Castejón Limas, Joaquín Ordieres Meré, Fco. Javier de Cos Juez, and Fco. Javier Martínez de Pisón Ascacibar. Control de Calidad. Metodologia para el analisis previo a la modelización de datos en procesos industriales. Fundamentos teóricos y aplicaciones con R. Servicio de Publicaciones de la Universidad de La Rioja, 2001. ISBN 84-95301-48-2.

This book, written in Spanish, is oriented to researchers interested in applying multivariate

analysis techniques to real processes. It combines the theoretical basis with applied examples coded in R.

[10] John Fox. An R and S-Plus Companion to Applied Regression. Sage Publications, Thousand Oaks, CA, USA, 2002. ISBN 0761922792.

A companion book to a text or course on applied regression (such as ``Applied Regression, Linear Models, and Related Methods'' by the same author). It introduces S, and concentrates on how to use linear and generalized-linear models in S while assuming familiarity with the statistical methodology.

[11] Peter Dalgaard. Introductory Statistics with R. Springer, 2002. ISBN 0-387-95475-9.

[12] Stefano Iacus and Guido Masarotto. Laboratorio di statistica con R. McGraw-Hill, Milano, 2003. ISBN 88-386-6084-0.

[13] John Maindonald and John Braun. Data Analysis and Graphics Using R. Cambridge University Press, Cambridge, 2003. ISBN 0-521-81336-0.

[14] Giovanni Parmigiani, Elizabeth S. Garrett, Rafael A. Irizarry, and Scott L. Zeger. The Analysis of Gene Expression Data. Springer, New York, 2003. ISBN 0-387-95577-1.

[15] Sylvie Huet, Annie Bouvier, Marie-Anne Gruet, and Emmanuel Jolivet. Statistical Tools for Nonlinear Regression. Springer, New York, 2003. ISBN 0-387-40081-8.

[16] S. Mase, T. Kamakura, M. Jimbo, and K. Kanefuji. Introduction to Data Science for engineers-Data analysis using free statistical software R (in Japanese). Suuri-Kogaku-sha, Tokyo, April 2004. ISBN 4901683128.

[17] Julian J. Faraway. Linear Models with R. Chapman & Hall/CRC, Boca Raton, FL, 2004. ISBN 1-584-88425-8.

The book focuses on the practice of regression and analysis of variance. It clearly demonstrates the different methods available and in which situations each one applies. It covers all of the standard topics, from the basics of estimation to missing data, factorial designs, and block designs, but it also includes discussion of topics, such as model uncertainty, rarely addressed in books of this type. The presentation incorporates an abundance of examples that clarify both the use of each technique and the conclusions one can draw from the results.

[18] Richard M. Heiberger and Burt Holland. Statistical Analysis and Data Display: An Intermediate Course with Examples in S-Plus, R, and SAS. Springer Texts in Statistics. Springer, 2004. ISBN 0-387-40270-5.

A contemporary presentation of statistical methods featuring 200 graphical displays for exploring data and displaying analyses. Many of the displays appear here for the first time. Discusses construction and interpretation of graphs, principles of graphical design, and relation between graphs and traditional tabular results. Can serve as a graduate-level standalone statistics text and as a reference book for researchers. In-depth discussions of regression analysis, analysis of variance, and design of experiments are followed by introductions to analysis of discrete bivariate data, nonparametrics, logistic regression, and ARIMA time series modeling. Concepts and techniques are illustrated with a variety of case studies. S-Plus, R, and SAS executable functions are provided and discussed. S functions are provided for each new graphical display format. All code, transcript and figure files are provided for readers to use as templates for their own analyses.

[19] John Verzani. Using R for Introductory Statistics. Chapman & Hall/CRC, Boca Raton, FL, 2005. ISBN 1-584-88450-9.

There are few books covering introductory statistics using R, and this book fills a gap as a true

``beginner'' book. With emphasis on data analysis and practical examples, `Using R for Introductory Statistics' encourages understanding rather than focusing on learning the underlying theory. It includes a large collection of exercises and numerous practical examples from a broad range of scientific disciplines. It comes complete with an online resource containing datasets, R functions, selected solutions to exercises, and updates to the latest features. A full solutions manual is available from Chapman & Hall/CRC.

[20] Uwe Ligges. Programmieren mit R. Springer-Verlag, Heidelberg, 2005. ISBN 3-540-20727-9, in German.

R ist eine objekt-orientierte und interpretierte Sprache und Programmierumgebung für Datenanalyse und Grafik - frei erhältlich unter der GPL. Das Buch führt in die Grundlagen der Sprache R ein und vermittelt ein umfassendes Verständnis der Sprachstruktur. Die enormen Grafikfähigkeiten von R werden detailliert beschrieben. Der Leser kann leicht eigene Methoden umsetzen, Objektklassen definieren und ganze Pakete aus Funktionen und zugehöriger Dokumentation zusammenstellen. Ob Diplomarbeit, Forschungsprojekte oder Wirtschaftsdaten, das Buch unterstützt alle, die R als flexibles Werkzeug zur Datenanalyse und -visualisierung einsetzen möchten.

[21] Fionn Murtagh. Correspondence Analysis and Data Coding with JAVA and R. Chapman & Hall/CRC, Boca Raton, FL, 2005. ISBN 1-584-88528-9.

This book provides an introduction to methods and applications of correspondence analysis, with an emphasis on data coding - the first step in correspondence analysis. It features a practical presentation of the theory with a range of applications from data mining, financial engineering, and the biosciences. Implementation of the methods is presented using JAVA and R software.

[22]Paul Murrell. R Graphics. Chapman & Hall/CRC, Boca Raton, FL, 2005. ISBN 1-584-88486-X.

A description of the core graphics features of R including: a brief introduction to R; an introduction to general R graphics features. The base graphics system of R: traditional S graphics. The power and flexibility of grid graphics. Building on top of the base or grid graphics: Trellis graphics and developing new graphics functions.

[23] Michael J. Crawley. Statistics: An Introduction using R. Wiley, 2005. ISBN 0-470-02297-3.

The book is primarily aimed at undergraduate students in medicine, engineering, economics and biology - but will also appeal to postgraduates who have not previously covered this area, or wish to switch to using R.

[24] Brian S. Everitt. An R and S-Plus Companion to Multivariate Analysis. Springer, 2005. ISBN 1-85233-882-2.

In this book the core multivariate methodology is covered along with some basic theory for each method described. The necessary R and S-Plus code is given for each analysis in the book, with any differences between the two highlighted.

[25] Richard C. Deonier, Simon Tavaré, and Michael S. Waterman. Computational Genome Analysis: An Introduction. Springer, 2005. ISBN: 0-387-98785-1.

Computational Genome Analysis: An Introduction presents the foundations of key p roblems in computational molecular biology and bioinformatics. It focuses on com putational and statistical principles applied to genomes, and introduces the mat hematics and statistics that are crucial for understanding these applications. A ll computations are done with R.

[26] Robert Gentleman, Vince Carey, Wolfgang Huber, Rafael Irizarry, and Sandrine Dudoit, editors. Bioinformatics and Computational Biology Solutions Using R and Bioconductor. Statistics

for Biology and Health. Springer, 2005. ISBN: 0-387-25146-4.

This volume's coverage is broad and ranges across most of the key capabilities of the Bioconductor project, including importation and preprocessing of high-throughput data from microarray, proteomic, and flow cytometry platforms.

[27] Terry M. Therneau and Patricia M. Grambsch. Modeling Survival Data: Extending the Cox Model. Statistics for Biology and Health. Springer, 2000. ISBN: 0-387-98784-3.

This is a book for statistical practitioners, particularly those who design and analyze studies for survival and event history data. Its goal is to extend the toolkit beyond the basic triad provided by most statistical packages: the Kaplan-Meier estimator, log-rank test, and Cox regression model.

[28] Brian Everitt and Torsten Hothorn. A Handbook of Statistical Analyses Using R. Chapman & Hall/CRC, Boca Raton, FL, 2006. ISBN 1-584-88539-4.

With emphasis on the use of R and the interpretation of results rather than the theory behind the methods, this book addresses particular statistical techniques and demonstrates how they can be applied to one or more data sets using R. The authors provide a concise introduction to R, including a summary of its most important features. They cover a variety of topics, such as simple inference, generalized linear models, multilevel models, longitudinal data, cluster analysis, principal components analysis, and discriminant analysis. With numerous figures and exercises, A Handbook of Statistical Analysis using R provides useful information for students as well as statisticians and data analysts.

[29] Julian J. Faraway. Extending Linear Models with R: Generalized Linear, Mixed Effects and Nonparametric Regression Models. Chapman & Hall/CRC, Boca Raton, FL, 2006. ISBN 1-584-88424-X.

This book surveys the techniques that grow from the regression model, presenting three extensions to that framework: generalized linear models (GLMs), mixed effect models, and nonparametric regression models. The author's treatment is thoroughly modern and covers topics that include GLM diagnostics, generalized linear mixed models, trees, and even the use of neural networks in statistics. To demonstrate the interplay of theory and practice, throughout the book the author weaves the use of the R software environment to analyze the data of real examples, providing all of the R commands necessary to reproduce the analyses.

[30] Jana Jureckova and Jan Picek. Robust Statistical Methods with R. Chapman & Hall/CRC, Boca Raton, FL, 2006. ISBN 1-584-88454-1.

This book provides a systematic treatment of robust procedures with an emphasis on practical application. The authors work from underlying mathematical tools to implementation, paying special attention to the computational aspects. They cover the whole range of robust methods, including differentiable statistical functions, distance of measures, influence functions, and asymptotic distributions, in a rigorous yet approachable manner. Highlighting hands- on problem solving, many examples and computational algorithms using the R software supplement the discussion. The book examines the characteristics of robustness, estimators of real parameter, large sample properties, and goodness-of-fit tests. It also includes a brief overview of R in an appendix for those with little experience using the software.

[31] Simon N. Wood. Generalized Additive Models: An Introduction with R. Chapman & Hall/CRC, Boca Raton, FL, 2006. ISBN 1-584-88474-6.

This book imparts a thorough understanding of the theory and practical applications of GAMs and related advanced models, enabling informed use of these very flexible tools. The author bases his approach on a framework of penalized regression splines, and builds a well- grounded

foundation through motivating chapters on linear and generalized linear models. While firmly focused on the practical aspects of GAMs, discussions include fairly full explanations of the theory underlying the methods. The treatment is rich with practical examples, and it includes an entire chapter on the analysis of real data sets using R and the author's add-on package mgcv. Each chapter includes exercises, for which complete solutions are provided in an appendix.

[32] Bernhard Pfaff. Analysis of Integrated and Cointegrated Time Series with R. Use R. Springer, 2006. ISBN 0-387-98784-3.

The book encompasses seasonal unit roots, fractional integration, coping with structural breaks, and inference in cointegrated vector autoregressive models.