

R Einführung

R Kurs Ziele

- R kennenlernen
- Die Konzepte hinter R verstehen
- Herausfinden, was Sie in R brauchen
- Einige einfache R Analysen durchführen können

What is R? (1)

- R is an integrated suite of software facilities for data manipulation, calculation and graphical display. It includes
 - an effective data handling and storage facility,
 - a suite of operators for calculations on arrays, in particular matrices,
 - a large, coherent, integrated collection of intermediate tools for data analysis,
 - graphical facilities for data analysis and display either on-screen or on hardcopy, and
 - A well-developed, simple and effective programming language which includes conditionals, loops, user-defined recursive functions and input and output facilities.

What is R? (2)

- an interface to computational procedures of many kinds;
- interactive, hands-on in real time;
- functional in its model of programming;
- object-oriented, "everything is an object";
- modular, built from standardized pieces; and,
- collaborative, a world-wide, open-source effort.

John Chambers

R installieren

- R ist plattform unabhängig
 - Windows
 - Unix
 - Macintosh
- <http://cran.r-project.org/faqs.html>

R als Rechner*

```
> 72387*12545
[1] 908094915
Komplexere Berechnungen
> log(73)-20*pi
[1] -58.54139
Aber R rundet die Darstellung der Ausgabe
> 723.87*125.45
[1] 90809.5
*Man kann mehr Genauigkeit fordern
> options(digits=15)
> 723.87*125.45
[1] 90809.4915
```

Grundlegende Konzepte in R

- (Fast) alles ist ein Objekt
- Vektoren sind sehr wichtig
- Wie in einer Sprache, kann man Kommandos kombinieren
- Funktion defaults sind oft sinnvoll, manchmal irreführend, gelegentlich schädlich

Datensatz laden

- Eingabe von der Tastatur (nicht empfohlen!)
 - `data(dataset name)` wenn in R oder einem Paket
 - `read.table` verwenden

```
bluenile6 <- read.table
("../bluenile6.txt", header=T, sep="\t", quote="")
```
- (R kann die meisten Formate lesen, cf. Paket *foreign*)

Was ist in einem Datensatz?

- `names(bluenile6)`
- `dim(bluenile6)`
- `str(bluenile6)`
- `head(bluenile6)`
- `summary(bluenile6)`
- `hist(bluenile6$price)`
- `table(bluenile6$type)`

Datensatz Objekte

- *Lists* Objekte verschiedener Typen
- *Vectors* von einem Typ (numerisch, logisch, string, ...)
- *Data.frames* Listen von Vektoren, alle der Länge n
- *Matrices* 2-d Datenstrukturen von einem Datentyp
- *Arrays* m-d Erweiterungen von Matrizen
- *Factors* Vektoren von kategoriellen Variablen

Verschiedene Objekte besitzen verschiedene Eigenschaften. Verschiedene Operationen können relevant sein.

Vektoren definieren

- `x <- c(1,2,3,7)`
- `x <- c(2,4,c(1,2,3,7))`
- `x <- rep(0,10)`
- `x <- seq(1,12,0.5)`

Und Vektorberechnungen

- `w <- y - 2*pnorm(x)`
- `ppc <- bluenile6$price/bluenile6$carat`

Zugang zu Variablen

- `bluenile6$carat`
- `bluenile6[, "carat"]`
- `bluenile6[,4]`
- **oder**
- `attach(bluenile6);carat`

Zugang zu Daten

- `v <- bluenile6[3,7]`
- `v <- bluenile6[,7]`
- `v <- bluenile6[2:4,6:8]`
- `v <- bluenile6[c(4,5:20),seq(2,10,2)]`
- `v <- bluenile6[-(1:3),-c((1:6),9)]`
- `v <- bluenile6[bluenile6$carat>1,]`

Überprüfen was Sie angegeben haben mit

`summary, head, length, dim, plot, ...`

Statistische Funktionen

- `mean(), median(), sd()`
- `min(), max(), range()`
- `var(), cov(), cor()`
- `length(), dim()`
- `quantile(), rank(), cusum()`
- `scale() ...`

Einige nützliche Funktionen

- `names, summary, table, plot`
- `apply, tapply, by, lapply, aggregate`
- `order, subset, sample, reshape, cbind, rbind`
- `is.na, unique, duplicated, cut, levels, which`
- `attributes, str`
- `if, for, repeat, while (better to use vectors)`
- `q()`

Notation

<code><-</code>	Zuordnung
<code>;</code>	Kommandotrennzeichen, <code>{ }</code> Codeblock
<code>[]</code>	array Index, <code>[[]]</code> Listenindex
<code>\$</code>	(Datensatz)\$Variable
<code>&</code>	und (<code>&&</code> gekürztes und)
<code> </code>	oder (<code> </code> gekürztes oder)
<code>==</code>	genaue Gleichheit
<code>!</code>	Nicht
<code>%*%</code>	Matrixmultiplikation
<code>" "</code>	String
<code>#</code>	Bemerkungen

Übung A Painters (1)

- 1) Den `painters` Datensatz von R laden.
- 2) Welche Variablen gibt es? Wieviele Fälle?
- 3) Wie sehen die ersten Fälle aus?
- 4) Fassen Sie den Datensatz zusammen.
- 5) Was ist der durchschnittliche `Drawing` Wert?
- 6) Erzeugen Sie eine Variable für den Unterschied zwischen den `Drawing` und `Expression` Werten.
- 7) Sind die Werte höher für `Drawing` oder `Expression`?

Übung A Painters (2)

- 1) Wieviele gibt es von jeder Schule?
- 2) Zeichnen Sie ein Histogramm der `Drawing` Werte.
- 3) Was bedeutet `mean(painters[,2:3])`?
- 4) Was bedeutet `painters[8,2]`?
- 5) Welcher Maler hat den höchsten `Drawing` Wert?
- 6) Was ist der höchste Gesamtwert?
- 7) Welcher Prozentsatz des Gesamtwerts beträgt `Drawing` im Durchschnitt?

Graphiken in R

- Einfache Graphiken
 - Histogramm, Boxplot
 - Streudiagramm
 - Säulendiagramm
- Plotfunktionen
- Multiple Plots
- Pakete (*vcd*, *grid*, *ggplot2*, *lattice*, *iplots*)

Histogramme

```
hist(carat)
```

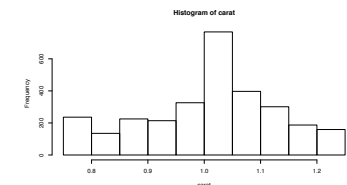
```
#als Dichte  
hist(carat, freq=FALSE)
```

```
#Wo sind die Klassengrenzen?
```

```
h1<-hist(carat, freq=FALSE)  
h1$breaks
```

```
#Klassengrenzen selbst bestimmen
```

```
hist(carat, freq=FALSE, breaks=seq(0.5, 1.3, 0.05))
```

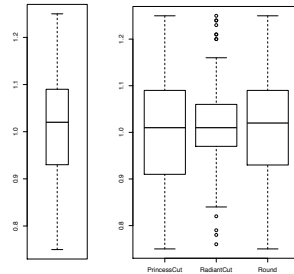


Boxplots

```
boxplot(carat)
```

```
#nach Gruppe
```

```
boxplot(carat~type)
```



Streudiagramme

```
plot(carat,price)
```

```
#Kleinere Punkte
```

```
plot(carat,price,pch=20)
```

```
#Andere Grenzen
```

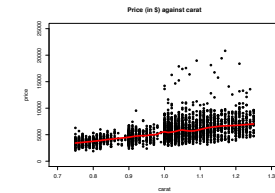
```
...xlim=c(0.7,1.3),ylim=c(0,25000)
```

```
#Mit Titel
```

```
...,main="Price (in $) against carat")
```

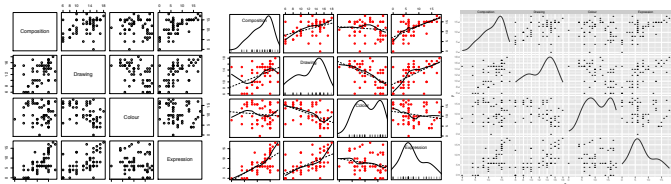
```
#Mit einer Glättung
```

```
lines(lowess(carat,price,f=0.2))
```



Streudiagramm Matrizen

- pairs
- spm im Paket car
- plotmatrix im Paket ggplot2

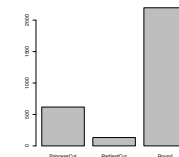


Säulendiagramme

```
#Brauchen Kategorie Häufigkeiten
```

```
table(type)
```

```
barplot(table(type))
```



```
#Reihenfolge der Kategorien ändern
```

```
neworder <- c(3,1,2)
```

```
#Der neugeordneten Variable einen Namen geben
```

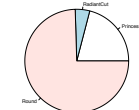
```
type2<- factor(type,levels=levels(type)
```

```
[neworder])
```

```
barplot(table(type2))
```

```
#Und wenn Sie müssen...
```

```
pie(table(type))
```

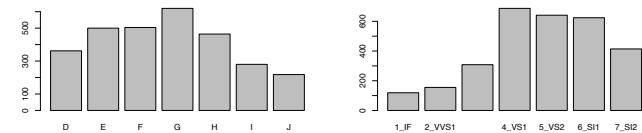


Graphische Optionen

- `xlim, ylim` z.B. `xlim=c(0,500)`
- `xlab, ylab, main` z.B. `main="Picture", legend`
- `col`, z.B. `col="red"`
- `pch` (Plotsymbol) z.B. `pch=20`
- `lty` (Linientyp), `lwd` (Linienbreite)
- `abline` (Linien hinzufügen) `abline(a,b)` oder `abline(v=1)`
- `lines` (Funktionen hinzufügen) z.B. `lines(qchisq(x,4),x)`

Multiple Plots (1)

- Neues Plotfenster z.B. `dev.new()`
- Layout setzen `par(mfrow=c(1,2))`
- Plots zeichnen, z.B.
> `barplot(table(bluenile6$clarity))`
> `barplot(table(bluenile6$color))`



Multiple Plots (2)

- Teildatensätze erstellen

```
sro<-subset(bluenile6,type=="Round")
sra<-subset(bluenile6,type=="RadiantCut")
sp<-subset(bluenile6,type=="PrincessCut")
JavaGD()
par(mfrow=c(3,1))
summary(bluenile6$price)
hist(sp$price,xlim=c(0,22000),breaks=seq(0,22000,2000))
hist(sra$price,xlim=c(0,22000),breaks=seq(0,22000,2000))
hist(sro$price,xlim=c(0,22000),breaks=seq(0,22000,2000))
```

- Oder `lattice` verwenden

```
> library(lattice)
> JavaGD()
> histogram(~bluenile6$price|bluenile6$type,layout=c(1,3))
```

Graphik Pakete

- `grid` — ein neues Bausystem für Graphiken
- `vcd` — für kategoriale Daten
- `ggplot2` — Verwirklichung von "Grammar of Graphics" einschließlich `qplot`
- `lattice` — für Trellis Plots
- `iplots` — für interaktive Graphik

Übung B Umfrage (1)

- 1) Den `survey` Datensatz in R laden.
- 2) Zeichnen Sie ein Histogramm von `Age`.
- 3) Zeichnen Sie boxplots von `Age` nach `Sex`.
- 4) Zeichnen Sie ein Streudiagramm von `Wr.Hnd` und `NW.Hnd`.
- 5) Zeichnen Sie eine Streudiagrammmatrix von den fünf stetigen Variablen.

Übung B Umfrage (2)

- 1) Zeichnen Sie `Height` gegen `Age` und fügen Sie eine Loessglättung hinzu.
- 2) Zeichnen Sie Säulendiagramme der ersten sechs kategoriellen Variablen im selben Plotfenster.
- 3) Erstellen Sie Teildatensätze für die Männer und Frauen. Zeichnen Sie Histogramme von `Height`, für Männer und Frauen getrennt.

Available Bundles and Packages

Currently, the CRAN package repository features 1798 objects including 1791 packages and 7 bundles containing 26 packages, for a total of 1817 available packages.

[A](#) [B](#) [C](#) [D](#) [E](#) [F](#) [G](#) [H](#) [I](#) [J](#) [K](#) [L](#) [M](#) [N](#) [O](#) [P](#) [Q](#) [R](#) [S](#) [T](#) [U](#) [V](#) [W](#) [X](#) [Y](#) [Z](#)

ADaCGH	Analysis of data from aCGH experiments
AER	Applied Econometrics with R
AIGIS	Areal Interpolation for GIS data
AIS	Tools to look at the data ("Ad Inidicia Spectata")
ALS	multivariate curve resolution alternating least squares (MCR-ALS)
AMORE	A MORE flexible neural network package
ARES	Allelic richness estimation, with extrapolation beyond the sample size
AcceptanceSampling	Creation and evaluation of Acceptance Sampling Plans
AdMit	Adaptive Mixture of Student-t distributions
AdaptFit	Adaptive Semiparametric Regression
AlgDesign	AlgDesign
Amelia	Amelia II: A Program for Missing Data
AnalyzefMRI	Functions for analysis of fMRI datasets stored in the ANALYZE or NIFTI format
Animal	Analyze time-coded animal behavior data

Benutzung von Paketen

- Ein für Ihre Zweckts sinnvolles Paket finden (CRAN, Task Views, suchen, ...)
- Paketwebseite lesen (ins. CRAN checks)
- Installieren und laden
- Beispiele durchlaufen, Syntax untersuchen
- Vignette lesen (falls es eine gibt)
- (Veröffentlichte Literatur durchblättern)
- Resultate überprüfen (Sind sie realistisch? Plotten)

Paket Beispiele

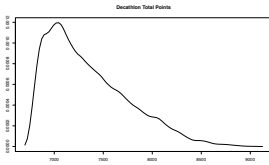
- `chron` für Kalenderberechnungen

```
> julian (7, 24, 1988)
```

```
> day.of.week (12, 25, 2009)
```

- `ash` für "average shifted histograms"

```
> plot(ash1(bin1(ZKq[, 1], nbin=100)), type="l")
```



- `amap` für das Clustern von großen Datensätzen

```
> hc <- hcluster(ZKq[, 32:41], link = "ward")
```

Pakete können gefährlich sein

- wegen
 - Fehler, Genauigkeit, Zuverlässigkeit
 - Inkonsistenz von Kommandos und Optionen
 - Überschreibung von Kommandos oder Datensätzen
 - Behandlung von Spezialfällen
 - Codequalität, Dokumentation, Hilfe

Pakete können schön sein

- weil sie liefern
 - bessere Werkzeuge (z.B., `amap` für Clustern)
 - Zugang zu modernen Methoden (z.B., `mda`, `lars`, `glasso`: Hastie, Tibshirani, Friedman)
 - mathematische Funktionen (z.B., `partitions`, `orthopolynom`)
 - Wahrscheinlichkeitsverteilungen (z.B., `actuar`, `evd`, `mvtnorm`)
 - Sondermodelle (z.B., `geoR`, `spatstat`, `TSA`)
 - Unterstützung für Anwendungen (z.B., `tm`, `fOptions`, `genetics`, `tuneR`)

Unterstützung

- ? Help
- R's Homepage: www.R-project.org
- Cran (cran.r-project.org) und BioConductor
 - Handbücher, Task Views, FAQs, Vignettes, Mailing lists, R News (jetzt R Journal), Google
- R Wiki, Rseek
- (N.B. `update.packages()`)

Hilfeseiten

- ? (Thema)

- Beschreibung
- Verwendung
- Arguments
- Einzelheiten
- Ausgabe
- Referenzen, Autor(en)
- Verwandtes
- Beispiele

```
median (stats)
Median Value
R Documentation

Description
Compute the sample median.

Usage
median(x, na.rm = FALSE)

Arguments
x an object for which a method has been defined, or a numeric vector
containing the values whose median is to be computed
na.rm logical value indicating whether NA values should be stripped before the
computation proceeds.

Details
This is a generic function for which methods can be written. However, the
default method makes use of sort and mean, both of which are generic, and so
the default method will work for most classes (e.g. matrix) for which a median
is a reasonable concept.

Value
The default method returns a length-one object of the same type as x, except
when x is a range of even length, when the result will be double.
If there are no values or if na.rm = TRUE and there are NA values the result is
NA of the same type as x (or more generally the result of is.na(x)).

References
Boker, R. A., Chambers, J. M. and Wilks, A. R. (1988) The New S Language.
Wiley-Interscience.

See Also
summary for general quantiles.

Examples
median(1:14) # 2.5 [even number]
median(c(1:3, 100, 1000)) # 3 [odd, robust]

```

Guter Stil in R

- Neuen Objekten Namen geben
- Kommandos und Optionen schrittweise überprüfen
- Defaults überprüfen
- Resultate überprüfen (plotten, Werte anschauen)
- Funktionen kombinieren (wie in einer Sprache)
- Sitzungen speichern (und mit Anmerkungen versehen)

Fortgeschrittenes R

- Coding in R
 - Makros, Funktionen, Pakete
- R und LaTeX — *Sweave*
- R und Datenbanken — *RMySQL*, *ROracle*, ...
- Große Datensätze
 - alles im Hauptspeicher
 - mehrere Kopien der Daten

Zusammenfassung

- R ist eine Sprache
- “This is R. There is no if. Only how.”
- Spielen mit R mit Datensätzen wofür Sie sich interessieren
- R Webseiten und Mailinglists verwenden