

# R für die Sozialwissenschaften - Teil 1

Jan-Philipp Kolb

29 Juli, 2017

# Einführung und Motivation

# Pluspunkte von R

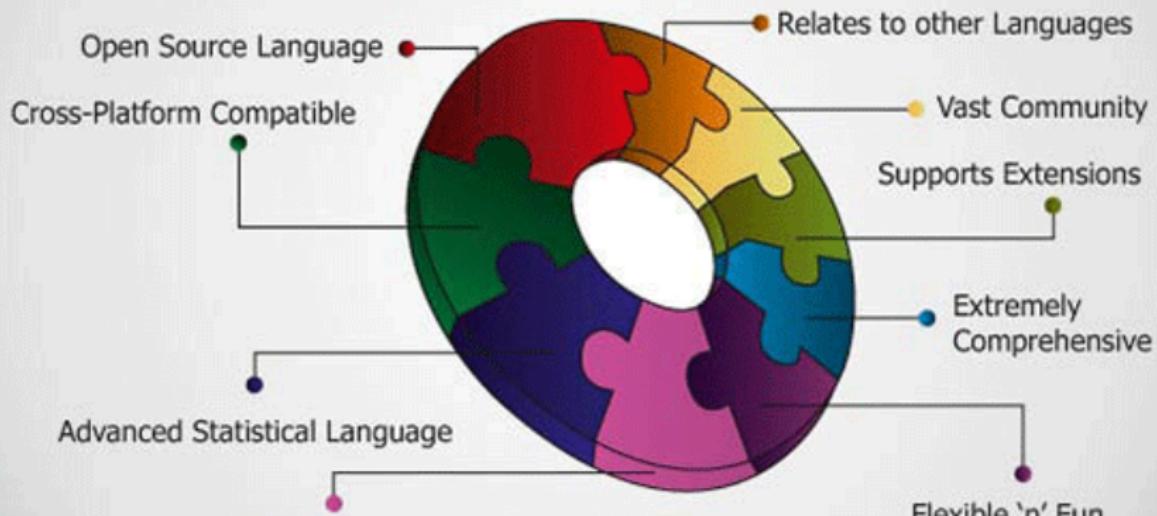
- Als Weg kreativ zu sein ...
- Graphiken, Graphiken, Graphiken
- In Kombination mit anderen Programmen nutzbar
- Zur Verbindung von Datenstrukturen
- Zum Automatisieren
- Um die Intelligenz anderer Leute zu nutzen ;-)
- ...

# Gründe

- R ist frei verfügbar. Es kann umsonst runtergeladen werden.
- R ist eine Skriptsprache / Popularität von R

## Why Learn R?

edureka!



# Ein Hauptgrund - die Community

## Revolutions

Daily news about using open source R for big data analysis, predictive modeling, data science, and visualization since 2008

[« Interactive R visuals in Power BI](#) | [Main](#) | [Because it's Friday: Mario in the Park »](#)

June 23, 2017

### The R community is one of R's best features

R is incredible software for statistics and data science. But while the bits and bytes of software are an essential component of its usefulness, software needs a **community** to be successful. And that's an area where R really shines, as Shannon Ellis explains in this [lovely ROpenSci blog post](#). For software, a thriving community offers developers, expertise, collaborators, writers and documentation, testers, agitators (to keep the community *and* software on track!), and so much more. Shannon provides links where you can find all of this in the R community:

- [\*\*#rstats hashtag\*\*](#) — a responsive, welcoming, and inclusive community of R users to interact with on Twitter
- [\*\*R-Ladies\*\*](#) — a world-wide organization focused on promoting gender diversity within the R community, with more than 30 local chapters
- [\*\*Local R meetup groups\*\*](#) — a google search may show that there's one in your area! If not, maybe consider starting one! Face-to-face meet-ups for users of all levels are incredibly valuable
- [\*\*Rweekly\*\*](#) — an incredible weekly recap of all things R

# Möglichkeiten auf dem neuesten Stand zu sein

- rweekly

R Weekly 2017-29 learnr,  
useR! 2017 Videos

Highlight

UseR! 2017

Insights

R in the Real World

[RWeekly.org](#) [Live](#) [Mail](#) [Feed](#) [Conf](#) [About](#) [All](#) [Draft](#) [Submit](#) [Night](#)

## Live

- [2017-07-22](#) ( developer.r-project.org )
- [Inter-country inequality and the World Development Indicators by @ellis2013nz](#)  
( ellisp.github.io )

- r-bloggers



[Home](#) [About](#) [RSS](#) [add your blog!](#) [Learn R](#) [R jobs](#) [Contact us](#)

WELCOME!

[Follow @rbloggers](#)

## Stan Weekly Roundup, 21 July 2017

Jan-Philippe Kolb

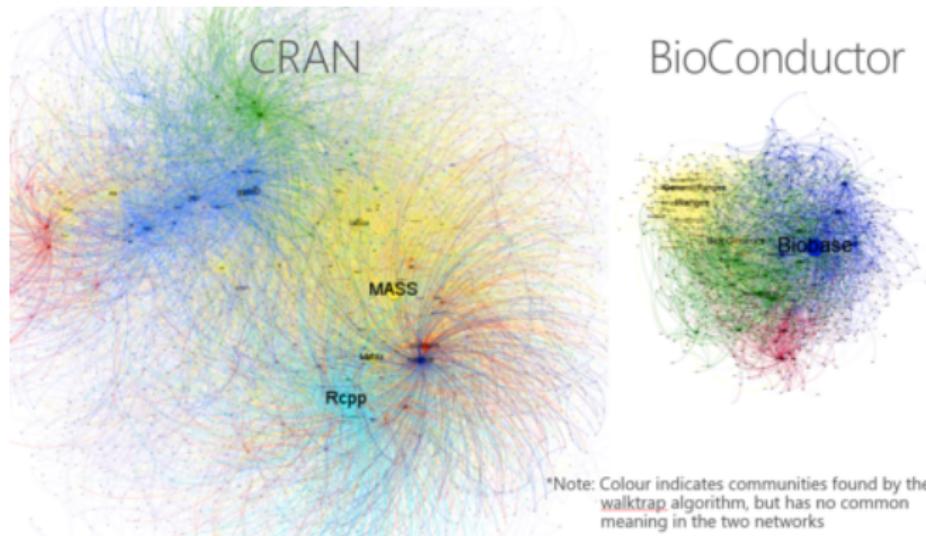
R für die Sozialwissenschaften - Teil 1

SEARCH R-BLOGGERS

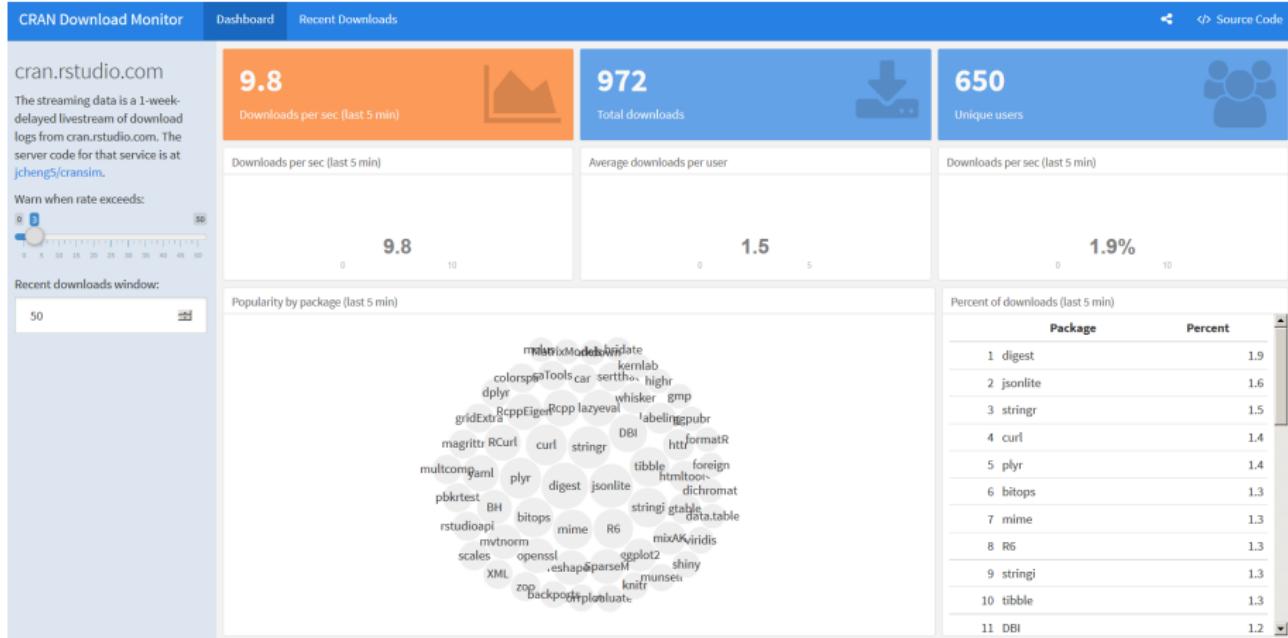
29 Juli, 2017

6 / 463

# Modularer Aufbau



# Viel genutzte Pakete



# Organisation des Kurses

- Unterlagen sind komplett auf Github hinterlegt, damit man den Kurs gleich mitverfolgen kann (mehr dazu gleich)
- Es werden viele verschiedene kleine Beispieldatensätze verwendet um spezifische Dinge zu zeigen
- Alle Funktionen in R sind mit diesen kleinen Beispielen hinterlegt
- An geeigneten Stellen verwende ich auch größere (sozialwissenschaftliche) Datensätze

# Dem Kurs folgen

- <http://japhilko.github.io/Rinter/>

## Rinter

Einführungsworkshop  
in R für  
Sozialwissenschaftler

 View On GitHub

This project is  
maintained by [Japhilko](#)

Hosted on [GitHub Pages](#)  
using the Dinky theme

## Gliederung

### Einführung

- Einführung und Motivation
- Erste Schritte mit R
- Wie bekommt man Hilfe?
- Modularer Aufbau
- Datenimport
- Datenaufbereitung
- Datenexport

### Liebe auf den ersten Plot – Grafiken und Datenanalyse mit R

- Basisgrafiken
- Datenanalyse

# Das Wiki zum Kurs

- <https://github.com/Japhilko/Rinter/wiki>

This repository Search Pull requests Issues Marketplace Gist

Japhilko / Rinter Unwatch 1 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 1 Wiki Settings Insights

**ersteSchritte**

Jan-Philipp Kolb edited this page an hour ago · 1 revision

**Den Datensatz laden**

```
## Warning: NAs introduced by coercion

library(foreign)
dat <- read.dta(
  "https://github.com/Japhilko/RSocialScience/blob/master/data/
  GPanel.dta?raw=true")
dat$bazq020a <- as.numeric(dat$bazq020a)
```

Pages 23

Find a Page...

Home

Basisgrafiken

Datenanalyse

Datenaufbereitung

# Komplette Foliensätze

Die kompletten Foliensätze kann man hier herunterladen:

[https://github.com/Japhilko/Rinter/blob/master/pdf\\_slides/  
R\\_intern.pdf](https://github.com/Japhilko/Rinter/blob/master/pdf_slides/R_intern.pdf)

# Der R-code

- Den R-code kann man direkt in die R-Konsole kopieren und ausführen.
- Begleitend zu den Folien wird meistens auch jeweils ein R-File angeboten.
- Der R-code befindet sich in folgendem Ordner:

<https://github.com/Japhilko/RInter/tree/master/code>

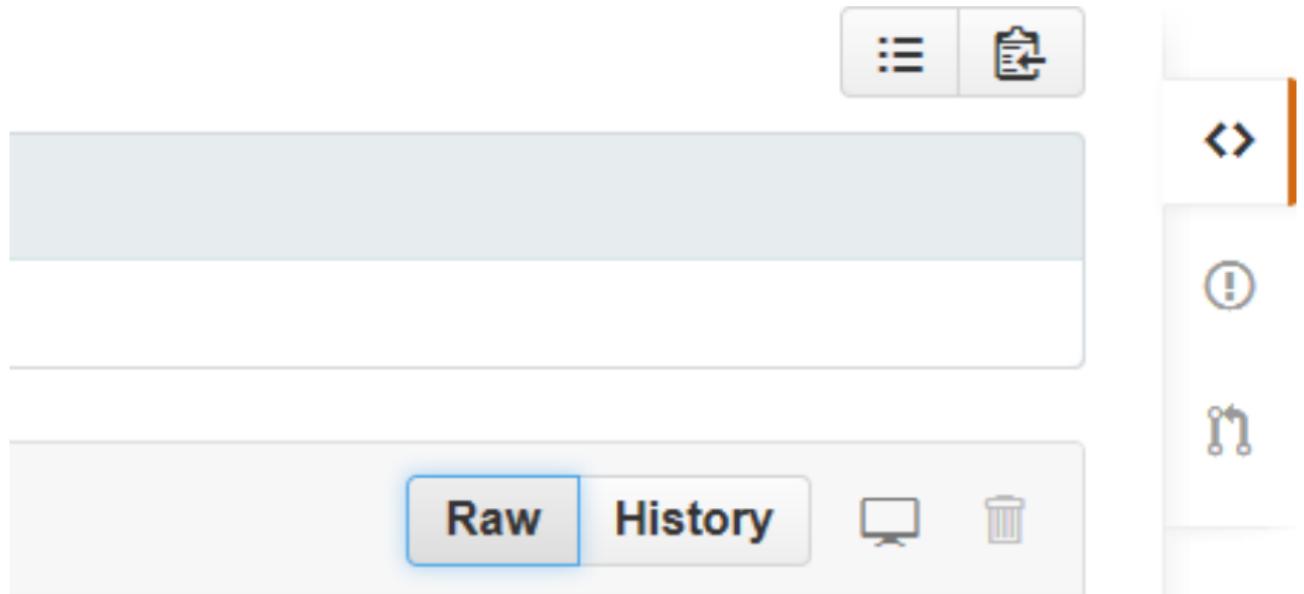
# Daten herunterladen

- Vereinzelt sind auch Datensätze vorhanden.
- .csv Dateien können direkt von R eingelesen werden (wie das geht, werde ich noch zeigen).
- Wenn die .csv Dateien heruntergeladen werden sollen - den Raw Button verwenden.
- Alle anderen Dateien (bspw. .RData) auch mittels Raw Button herunterladen.

# Ausdrucken

- Zum Ausdrucken eignen sich die pdf-Dateien am besten.
- Diese können mit dem Raw Button heruntergeladen werden.

## Raw Button bei Github



# Basis R . . .

- Wenn man nur R herunterlädt und installiert, sieht das so aus:
- So habe ich bis 2012 mit R gearbeitet.

R version 2.8.0 (2008-10-28)  
Copyright (C) 2008 The R Foundation for Statistical Computing  
ISBN 3-900051-07-8

R is free software and comes with ABSOLUTELY NO WARRANTY.  
You are welcome to redistribute it under certain conditions.  
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.  
Type 'contributors()' for more information and  
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or  
'help.start()' for an HTML browser interface to help.  
Type 'q()' to quit R.

> plot(c(1,2,3))  
>

Quartz 2 [\*]

c(1,2,3)

Index

0 1.0 1.5 2.0 2.5 3.0

R (Version 1.1-16, built: 2008-9-24)  
Working directory is /Users/Rense/Documents/RWork  
options( digits = 2 )  
Loading required package: car  
Loading required package: foreign

Attaching package: 'arm'

The following object(s) are masked from package:coda :

traceplot

> dat.stijn <- read.table("~/Users/Rense/Documents/RWork/stijndata.txt")

> names(dat.stijn) <- c("cons", "cmywave", "aantal", "resprn", "wave",  
+ "country", "sex", "agec", "agesq", "educ",  
+ "married", "cohabitend", "separate", "widow", "single",  
+ "catb", "prot", "other", "none", "attendance",  
+ "volund", "mlavtend", "gdpnormal", "gastil\_r", "cwave",  
+ "numcountry", "na1", "na2", "na3", "na4")

> model.stijn <- glmer(  
+ volund ~ sex + educ + agec + agesq + cohabit + separate + widow + single + catb +  
+ prot + other + attendance + mlavtend + gdpnormal + gastil\_r + (attendance | cwave) +  
+ (attendance | numcountry),  
+ family="binomial",  
+ data=dat.stijn)

>  
> for(i in 1:96)  
{  
+  
+ output <- HI.influence(model.stijn, "cwave", select=i, count=TRUE)  
+ save(output, file=paste("~/Users/Rense/Documents/Sociologie/Research Master/Diagnostics.ME/Influence.ME/Testing on Stijn/output ", i, ".RData", sep=""))  
+  
+ }

# ... und Rstudio

- Rstudio bietet Heute sehr viel Unterstützung:
- und macht einige Themen dieses Workshops erst möglich

The screenshot shows the RStudio interface. The top menu bar includes File, Edit, Code, View, Project, Workspace, Plots, Tools, and Help. The left sidebar has tabs for diamondPricing.R\*, formatPlot.R\*, diamonds, and Source. The main workspace shows the following R code:

```
1 library(ggplot2)
2 source("plots/formatPlot.R")
3
4 view(diamonds)
5 summary(diamonds)
6
7 summary(diamonds$price)
8 aveSize <- round(mean(diamonds$carat), 4)
9 clarity <- levels(diamonds$clarity)
10
11 p <- qplot(carat, price,
12             data=diamonds, color=clarity,
13             xlab="Carat", ylab="Price",
14             main="Diamond Pricing")
15
```

The console window at the bottom displays summary statistics for the diamonds dataset and the generated plot command.

The right panel shows the "Diamond Pricing" ggplot2 scatter plot with Price on the y-axis (0 to 15000) and Carat on the x-axis (0 to 10). The plot uses color to represent Clarity levels, with categories: I1 (red), SI2 (orange), SI1 (yellow-green), VS2 (green), VS1 (teal), VVS2 (blue), VVS1 (purple), and IF (pink).

# Aufgabe - Vorbereitung

- Prüfen Sie, ob eine Version von R auf Rechner installiert ist.
- Falls dies nicht der Fall ist, laden Sie R runter und installieren Sie R.
- Prüfen Sie, ob Rstudio installiert ist.
- Falls nicht - Installieren sie Rstudio.
- Laden Sie die R-Skripte von meinem GitHub-Account
- Erstellen Sie ein erstes Script und finden Sie das Datum mit dem Befehl `date()` und die R-version mit `sessionInfo()` heraus.

`date()`

```
## [1] "Sun Jul 23 12:13:07 2017"
```

`sessionInfo()`

```
## R version 3.3.3 (2017-03-06)
```

```
## Platform: x86_64-w64-mingw32/x64 (64-bit)
```

```
## Running under: Windows 7 x64 (build 7601) Service Pack 1
```

...

# Erste Schritte mit R

# R ist eine Objekt-orientierte Sprache

## Vektoren und Zuweisungen

- R ist eine Objekt-orientierte Sprache
- <- ist der Zuweisungsoperator (Shortcut: "Alt" + "-")

```
b <- c(1,2) # erzeugt ein Objekt mit den Zahlen 1 und 2
```

- Eine Funktion kann auf dieses Objekt angewendet werden:

```
mean(b) # berechnet den Mittelwert
```

```
## [1] 1.5
```

Mit den folgenden Funktionen können wir etwas über die Eigenschaften des Objekts lernen:

```
length(b) # b hat die Länge 2
```

```
## [1] 2
```

# Objektstruktur - Datentypen

```
str(b) # b ist ein numerischer Vektor
```

```
## num [1:2] 1 2
```

- mehr zu den möglichen Datentypen später

# Funktionen im base-Paket

Funktion	Bedeutung	Beispiel
length()	Länge	length(b)
max()	Maximum	max(b)
min()	Minimum	min(b)
sd()	Standardabweichung	sd(b)
var()	Varianz	var(b)
mean()	Mittelwert	mean(b)
median()	Median	median(b)

Diese Funktionen brauchen nur ein Argument.

# Funktionen mit mehr Argumenten

Andere Funktionen brauchen mehr:

Argument	Bedeutung	Beispiel
<code>quantile()</code>	90 % Quantile	<code>quantile(b,.9)</code>
<code>sample()</code>	Stichprobe ziehen	<code>sample(b,1)</code>

## Beispiel - Funktionen mit einem Argument

```
max(b)
```

```
## [1] 2
```

```
min(b)
```

```
## [1] 1
```

```
sd(b)
```

```
## [1] 0.7071068
```

```
var(b)
```

```
## [1] 0.5
```

# Funktionen mit einem Argument

```
mean(b)
```

```
## [1] 1.5
```

```
median(b)
```

```
## [1] 1.5
```

# Funktionen mit mehr Argumenten

```
quantile(b, .9)
```

```
## 90%
## 1.9
```

```
sample(b, 1)
```

```
## [1] 2
```

# Übersicht Befehle

<http://cran.r-project.org/doc/manuals/R-intro.html>

## An Introduction to R

### Table of Contents

#### Preface

#### 1 Introduction and preliminaries

1.1 The R environment

1.2 Related software and documentation

1.3 R and statistics

1.4 R and the window system

1.5 Using R interactively

1.6 An introductory session

1.7 Getting help with functions and features

1.8 R commands, case sensitivity, etc.

1.9 Recall and correction of previous commands

1.10 Executing commands from or diverting output to a file

1.11 Data permanency and removing objects

# Aufgabe - Zuweisungen und Funktionen

Erzeugt einen Vektor  $b$  mit den Zahlen von 1 bis 5 und berechnet...

- ① den Mittelwert
- ② die Varianz
- ③ die Standardabweichung
- ④ die quadratische Wurzel aus dem Mittelwert

# Verschiedene Datentypen

Datentyp	Beschreibung	Beispiel
numeric	ganze und reelle Zahlen	5, 3.462
logical	logische Werte	FALSE, TRUE
character	Buchstaben und Zeichenfolgen	"Hallo"

Quelle: R. Münnich und M. Knobelgespieß (2007): Einführung in das statistische Programmpaket R

# Verschiedene Datentypen

```
b <- c(1,2) # numeric  
log <- c(T,F) # logical  
char <-c("A","b") # character  
fac <- as.factor(c(1,2)) # factor
```

Mit `str()` bekommt man den Objekttyp.

```
str(fac)
```

```
## Factor w/ 2 levels "1","2": 1 2
```

## Indizieren eines Vektors:

```
A1 <- c(1,2,3,4)
```

```
A1
```

```
## [1] 1 2 3 4
```

```
A1[1]
```

```
## [1] 1
```

```
A1[4]
```

```
## [1] 4
```

```
A1[1:3]
```

```
## [1] 1 2 3
```

```
A1[-4]
```

# Logische Operatoren

```
# Ist 1 größer als 2?
```

```
1>2
```

```
## [1] FALSE
```

```
1<2
```

```
## [1] TRUE
```

```
1==2
```

```
## [1] FALSE
```

# Sequenzen

```
# Sequenz von 1 bis 10
```

```
1:10
```

```
## [1] 1 2 3 4 5 6 7 8 9 10
```

```
# das gleiche Ergebnis
```

```
seq(1,10)
```

```
## [1] 1 2 3 4 5 6 7 8 9 10
```

## Weitere Sequenzen

```
seq(-2,8,by=1.5)
```

```
## [1] -2.0 -0.5  1.0  2.5  4.0  5.5  7.0
```

```
a <- seq(3,12,length=12)
```

```
a
```

```
## [1] 3.000000 3.818182 4.636364 5.454545 6.272727 7.081818
```

```
## [8] 8.727273 9.545455 10.363636 11.181818 12.000000
```

```
b <- seq(to=5,length=12,by=0.2)
```

```
b
```

```
## [1] 2.8 3.0 3.2 3.4 3.6 3.8 4.0 4.2 4.4 4.6 4.8 5.0
```

# Reihenfolge von Argumenten

1:10

```
## [1] 1 2 3 4 5 6 7 8 9 10
```

```
seq(1,10,1)
```

```
## [1] 1 2 3 4 5 6 7 8 9 10
```

```
seq(length=10,from=1,by=1)
```

```
## [1] 1 2 3 4 5 6 7 8 9 10
```

# Wiederholungen

```
# wiederhole 10 mal
rep(1,10)

## [1] 1 1 1 1 1 1 1 1 1 1

rep("A",10)

## [1] "A" "A" "A" "A" "A" "A" "A" "A" "A" "A"
```

# Die Funktion paste

```
?paste
```

```
paste(1:4)
```

```
## [1] "1" "2" "3" "4"
```

```
paste("A", 1:6, sep = "")
```

```
## [1] "A1" "A2" "A3" "A4" "A5" "A6"
```

- Ein weiteres Beispiel:

```
paste0("A", 1:6)
```

```
## [1] "A1" "A2" "A3" "A4" "A5" "A6"
```

# Wie bekommt man Hilfe?

# Wie bekommt man Hilfe?

- Um generell Hilfe zu bekommen:

`help.start()`

- Online Dokumentation für die meisten Funktionen:

`help(name)`

- Nutze `? name` um Hilfe zu bekommen.

`?mean`

- `example(lm)` gibt ein Beispiel für die lineare Regression

`example(lm)`

# Vignetten

- Dokumente zur Veranschaulichung und Erläuterung von Funktionen im Paket

`browseVignettes()`

# Demos

- zu manchem Paketen gibt es Demonstrationen, wie der Code zu verwenden ist

```
demo()
```

```
demo(nlm)
```

# Die Funktion apropos

- sucht alles, was mit dem eingegebenen String in Verbindung steht

```
apropos("lm")
```

```
## [1] ".__C__anova.glm"      ".__C__anova.glm.null" ".__C__  
## [4] ".__C__generalMatrix" ".__C__glm"                 ".__C__g  
## [7] ".__C__glmerMod"       ".__C__lm"                  ".__C__l  
## [10] ".__C__lmerMod"        ".__C__lmList4"             ".__C__m  
## [13] ".__C__nlmerMod"       ".__C__optionalMethod" ".__T__o  
## [16] ".__T__getL:lme4"     ".colMeans"                ".lm.fit  
## [19] "colMeans"              "colMeans"                 "confint  
## [22] "contr.helmert"        "contr.Helmert"            "cv.lm"  
## [25] "CVlm"                   "dummy.coef.lm"           "getAllM  
## [28] "glm"                    "glm.control"             "glm.fit  
## [31] "glmer"                  "glmer.nb"                 "glmerCo  
## [34] "glmerLaplaceHandle"    "glmFamily"               "glmRespo  
## [37] "isGLMM"                 "isLM"                   "isNTMM"
```

# Suchmaschine für die R-Seite

```
RSiteSearch("glm")
```

# Nutzung Suchmaschinen

- Ich nutze meistens google
- Tippe:

R-project + Was ich schon immer wissen wollte

- Das funktioniert natürlich mit jeder Suchmaschine!

# Stackoverflow

- Für Fragen zum Programmieren
- Ist nicht auf R fokussiert, es gibt aber viele Diskussionen zu R
- Sehr detaillierte Diskussionen

The screenshot shows the Stackoverflow homepage. At the top, there's a navigation bar with links for Questions, Jobs, Documentation (BETA), Tags, and Users. A search bar contains the query '[r]'. On the right, there are links for Log In and Sign Up.

Below the navigation, a section titled "Tagged Questions" is shown. It includes a search bar with "[r]" and filters for info, newest, featured (selected), frequent, votes, active, and unanswered. To the right, there's a button to "Ask Question".

A main content area displays a question titled "How to make a great R reproducible example?". The question has 1776 votes, 22 answers (highlighted in green), and 147k views. It includes tags for r and r-faq. The question text discusses the importance of reproducibility in R. Below the question, there are links for community wiki, 11 revs, 8 users 54%, and Hack-R.

To the right, there's a sidebar for the "R Language" documentation, which has 22,187 frequent questions tagged. It includes a link to "about »" and a "Find a request to handle or browse 121 topics." button.

At the bottom, there's a "Related Tags" section with links for ggplot2 (x 2875), dataframe (x 1351), and plot (x 1105).

# Quick R

## Quick R

- Eine Seite mit Beispielen und Hilfe zu einem Thema
- Beispiel: Quick R - Getting Help

## Weitere Links

- Übersicht - Hilfe bekommen in R
- Eine Liste mit HowTo's
- Eine Liste der wichtigsten R-Befehle

# Ein Schummelzettel - Cheatsheet

<https://www.rstudio.com/resources/cheatsheets/>

## Base R Cheat Sheet

### Getting Help

Accessing the help files

**Print**  
Get help of a particular function  
`help(weighted.mean)`  
Search for help on a word or phrase  
`help(package = "plyr")`  
Find help for a package  
`?plyr`

**Viewing an object**

`str(x)`  
Get a summary of an object's structure  
`class(x)`  
Find the class of an object belonging to

### Using Libraries

**Install**  
`install.packages("plyr")`  
Download and install a package from CRAN

**library**  
`library(plyr)`  
Load the package into the session, making all its functions available to use

**dyn**  
`dyn(x)`  
Use a particular function from a package

**data**  
`data(x)`  
Load a built-in dataset into the environment

### Working Directory

**getwd()**  
Find the current working directory (where inputs are found and outputs are sent)

**setwd**  
`setwd("C:/Users/pk")`  
Change the current working directory

Use `projdir` in RStudio to set the working directory to the folder you are working in.

### Vectors

Creating Vectors	For Loop	While Loop
<code>c(1, 4, 9)</code> A vector <code>rep(1, 4)</code> A numeric vector <code>seq(1, 4, length=3)</code> A complex vector <code>replicate(3, 1:3)</code> Repeat a vector <code>replicate(3, matrix(1, 3))</code> Repeat a matrix	<code>for (i in 1:length(x))</code> Do something Example: <pre>for (i in 1:3) {   j &lt;- i * 10   print(j)}</pre>	<code>while (i &lt; n)</code> Do something Example: <pre>while (i &lt; 10) {   print(i)   i &lt;- i + 1}</pre>

### Selecting Vector Elements

By Position	If Statements	Functions
<code>x[4]</code> The fourth element. <code>x[1:4]</code> All but the fourth. <code>x[1:3]</code> Elements two-to-four. <code>x[-1:4]</code> All elements except first four. <code>x[0:1], 5:1</code> Elements one and five.	<code>if (x == 10)</code> Is continuous? Example: <pre>if (x == 10) {   print("Yes")} else {   print("No")}</pre>	<code>is.na(x) == FALSE</code> Is something missing? Example: <pre>is.na(is.na(x))</pre>
By Value		
<code>x[x == 10]</code> Elements which are equal to 10. <code>x[x &gt; 0]</code> All elements less than zero. <code>x[x %in% c(1, 2, 5)]</code> Elements in the set 1, 2, 5.  <code>x["apple"]</code> Element with name "apple".	<code>if (is.na(x))</code> Is something missing? Example: <pre>is.na(is.na(x))</pre>	<code>is.na(x)</code> Is something missing? Example: <pre>is.na(is.na(x))</pre>

### Reading and Writing Data

Read	Write	Description
<code>df &lt;- read.table("file.txt")</code>	<code>write.table(df, "file.txt")</code>	Read and write a delimited file
<code>df &lt;- read.csv("file.csv")</code>	<code>write.csv(df, "file.csv")</code>	Read and write a comma-separated file ("file" is a special case of most binary file formats)
<code>load("file.RData")</code>	<code>save(df, file = "file.RData")</code>	Read and write an RData file. The type specification is optional

Comparison Operators

	<code>a == b</code>	<code>!a == b</code>	<code>a &gt; b</code>	<code>a &lt; b</code>	<code>isTRUE(a == b)</code>	<code>isTRUE(!a == b)</code>	<code>isTRUE(a &gt; b)</code>	<code>isTRUE(a &lt; b)</code>
	Are equal	Are not equal	Greater than	Less than	True if a == b	True if !a == b	True if a > b	True if a < b

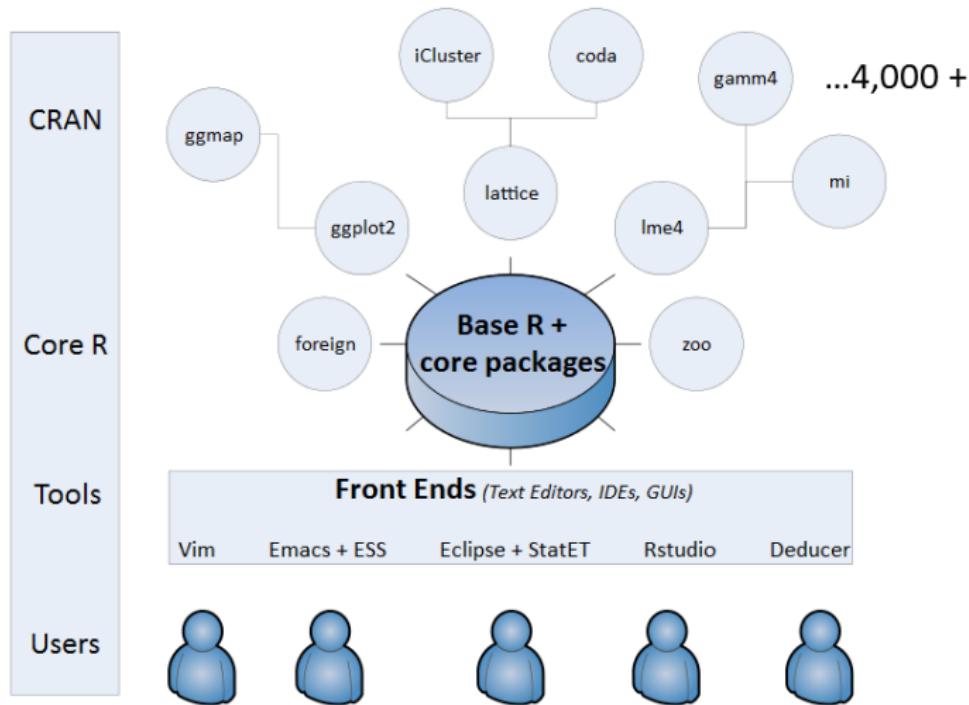
# Modularer Aufbau

# Wo sind die Routinen enthalten

```
## [1] 11007
```

- Viele Funktionen sind im Basis-R enthalten
- Viele spezifische Funktionen sind in zusätzlichen Bibliotheken integriert
- R kann modular erweitert werden durch sog. packages bzw. libraries
- Auf CRAN werden die wichtigsten packages gehostet (im Moment 11007)
- Mehr Pakete (v.a. Biostatistik, Medizin) finden sich z.B. bei bioconductor

# Übersicht R-Pakete



# Installation

```
install.packages("lme4")
```

```
library(lme4)
```

# Installation von Paketen mit RStudio

The screenshot shows the RStudio interface. The top-left pane contains a code editor with the file 'paths.R' open, showing the command `setwd("D:/Projekte/Rpackages/germanwebr/Rfunctions")`. The top-right pane is the 'Environment' view, which displays the message 'Environment is empty'. The bottom-right pane is the 'Packages' view, listing various R packages with their versions and download links. The bottom-left pane is the 'Console' view, displaying a welcome message about R being a collaborative project.

Packagename	Version	Link
AER	1.2-2	<a href="#">Download</a>
arules	1.1-2	<a href="#">Download</a>
bitops	1.0-6	<a href="#">Download</a>
boot	1.3-11	<a href="#">Download</a>
brew	1.0-6	<a href="#">Download</a>
car	2.0-19	<a href="#">Download</a>
caTools	1.17	<a href="#">Download</a>
class	7.3-10	<a href="#">Download</a>
cluster	1.15.2	<a href="#">Download</a>
codetools	0.2-8	<a href="#">Download</a>
colorspace	1.2-4	<a href="#">Download</a>
compiler	3.1.0	<a href="#">Download</a>
DAAG	1.18	<a href="#">Download</a>

# Vorhandene Pakete und Installation

The screenshot shows the RStudio interface with the 'Packages' tab selected in the top navigation bar. Below the navigation bar, there are buttons for 'Install Packages', 'Check for Updates', and a search icon. The main area displays a list of installed packages with their details:

Package	Description	Version	Status
<a href="#">AER</a>	Applied Econometrics with R	1.2-2	
<a href="#">arules</a>	Mining Association Rules and Frequent Itemsets	1.1-2	
<a href="#">bitops</a>	Bitwise Operations	1.0-6	
<a href="#">boot</a>	Bootstrap Functions (originally by Angelo Canty for S)	1.3-11	
<a href="#">brew</a>	Templating Framework for Report Generation	1.0-6	

## Übersicht viele nützliche Pakete:

- Luhmann - Tabelle mit vielen nützlichen Paketen

### Weitere interessante Pakete:

- Paket für den Import/Export - foreign
- Pakete für Survey Sampling
- xtable Paket für die Integration von Latex und R (xtable Galerie)
- Paket zur Erzeugung von Dummies
- Multivariate Normalverteilung
- Paket für Karten

# Pakete installieren

Pakete von CRAN Server installieren

```
install.packages("lme4")
```

Pakete von Bioconductor Server installieren

```
source("https://bioconductor.org/biocLite.R")
biocLite(c("GenomicFeatures", "AnnotationDbi"))
```

Pakete von Github installieren

```
install.packages("devtools")
library(devtools)
```

```
install_github("hadley/ggplot2")
```

# Wie bekomme ich einen Überblick

- Pakete entdecken, die neulich auf CRAN hochgeladen wurden
- Pakete nachschauen, die in letzter Zeit von CRAN heruntergeladen wurden
- Eine Quick-list nützlicher Pakete auf der Support Seite von Rstudio
- Computerworld hat die besten Pakete für Datenbearbeitung und Analyse aufgelistet
- Auf R-Bloggers gibt es eine Liste mit den 50 meist genutzten Pakete

# CRAN Task Views

- Zu einigen Themen sind nützliche Pakete/Funktionen in einer Übersicht zusammengestellt.
- Zur Zeit gibt es 35 Task Views
- Alle Pakete eines Task Views können mit folgendem Befehl installiert werden:

```
install.packages("ctv")
library("ctv")
install.views("Bayesian")
```

## CRAN Task Views

<a href="#">Bayesian</a>	Bayesian Inference
<a href="#">ChemPhys</a>	Chemometrics and Computational Physics
<a href="#">ClinicalTrials</a>	Clinical Trial Design, Monitoring, and Analysis
<a href="#">Cluster</a>	Cluster Analysis & Finite Mixture Models
<a href="#">DifferentialEquations</a>	Differential Equations
<a href="#">Distributions</a>	Probability Distributions
<a href="#">Econometrics</a>	Econometrics
<a href="#">Environmetrics</a>	Analysis of Ecological and Environmental Data

## Aufgabe - Zusatzpakete

Gehen Sie auf <https://cran.r-project.org/> und suchen Sie in dem Bereich, wo die Pakete vorgestellt werden, nach Paketen,...

- die für die deskriptive Datenanalyse geeignet sind.
- um Regressionen zu berechnen
- um fremde Datensätze einzulesen (z.B. SPSS-Daten)
- um mit großen Datenmengen umzugehen

# Datenimport

# Datenimport



# Dateiformate in R

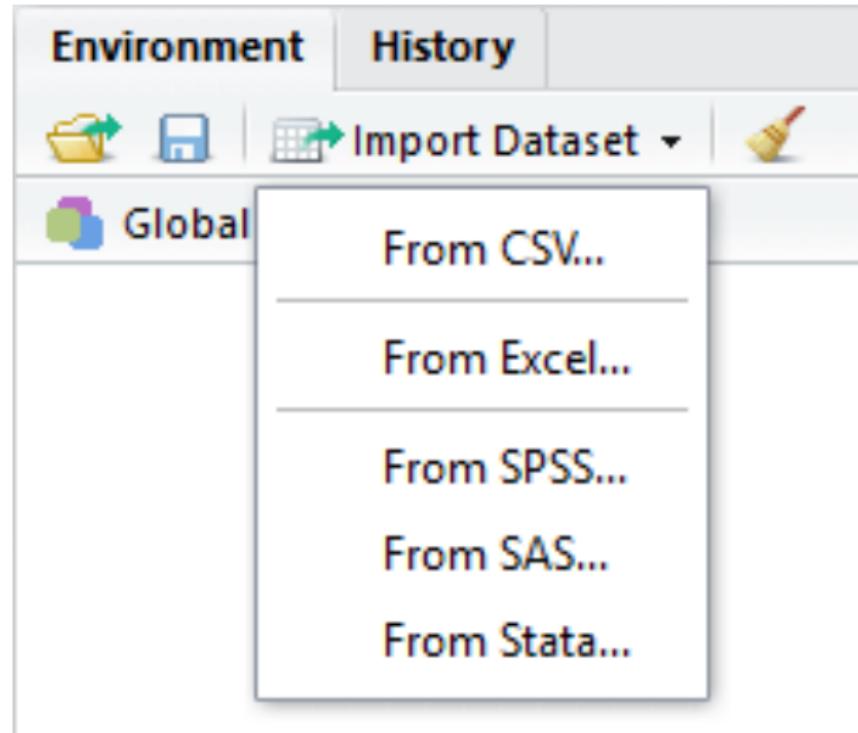
- Von R werden quelloffene, nicht-proprietary Formate bevorzugt
- Es können aber auch Formate von anderen Statistik Software Paketen eingelesen werden
- R-user speichern Objekte gerne in sog. Workspaces ab
- Auch hier jedoch gilt: (fast) alles andere ist möglich

## Formate - base package

R unterstützt von Haus aus schon einige wichtige Formate:

- CSV (Comma Separated Values): `read.csv()`
- FWF (Fixed With Format): `read.fwf()`
- Tab-getrennte Werte: `read.delim()`

# Datenimport leicht gemacht mit Rstudio



# Der Arbeitsspeicher

So findet man heraus, in welchem Verzeichnis man sich gerade befindet

`getwd()`

Und ändert dann den Pfad mit `setwd()`

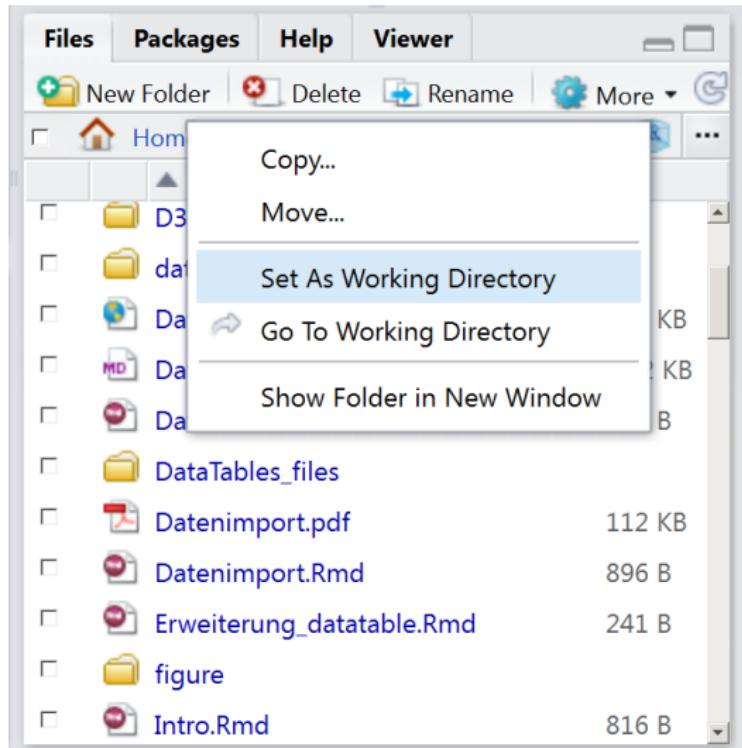
`setwd("C:/")`

Man erzeugt ein Objekt in dem man den Pfad abspeichert:

```
main.path <- "C://" # Beispiel für Windows  
main.path <- "/users/Name/" # Beispiel für Mac  
main.path <- "/home/user/" # Beispiel für Linux
```

Bei Windows ist es wichtig Slashes anstelle von Backslashes zu verwenden.

# Alternative - Arbeitsspeicher



# Import von Excel-Daten

- `library(foreign)` ist für den Import von fremden Datenformaten nötig
- Wenn Excel-Daten vorliegen - als .csv abspeichern
- Dann kann `read.csv()` genutzt werden um die Daten einzulesen.
- Bei Deutschen Daten kann es sein, dass man `read.csv2()` wegen der Komma-Separierung braucht.

```
library(foreign)
?read.csv
?read.csv2
```

# CSV Dateien einlesen

Zunächst muss das Arbeitsverzeichnis gesetzt werden, in dem sich die Daten befinden:

```
Dat <- read.csv("schuldaten_export.csv")
```

Wenn es sich um Deutsche Daten handelt:

```
Dat <- read.csv2("schuldaten_export.csv")
```

# CSV aus dem Web einladen

- Datensatz:

<https://data.montgomerycountymd.gov/api/views/6rqk-pdub/rows.csv?accessType=DOWNLOAD>

- Datenimport mit Rstudio

Import Test Data

File/Url  
<https://data.montgomerycountymd.gov/api/views/6rqk-pdub/rows.csv?accessType=DOWNLOAD>

Data Preview:

Full Name (character) ^	Gender (character) ^	Current Annual Salary (character) ^	2015 Gross Pay Received (character) ^	2015 Overtime Pay (character) ^	Department (character) ^	Department Name (character) ^	Division (character) ^	Assignment Category (character) ^	Position Title (character) ^	Underfilled Job Title (character) ^
Aarhus, Pam J.	F	\$68878.16	\$72356.79	NRA	POL	Department of Police	MSB Information Management and Technology Division	Parttime-Regular	Office Services Coordinator	NRA
Aaron, David J.	M	\$96080.09	\$101857.79	\$4640.99	POL	Department of Police	ISB Major Crimes Division Fugitive Section	Parttime-Regular	Master Police Officer	NRA
Aaron, Marsha M.	F	\$104196.06	\$103019.73	NRA	HHS	Department of Health and Human Services	Adult Protective and Case Management Services	Parttime-Regular	Social Worker IV	NRA
Ababio, Codie A.	M	\$10697.79	\$54181.46	\$4445.15	COR	Correction and Rehabilitation	PRRS Facility and Security	Parttime-Regular	Resident Supervisor II	NRA
Ababio, Essaysay	M	\$92931.00	\$93468.35	NRA	HCA	Department of Housing and Community Affairs	Single Family Housing Program	Parttime-Regular	Planning Specialist III	NRA
Abbamonte, Drew B.	M	\$67715.00	\$81392.40	\$10027.11	POL	Department of Police	PSB 6th District Special Assignment Team	Parttime-Regular	Police Officer III	NRA
Abdelmonem, Marwan M.	M	\$46228.30	\$59663.27	NRA	HHS	Department of Health and Human Services	Head Start	Parttime-Regular	Administrative Specialist II	NRA
Abdul-Chari, Nasirah J.	F	\$45828.92	\$46783.23	\$6.38	POL	Department of Police	PSB Traffic Division Automated Traffic Enforcement S...	Parttime-Regular	Police Aide	NRA
Abdullah, Sameef	M	\$61040.57	\$66861.96	\$6569.81	DGS	Department of General Services	Facilities Maintenance	Parttime-Regular	Electrician I	NRA
Abdur-Rohem, Mikael A.	M	\$56404.00	\$71943.00	\$15342.84	DOT	Department of Transportation	Transit Silver Spring Ride On	Parttime-Regular	Bus Operator	NRA
Abuze, Hirzah	F	\$151585.60	\$164945.06	NRA	HHS	Department of Health and Human Services	STD and HIV Services	Parttime-Regular	Medical Doctor III - Physician	NRA
Abuze, Zecharia S.	M	\$44825.99	\$51693.47	\$5240.75	DOT	Department of Transportation	Transit Nicholson Ride On	Parttime-Regular	Bus Operator	NRA
Abudin, Amireza	M	\$39062.00	\$4540.00	NRA	DOT	Department of Transportation	Transportation Management	Parttime-Regular	Traffic Management Technician II	Traffic Management Technicis
Abelove, Sherry R.	F	\$93436.50	\$90833.10	NRA	HHS	Department of Health and Human Services	Adult Protective and Case Management Services	Parttime-Regular	Social Worker III	NRA
Abera, Joseph M.	M	\$11781.00	\$115786.22	NRA	OTS	Department of Technology Services	EASD - ERP Applications Support	Parttime-Regular	Senior Information Technology Specialist	NRA
Abi-Jamra, Ramia F.	F	\$53009.99	\$46850.36	NRA	LIB	Department of Public Libraries	Olney Library	Parttime-Regular	Library Assistant I	NRA
Abijamra, Ryan Z.	M	\$17288.00	\$14663.33	NRA	LIB	Department of Public Libraries	Silver Spring Library	Parttime-Regular	Library Desk Assistant	NRA
Abito, Lydia B.	F	\$40429.58	\$41746.34	\$5500.53	DOT	Department of Transportation	Transit Gaithersburg Ride On	Parttime-Regular	Bus Operator	NRA
Abikarian, Maral	F	\$20925.51	\$97967.52	\$45.28	POL	Department of Police	PSB Traffic Division School Safety Section	Parttime-Regular	Crossing Guard	NRA
Abouraya, Nadia L.	F	\$16602.01	\$15592.92	NRA	HHS	Department of Health and Human Services	Community Support Network for People with Disabilities	Parttime-Regular	Office Clerk	NRA

Previewing first 50 entries.

Import Options:

Jan-Philipps Kolb

Create Preview:

R für die Sozialwissenschaften - Teil 1

29.Juli.2017

68 / 463

# Das Paket `readr`

```
install.packages("readr")
```

```
library(readr)
```

- `readr` auf dem Rstudio Blogg

# Import von Excel-Daten

- `library(readr)` ist für den Import von fremden Datenformaten hilfreich
- Wenn Excel-Daten vorliegen - als .csv abspeichern

## Beispiel Weltkulturerbestätten

```
url<-"https://raw.githubusercontent.com/Japhilko/GeoData/master/whcSites.csv"
whcSites <- read.csv(url)
```

# Der Beispieldatensatz

X	unique_number	id_no	rev_bis	name_en	name_fr	short_description_en
1	1	230	208	Rev	Cultural Landscape and Archaeological Remains of the Bam...	Paysage culturel et vestiges archéologiques de la vallée...
2	2	234	211	Rev	Minaret and Archaeological Remains of Jam	Minaret et vestiges archéologiques de Djam
3	3	1590	569	Bis	Historic Centres of Berat and Gjirokastra	Centres historiques de Berat et de Gjirokastra
4	4	1563	570	ter	Butrint	Butrint
5	5	111	102	NA	Al Qala of Beni Hammad	La Kalâ'a des Bâni Hammad

# Das Paket haven

*Import and Export 'SPSS', 'Stata' and 'SAS' Files*

```
install.packages("haven")  
  
library(haven)
```

- Das R-Paket haven auf dem Rstudio Blogg

# SPSS Dateien einlesen

- Zunächst muss wieder der Pfad zum Arbeitsverzeichnis angeben werden.
- SPSS-Dateien können auch direkt aus dem Internet geladen werden:

```
library(haven)
mtcars <- read_sav(
  "https://github.com/Japhilko/RInterfaces/raw/master/
  data/mtcars.sav")
```

## foreign kann ebenfalls zum Import genutzt werden

```
library(foreign)
link<- "http://www.statistik.at/web_de/static/
mz_2013_sds_-_datensatz_080469.sav"

?read.spss
Dat <- read.spss(link,to.data.frame=T)
```

# stata Dateien einlesen

```
library(haven)
oecd <- read_dta("https://github.com/Japhilko/IntroR/
                  raw/master/2017/data/oecd.dta")
```

- Einführung in Import mit R (is.R)

# Das Paket `readstata13`

```
# install.packages("readstata13")  
  
library(readstata13)  
?read.dta13
```

# Das Paket rio

```
install.packages("rio")  
  
library("rio")  
x <- import("mtcars.csv")  
y <- import("mtcars.rds")  
z <- import("mtcars.dta")
```

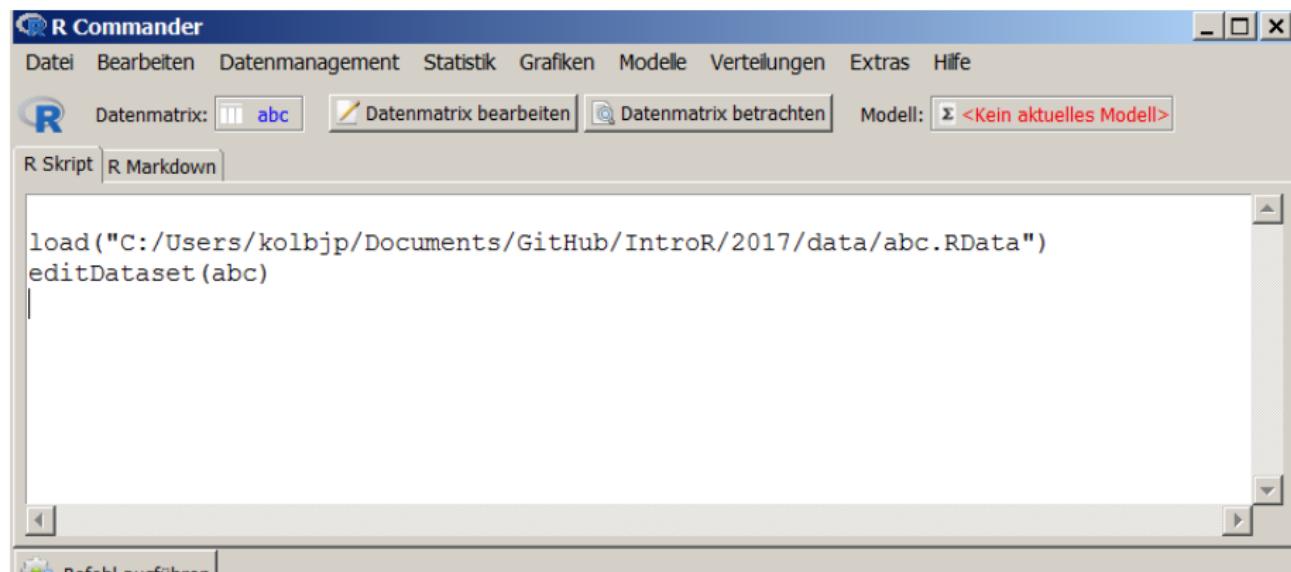
- rio: A Swiss-Army Knife for Data I/O

## Weitere Alternative Rcmdr

```
install.packages("Rcmdr")
```

- Funktioniert auch mit Rstudio

```
library(Rcmdr)
```



## Die Gesis Panel Daten



---

gesis



- Gesis Panel Campus File
- Codebuch für die Gesis Panel Daten im DBK

# Aufgabe

- Gehen Sie auf meine Github Seite

<https://github.com/Japhilko/RSocialScience/tree/master/data>

- Importieren Sie den Datensatz GPanel.dta

# Datenaufbereitung

# Data Frames

Beispieldaten einlesen:

```
library(foreign)
dat<-read.dta("https://github.com/Japhilko/RSocialScience/
               raw/master/data/GPanel.dta")
```

# Übersicht mittels Rstudio

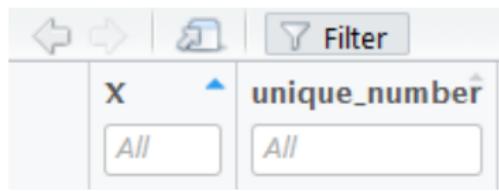
The screenshot shows the RStudio interface with the 'Environment' tab selected. The top bar includes tabs for Environment, History, Plots, and Git, along with various icons for file operations like Import Dataset and a search bar.

The main area displays the 'Data' section, which lists three data objects:

Object	Description	Action
Chem97	31022 obs. of 8 variables	View
dat	100 obs. of 23 variables	View
Daten	100 obs. of 23 variables	View

# Den Datensatz anschauen

Die Daten filtern



# Daten filtern

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
	All	[...]	All	All	All
14	4.3	2	3.6	0.1	setosa
9	4.4	2.0	4.0	0.2	setosa
39	4.4	3.0	1.3	0.2	setosa
43	4.4	3.2	1.3	0.2	setosa
42	4.5	2.3	1.3	0.3	setosa
4	4.6	3.1	1.5	0.2	setosa
7	4.6	3.4	1.4	0.3	setosa
23	4.6	3.6	1.0	0.2	setosa
48	4.6	3.2	1.4	0.2	setosa

Showing 1 to 9 of 135 entries (filtered from 150 total entries)

# Struktur des Datensatzes

Der Datentyp

```
typeof(dat)
```

```
## [1] "list"
```

Die Funktion `glimpse` im Paket `dplyr`

```
library(dplyr)
```

```
glimpse(dat)
```

```
## Observations: 100
```

```
## Variables: 23
```

```
## $ a11c019a <fctr> Eher zufrieden, Sehr zufrieden, Eher zufri
```

```
## $ a11c020a <fctr> Weder zufrieden noch unzufrieden, Eher un
```

```
## $ a11c021a <fctr> Sehr zufrieden, Eher unzufrieden, Eher un
```

```
## $ a11c022a <fctr> Stimme eher nicht zu, Stimme eher nicht zu
```

```
## $ a11c023a <fctr> Stimme eher zu, Stimme eher zu, Stimme eh
```

# In Dataframe übertragen

Die Vektoren (von dat) zu einem data.frame verbinden:

```
Daten <- data.frame(dat)
```

Anzahl der Zeilen/Spalten herausfinden

```
nrow(Daten) # Zeilen
```

```
## [1] 100
```

```
ncol(Daten) # Spalten
```

```
## [1] 23
```

# Die Daten in der Console anschauen

Die ersten Zeilen anschauen

`head(Daten)`

Die letzten Zeilen anschauen

`tail(Daten)`

# Indizieren

Indizieren eines dataframe:

```
Daten[1,1]
```

```
## [1] Eher zufrieden
```

```
## 7 Levels: Item nonresponse Sehr zufrieden ... Weiß nicht
```

```
Daten[2,]
```

```
##           a11c019a           a11c020a           a11c021a
```

```
## 2 Sehr zufrieden Eher unzufrieden Eher unzufrieden Stimme e
```

```
##           a11c023a           a11c024a
```

```
## 2 Stimme eher zu Stimme voll und ganz zu Eher niedrigeren I
```

```
##           a11c026a           a11c027a a11c028a a1
```

```
## 2 Mehrmals die Woche Mindestens einmal im Monat Täglich T
```

```
##           a11c031a           a11c032a a11c033a
```

```
## 2 Mindestens einmal im Monat Mehrmals im Monat Seltener
```

```
##
```

# Operatoren um Subset für Datensatz zu bekommen

Diese Operatoren eignen sich gut um Datensätze einzuschränken

```
Dauer <- as.numeric(Daten$bazq020a)
```

```
## Warning: NAs durch Umwandlung erzeugt
```

```
head(Daten[Dauer>20,])
```

```
##           a11c019a           a11c020a           a11c021a
## 2    Sehr zufrieden   Eher unzufrieden   Eher unzufrieden Stimme
## 3    Eher zufrieden    Sehr zufrieden   Eher unzufrieden Stimme
## 5    Eher zufrieden    Eher zufrieden    Eher zufrieden
## NA          <NA>          <NA>          <NA>
## 9    Sehr zufrieden    Eher zufrieden   Sehr zufrieden
## 15   Sehr zufrieden    Sehr zufrieden   Sehr zufrieden
##           a11c023a           a11c024a
## 2      Stimme eher zu Stimme voll und ganz zu
## 3  Stimme eher nicht zu Stimme eher zu
```

# Einschränken mit dem Paket tidyverse

- einfacher geht es mit dem Paket tidyverse

```
library(tidyverse)
filter(Daten, Dauer>20)
```

##	a11c019a	
## 1	Sehr zufrieden	Eher unzufrieden
## 2	Eher zufrieden	Sehr zufrieden
## 3	Eher zufrieden	Eher zufrieden
## 4	Sehr zufrieden	Eher unzufrieden
## 5	Sehr zufrieden	Sehr zufrieden
## 6	Eher zufrieden	Sehr zufrieden
## 7	Sehr zufrieden	Sehr zufrieden
## 8	Sehr zufrieden	Sehr zufrieden
## 9	Eher zufrieden	Eher zufrieden
## 10	Sehr zufrieden	Eher unzufrieden
## 11	Eher zufrieden	Eher zufrieden

# Datensätze einschränken

```
SEX <- Daten$a11d054a
```

```
Daten[SEX=="Männlich",]
```

```
##                                     a11c019a  
## 1          Eher zufrieden Weder zufrieden noch unzufrieden  
## 2          Sehr zufrieden Eher unzufrieden  
## 3          Eher zufrieden Sehr zufrieden  
## 4          Eher zufrieden Sehr zufrieden  
## 5          Eher zufrieden Eher zufrieden  
## 7          Eher zufrieden Eher zufrieden  
## 9          Sehr zufrieden Eher zufrieden  
## 12         Sehr zufrieden Eher zufrieden  
## 15         Sehr zufrieden Sehr zufrieden  
## 16         Sehr zufrieden Sehr zufrieden  
## 17 Weder zufrieden noch unzufrieden Eher unzufrieden
```

# Weitere wichtige Optionen

# Ergebnis in ein Objekt speichern

```
subDat <- Daten[Dauer>20,]
```

# mehrere Bedingungen können mit

# & verknüpft werden:

```
Daten[Dauer>18 & SEX=="Männlich",]
```

##	a11c019a	
## 2	Sehr zufrieden	Eher unzufrieden
## 3	Eher zufrieden	Sehr zufrieden
## 5	Eher zufrieden	Eher unzufrieden
## 9	Sehr zufrieden	Eher unzufrieden
## 15	Sehr zufrieden	Sehr zufrieden
## 18	Eher zufrieden	Sehr zufrieden
## 20	Eher zufrieden	Eher unzufrieden
## 29	Sehr zufrieden	Sehr zufrieden
## 41	Sehr zufrieden	Eher unzufrieden

# Die Nutzung einer Sequenz

Daten[1:3,]

```
##           a11c019a                  a11c020a      a11c021a  
## 1 Eher zufrieden Weder zufrieden noch unzufrieden Sehr zufrieden  
## 2 Sehr zufrieden                      Eher unzufrieden Eher unzufrieden  
## 3 Eher zufrieden                      Sehr zufrieden Eher unzufrieden  
##           a11c022a                  a11c023a      a11c024a  
## 1 Stimme eher nicht zu      Stimme eher zu      Stimme eher zu  
## 2 Stimme eher nicht zu      Stimme eher zu      Stimme voll und ganz zu  
## 3 Stimme eher nicht zu      Stimme eher nicht zu     Stimme voll und ganz zu  
##           a11c025a                  a11c026a      a11c027a  
## 1 Eher niedrigeren Lebensstandard      Seltener  
## 2 Eher niedrigeren Lebensstandard Mehrmals die Woche  
## 3 Denselben Lebensstandard            Täglich  
##           a11c028a a11c029a      a11c030a  
## 1 Mehrmals die Woche   Täglich   Täglich
```

## Variablen Labels

```
library(foreign)
dat<-read.dta("https://github.com/Japhilko/RSocialScience
               /blob/master/data/GPanel.dta?raw=true")

attributes(dat)

var.labels <- attr(dat,"var.labels")
```

Genauso funktioniert es auch mit dem Paket haven

```
library(haven)
dat2 <- read_dta(
  "https://github.com/Japhilko/RSocialScience/
  blob/master/data/GPanel.dta?raw=true")

var.labels2 <- attr(dat,"var.labels")
```

# Die Spaltennamen umbenennen

Mit `colnames` bekommt man die Spaltennamen angezeigt

```
colnames(dat)
```

```
## [1] "a11c019a" "a11c020a" "a11c021a" "a11c022a" "a11c023a"  
## [7] "a11c025a" "a11c026a" "a11c027a" "a11c028a" "a11c029a"  
## [13] "a11c031a" "a11c032a" "a11c033a" "a11c034a" "bazq020a"  
## [19] "a11d056z" "a11d092a" "a11c100a" "a11c111a" "a11c109a"
```

So kann man die Spaltennamen umbenennen:

```
colnames(dat) <- var.labels
```

Analog geht das für die Reihennamen

```
rownames(dat)
```

```
## [1] "1"    "2"    "3"    "4"    "5"    "6"    "7"    "8"    "9"  
## [12] "12"   "13"   "14"   "15"   "16"   "17"   "18"   "19"   "20"  
## [20] "20"   "20"   "20"   "20"   "20"   "20"   "20"   "20"   "20"   "21"
```

# Indizieren

- Das Dollarzeichen kann man auch nutzen um einzelne Spalten anzusprechen

```
head(dat$a11c019a)
```

```
## [1] Eher zufrieden Sehr zufrieden Eher zufrieden Eher zufrieden  
## [5] Eher zufrieden Sehr zufrieden  
## 7 Levels: Item nonresponse Sehr zufrieden ... Weiß nicht
```

```
dat$a11c019a[1:10]
```

```
## [1] Eher zufrieden Sehr zufrieden Eher zufrieden Eher zufrieden  
## [5] Eher zufrieden Sehr zufrieden Eher zufrieden Eher zufrieden  
## [9] Sehr zufrieden Sehr zufrieden  
## 7 Levels: Item nonresponse Sehr zufrieden ... Weiß nicht
```

# Auf Spalten zugreifen

- Wie bereits beschrieben kann man auch Zahlen nutzen um auf die Spalten zuzugreifen

```
head(dat[, 1])
```

```
head(dat[, "a11c019a"]) # liefert das gleiche Ergebnis
```

# Exkurs - Labels wie verwenden

## *Tools for Working with Categorical Variables (Factors)*

```
library("forcats")
```

- fct\_collapse - um Faktorlevel zusammenzufassen
- fct\_count - um die Einträge in einem Faktor zu zählen
- fct\_drop - Unbenutzte Levels raus nehmen

# Rekodieren

```
library(car)

head(dat$a11c020a)

## [1] Weder zufrieden noch unzufrieden Eher unzufrieden
## [3] Sehr zufrieden                      Sehr zufrieden
## [5] Eher zufrieden                      Eher zufrieden
## 7 Levels: Item nonresponse Sehr zufrieden ... Weiß nicht

head(recode(dat$a11c020a, "'Eher unzufrieden'='A';else='B'"))

## [1] B A B B B B
## Levels: A B
```

# Das Paket tibble

```
install.packages("tibble")

library(tibble)
gpanel1 <- as_tibble(dat)
gpanel1

## # A tibble: 100 × 23
##       a11c019a      a11c020a
## * <fctr>          <fctr>
## 1 Eher zufrieden Weder zufrieden noch unzufrieden Sehr zufrieden
## 2 Sehr zufrieden                         Eher unzufrieden Eher unzufrieden
## 3 Eher zufrieden                         Sehr zufrieden   Eher unzufrieden
## 4 Eher zufrieden                         Sehr zufrieden   Eher unzufrieden
## 5 Eher zufrieden                         Eher zufrieden   Eher unzufrieden
## 6 Sehr zufrieden                         Eher zufrieden   Eher unzufrieden
## 7 Eher zufrieden                         Eher zufrieden   Eher unzufrieden
## 8 Eher zufrieden                         Eher zufrieden   Eher unzufrieden
## 9 Eher zufrieden                         Eher zufrieden   Eher unzufrieden
## 10 Eher zufrieden                        Eher zufrieden   Eher unzufrieden
```

# Schleifen

```
erg <- vector()  
  
for (i in 1:ncol(dat)){  
  erg[i] <- length(table(dat[,i]))  
}  
}
```

# Fehlende Werte ausschließen

- Mathe-Funktionen haben in der Regel einen Weg, um fehlende Werte in ihren Berechnungen auszuschließen.
- `mean()`, `median()`, `colSums()`, `var()`, `sd()`, `min()` und `'max()'` all take the `na.rm` argument.

# Fehlende Werte umkodieren

```
Daten$bazq020a[Daten$bazq020a==99] <- NA
```

- Quick-R zu fehlenden Werten
- Fehlende Werte rekodieren

# Mit Strings arbeiten

```
gsub("l","L","Hallo Welt")
```

```
## [1] "HaLLo WeLt"
```

- Natural Language Processing - Tutorial auf der UseR 2017

## Weitere Links

- Das googleVis Paket für einen besseren Überblick
- Tidy data - das Paket `tidyverse`
- Die `tidyverse` Sammlung
- Data wrangling with R and RStudio

# Datenexport

# Die Exportformate von R

- In R werden offene Dateiformate bevorzugt
- Genauso wie `read.X()` Funktionen stehen viele `write.X()` Funktionen zur Verfügung
- Das eigene Format von R sind sog. Workspaces (`.RData`)

# Beispieldatensatz erzeugen

```
A <- c(1,2,3,4)  
B <- c("A","B","C","D")  
  
mydata <- data.frame(A,B)
```

		A    B
1		A
2		B
3		C
4		D

# Überblick Daten Import/Export

- wenn mit R weitergearbeitet wird, eignet sich das .RData Format am Besten:

```
save(mydata, file="mydata.RData")
```

- Der Datensatz kann dann mit load wieder eingelesen werden

```
load("mydata.RData")
```

## Daten in .csv Format abspeichern

```
write.csv(mydata, file="mydata.csv")
```

- Wenn mit Deutschem Excel weitergearbeitet werden soll, eignet sich write.csv2 besser

```
write.csv2(mydata, file="mydata.csv")
```

- Sonst sieht das Ergebnis so aus:

	A
1	,"A","B"
2	1,1,"A"
3	2,2,"B"
4	3,3,"C"
5	4,4,"D"
6	

# Das Paket xlsx

## R xlsx package : A quick start guide to manipulate Excel files in R

AdChoices

Microsoft Excel

Download for Java

Excel Tutorial



- Install and load xlsx package
- Read an Excel file
- Write data to an Excel file

```
library(xlsx)
write.xlsx(mydata, file="mydata.xlsx")
```

# Reading/Writing Stata (.dta) files with Foreign

December 4, 2012

By `is.R()`

- Funktionen im Paket `foreign`

## R topics documented:

<code>lookup.xport</code> . . . . .	2
<code>read.arff</code> . . . . .	3
<code>read.dbf</code> . . . . .	4
<code>read.dta</code> . . . . .	5
<code>read.epiinfo</code> . . . . .	7
<code>read.mtp</code> . . . . .	8
<code>read.octave</code> . . . . .	9
<code>read.spss</code> . . . . .	10
<code>read.ssdi</code> . . . . .	12

# Daten in stata Format abspeichern

```
library(foreign)
write.dta(mydata,file="data/mydata.dta")
```

# Das Paket rio

```
install.packages("rio")
```

## Import, Export, and Convert Data Files

The idea behind `rio` is to simplify the process of importing data into R and exporting data from R. This process is, probably unnecessarily, extremely complex for beginning R users. Indeed, R supplies [an entire manual](#) describing the process of data import/export. And, despite all of that text, most of the packages described are (to varying degrees) out-of-date. Faster, simpler, packages with fewer dependencies have been created for many of the file types described in that document. `rio` aims to unify data I/O (importing and exporting) into two simple functions: `import()` and `export()` so that beginners (and experienced R users) never have to think twice (or even once) about the best way to read and write R data.

# Daten als .sav abspeichern (SPSS)

```
library("rio")
# create file to convert

export(mtcars, "data/mtcars.sav")
```

# Dateiformate konvertieren

```
export(mtcars, "data/mtcars.dta")  
  
# convert Stata to SPSS  
convert("data/mtcars.dta", "data/mtcars.sav")
```

# Links Export

- Quick R für das Exportieren von Daten:
- Hilfe zum Export auf dem cran Server
- Daten aus R heraus bekommen

# Basisgrafiken

# Ein Plot sagt mehr als 1000 Worte

- Grafisch gestützte Datenanalyse ist toll
- Gute Plots können zu einem besseren Verständnis beitragen
- Einen Plot zu generieren geht schnell
- Einen guten Plot zu machen kann sehr lange dauern
- Mit R Plots zu generieren macht Spaß
- Mit R erstellte Plots haben hohe Qualität
- Fast jeder Plottyp wird von R unterstützt
- R kennt eine große Menge an Exportformaten für Grafiken

# Plot ist nicht gleich Plot

- Bereits das base Package bringt eine große Menge von Plot Funktionen mit
- Das lattice Packet erweitert dessen Funktionalität
- Eine weit über diese Einführung hinausgehende Übersicht findet sich in Murrell, P (2006): R Graphics.

# Task View zu Thema Graphiken

CRAN Task View: Graphic Displays & Dynamic Graphics & Graphic Devices & Visualization

**Maintainer:** Nicholas Lewin-Koh

**Contact:** nikko at hailmail.net

**Version:** 2015-01-07

**URL:** <https://CRAN.R-project.org/view=Graphics>

R is rich with facilities for creating and developing interesting graphics. Base R contains functionality for many plot types including coplots, mosaic plots, biplots, and the list goes on. There are devices such as postscript, png, jpeg and pdf for outputting graphics as well as device drivers for all platforms running R. [lattice](#) and grid are supplied with R's recommended packages and are included in every binary distribution. [lattice](#) is an R implementation of William Cleveland's trellis graphics, while grid defines a much more flexible graphics environment than the base R graphics.

# Datensatz

```
library(mlmRev)  
data(Chem97)
```

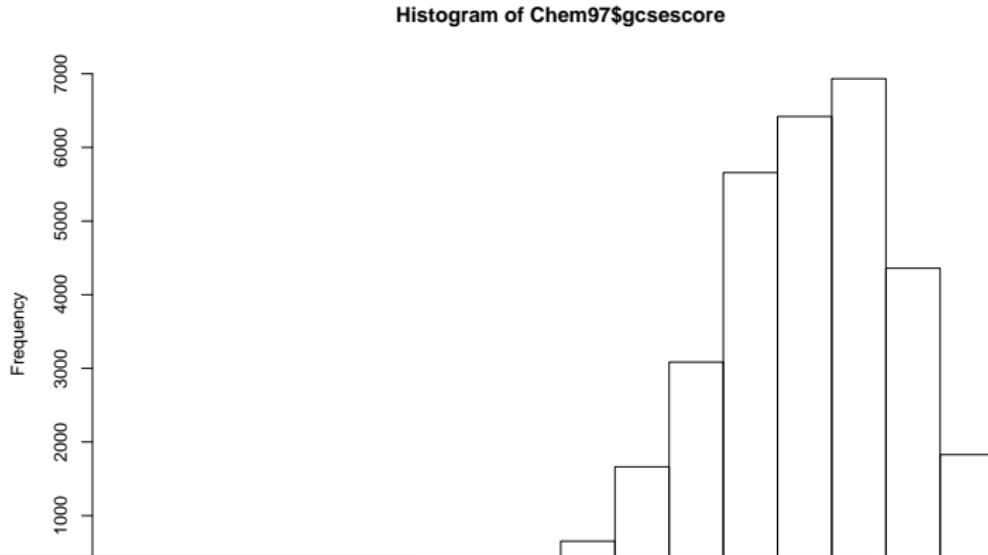
- [lea] Local Education Authority - a factor
- [school] School identifier - a factor
- [student] Student identifier - a factor
- [score] Point score on A-level Chemistry in 1997
- [gender] Student's gender
- [age] Age in month, centred at 222 months or 18.5 years
- [gcsescore] Average GCSE score of individual.
- [gcsecnt] Average GCSE score of individual, centered at mean.

# Histogramm - Die Funktion hist()

Wir erstellen ein Histogramm der Variable gcsescore:

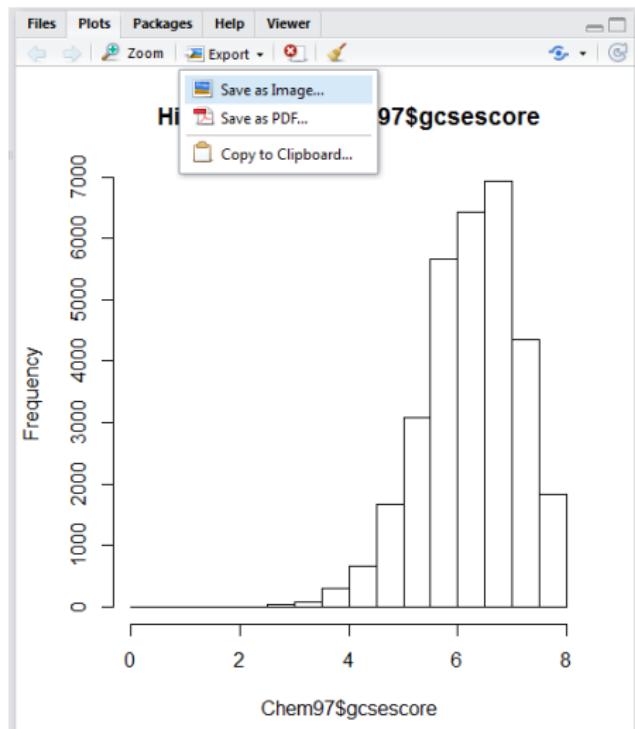
?hist

`hist(Chem97$gcsescore)`



# Graphik speichern

- Mit dem button Export in Rstudio kann man die Grafik speichern.



# Befehl um Graphik zu speichern

- Alternativ auch bspw. mit den Befehlen png, pdf oder jpeg

```
png("Histogramm.png")
hist(Chem97$gcsescore)
dev.off()
```

# Histogramme

- Die Funktion `hist()` plottet ein Histogramm der Daten
- Der Funktion muss mindestens ein Beobachtungsvektor übergeben werden
- `hist()` hat noch sehr viel mehr Argumente, die alle (sinnvolle) default values haben

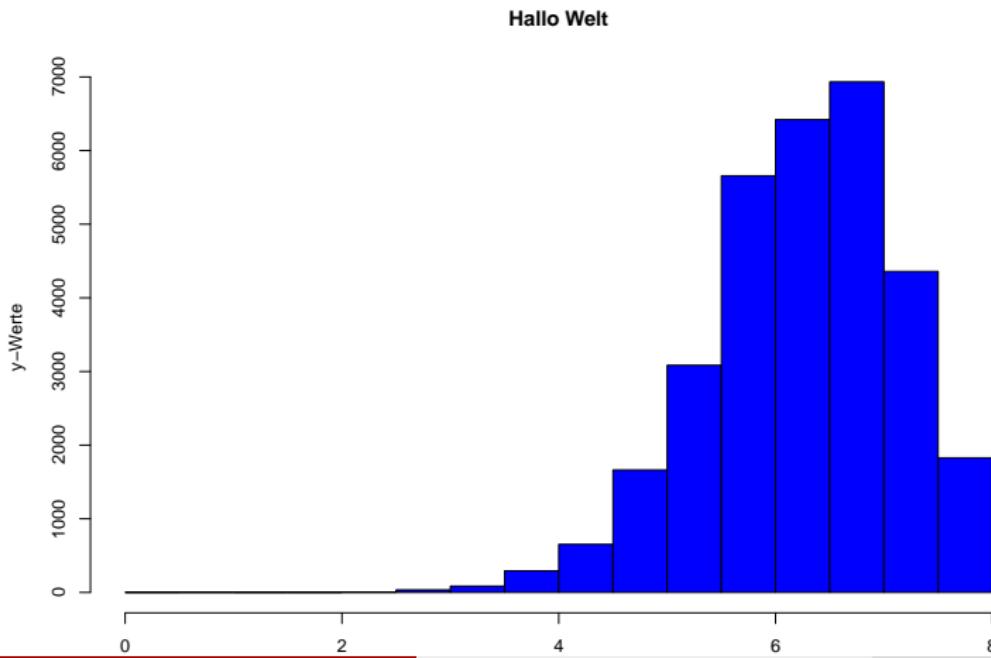
---

Argument	Bedeutung	Beispiel
main	Überschrift	main="Hallo Welt"
xlab	x-Achsenbeschriftung	xlab="x-Werte"
ylab	y-Achsenbeschriftung	ylab="y-Werte"
col	Farbe	col="blue"

---

# Histogramm

```
hist(Chem97$gcsescore, col="blue",  
main="Hallo Welt", ylab="y-Werte", xlab="x-Werte")
```



# Barplot

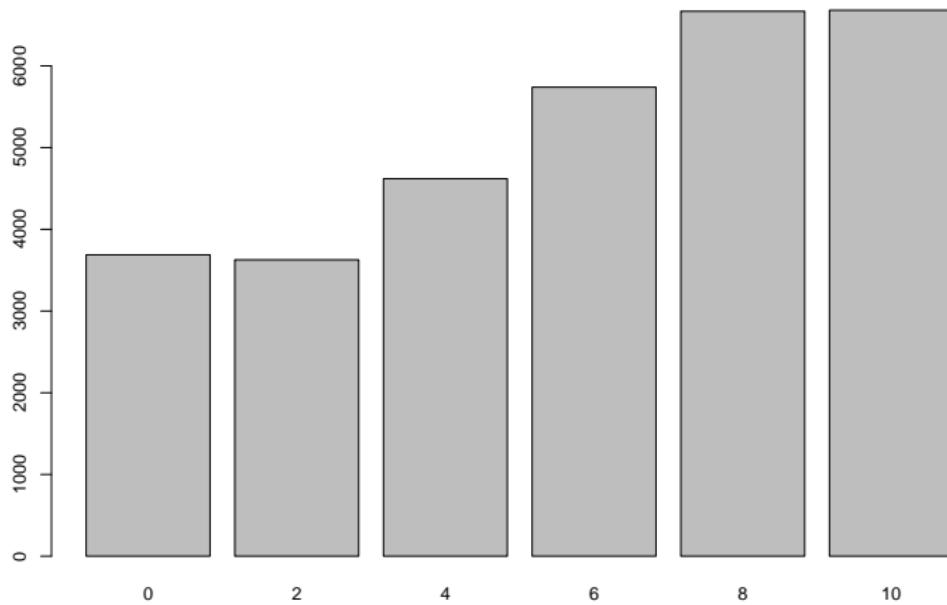
- Die Funktion `barplot()` erzeugt aus einer Häufigkeitstabelle einen Barplot
- Ist das übergebene Tabellen-Objekt zweidimensional wird ein bedingter Barplot erstellt

```
tabScore <- table(Chem97$score)
```

```
barplot(tabScore)
```

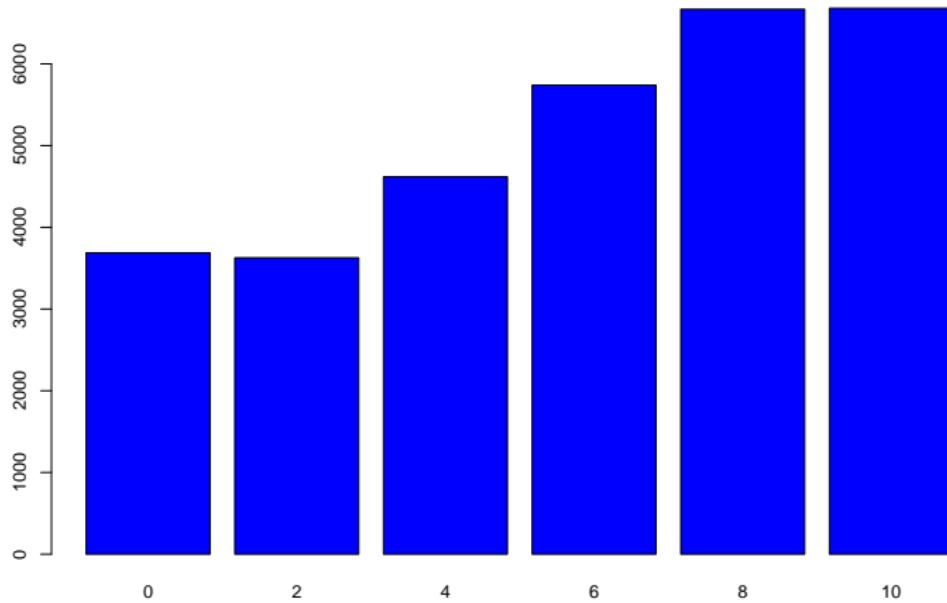
# Barplots und barcharts

```
barplot(tabScore)
```



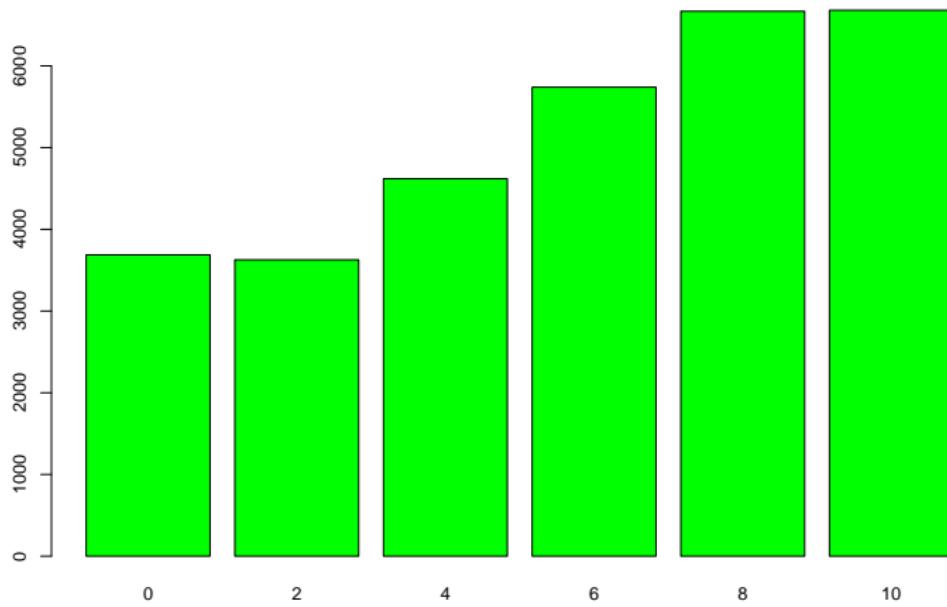
## Mehr Farben:

```
barplot(tabScore, col=rgb(0,0,1))
```



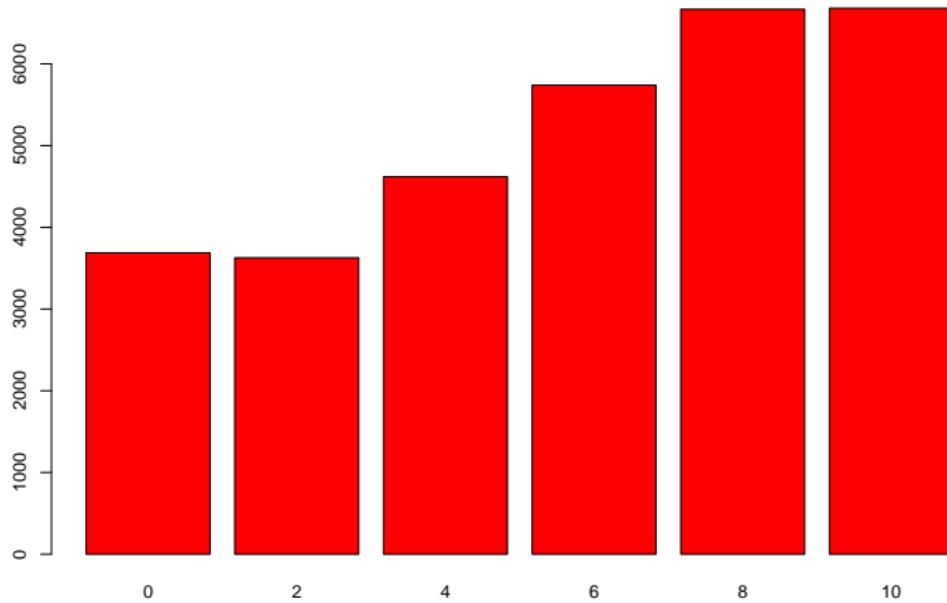
# Grüne Farbe

```
barplot(tabScore, col=rgb(0,1,0))
```



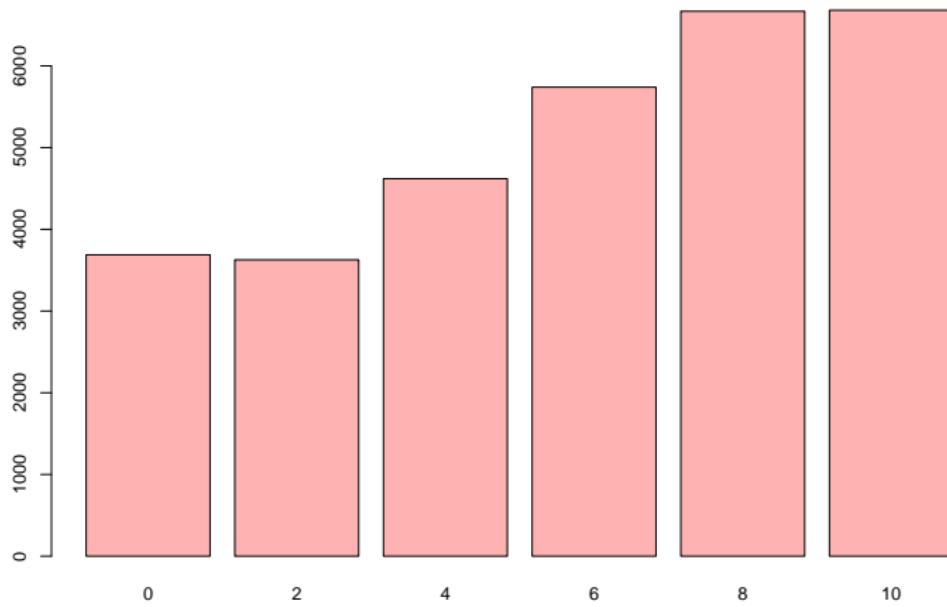
# Rote Farbe

```
barplot(tabScore, col=rgb(1,0,0))
```



# Transparent

```
barplot(tabScore, col=rgb(1,0,0,.3))
```



# Scatterplots

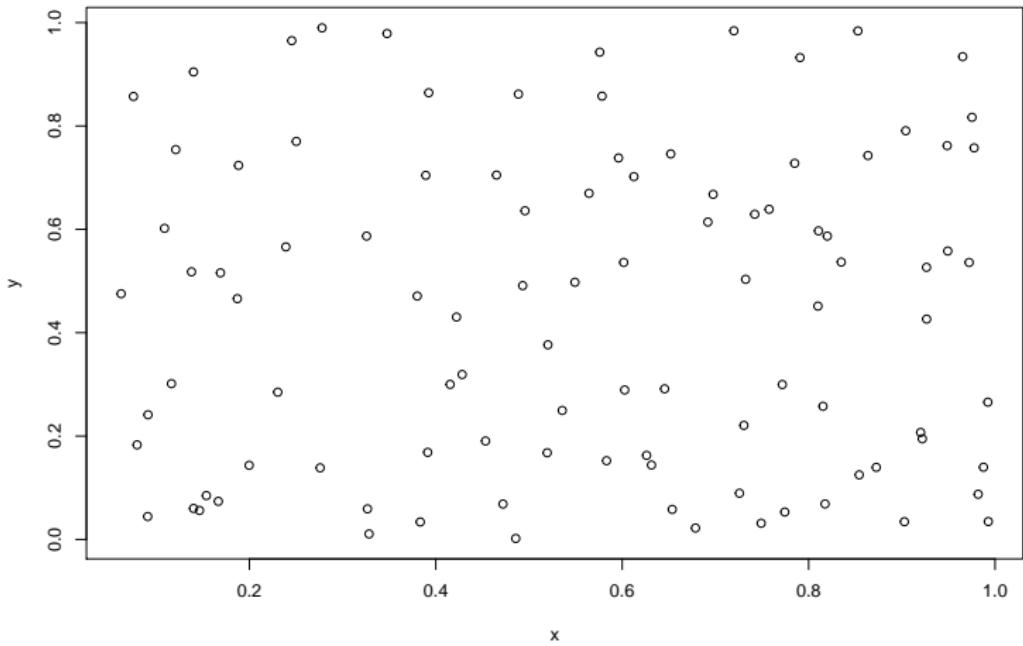
- Ein einfacher two-way Scatterplot kann mit der Funktion `plot()` erstellt werden
- `plot()` muss mindestens ein `x` und ein `y` Beobachtungsvektor übergeben werden
- Um die Farbe der Plot-Symbole anzupassen gibt es die Option `col` (Farbe als character oder numerisch)
- Die Plot-Symbole selbst können mit `pch` (plotting character) angepasst werden (character oder numerisch)
- Die Achsenbeschriftungen (`labels`) werden mit `xlab` und `ylab` definiert

# Beispieldaten für Scatterplot

```
x <- runif(100)  
y <- runif(100)
```

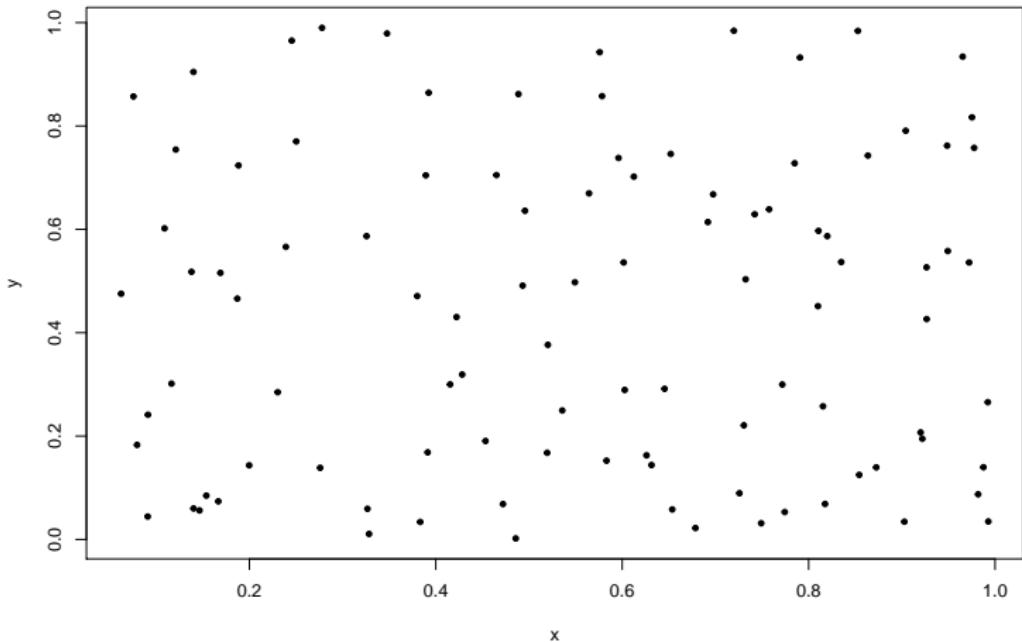
# Einfacher Scatterplot

`plot(x, y)`



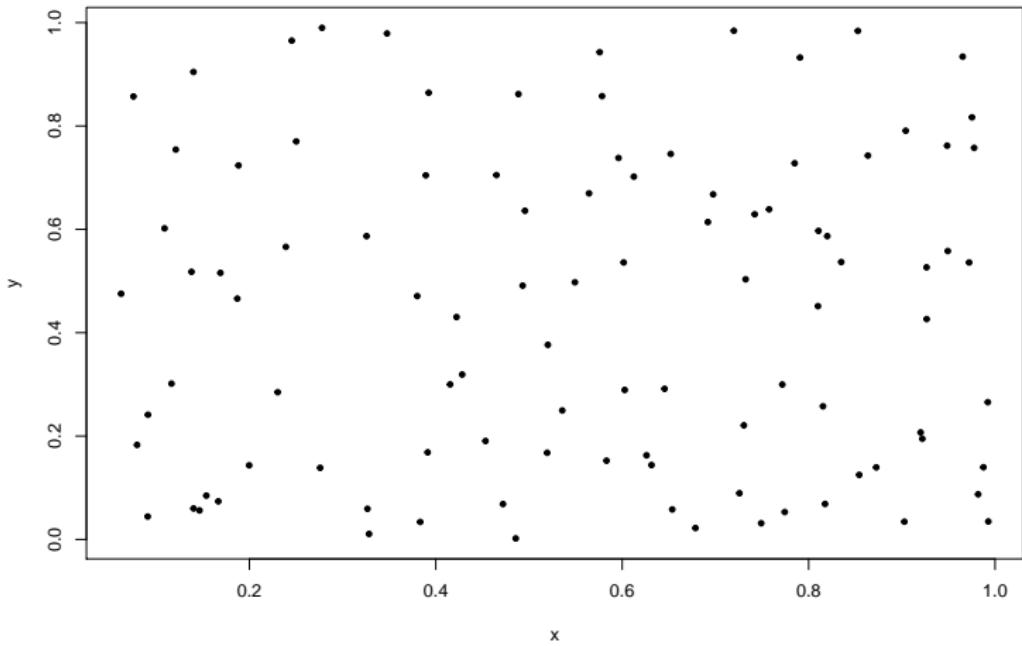
# Einfacher Scatterplot II

```
plot(x,y,pch=20)
```



# Einfacher Scatterplot III

```
plot(x,y,pch=20)
```



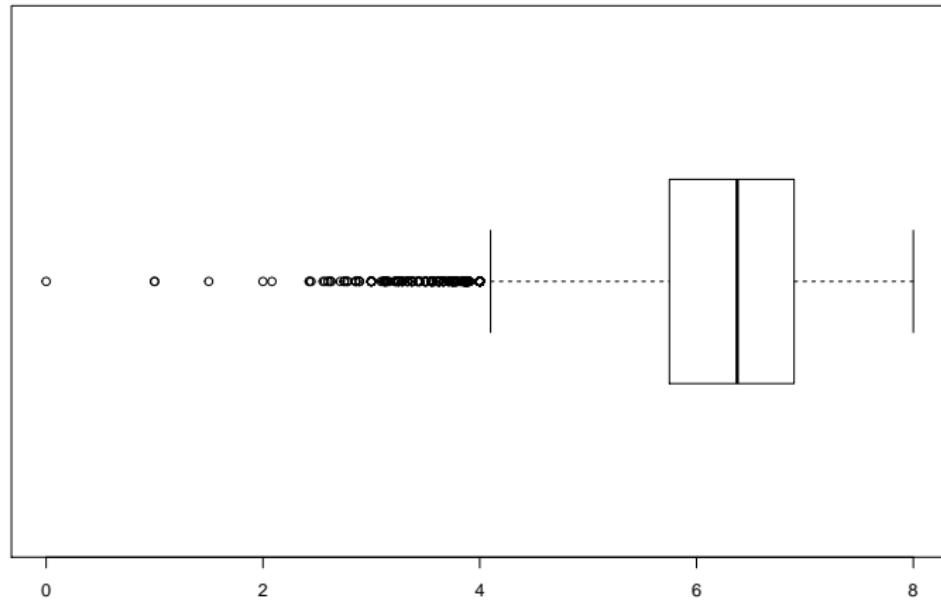
# Boxplot

- Einen einfachen Boxplot erstellt man mit `boxplot()`
- Auch `boxplot()` muss mindestens ein Beobachtungsvektor übergeben werden

?`boxplot`

# Horizontaler Boxplot

```
boxplot(Chem97$gcsescore,  
horizontal=TRUE)
```

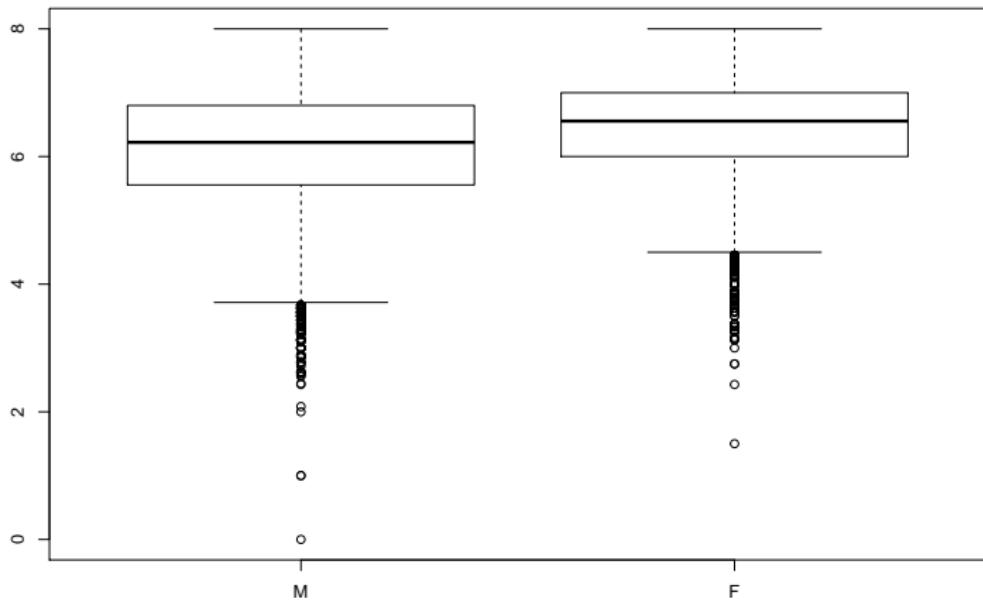


# Gruppierte Boxplots

- Ein sehr einfacher Weg, einen ersten Eindruck über bedingte Verteilungen zu bekommen ist über sog. Gruppierte notched Boxplots
- Dazu muss der Funktion `boxplot()` ein sog. Formel-Objekt übergeben werden
- Die bedingende Variable steht dabei auf der rechten Seite einer Tilde

# Beispiel grupierter Boxplot

```
boxplot(Chem97$gcsescore~Chem97$gender)
```



# Alternativen zu Boxplot

## Violinplot

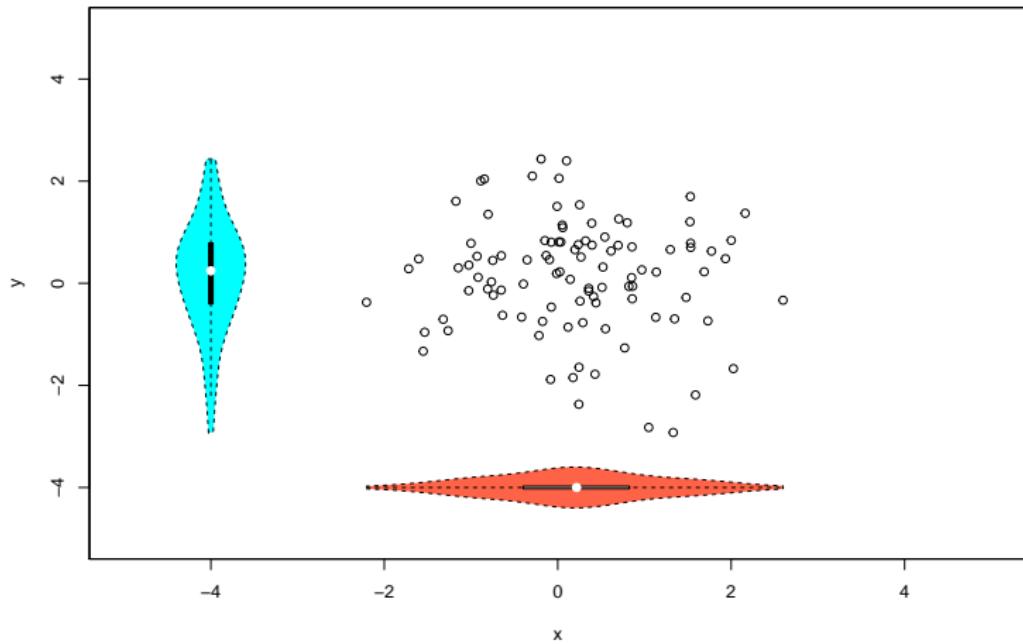
- Baut auf Boxplot auf
- Zusätzlich Informationen über Dichte der Daten
- Dichte wird über Kernel Methode berechnet.
- weißer Punkt - Median
- Je weiter die Ausdehnung, desto größer ist die Dichte an dieser Stelle.

```
# Beispieldaten erzeugen
x <- rnorm(100)
y <- rnorm(100)
```

# Die Bibliothek vioplot

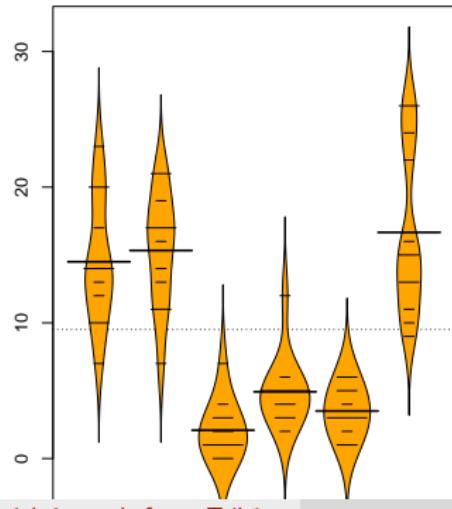
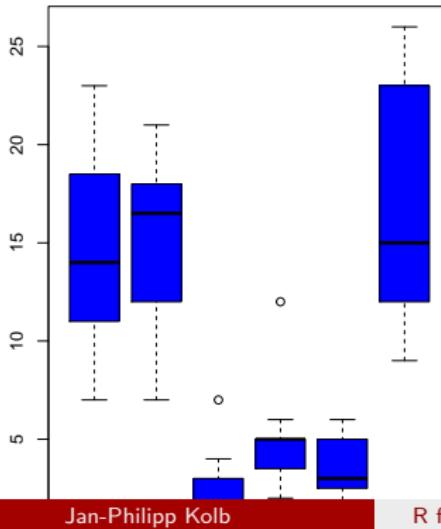
```
library(vioplot)
plot(x, y, xlim=c(-5,5), ylim=c(-5,5))
vioplot(x, col="tomato", horizontal=TRUE, at=-4,
        add=TRUE, lty=2, rectCol="gray")
vioplot(y, col="cyan", horizontal=FALSE, at=-4,
        add=TRUE, lty=2)
```

# vioplot - Das Ergebnis



## Alternativen zum Boxplot

```
library(beanplot)
par(mfrow = c(1,2))
boxplot(count~spray,data=InsectSprays,col="blue")
beanplot(count~spray,data=InsectSprays,col="orange")
```

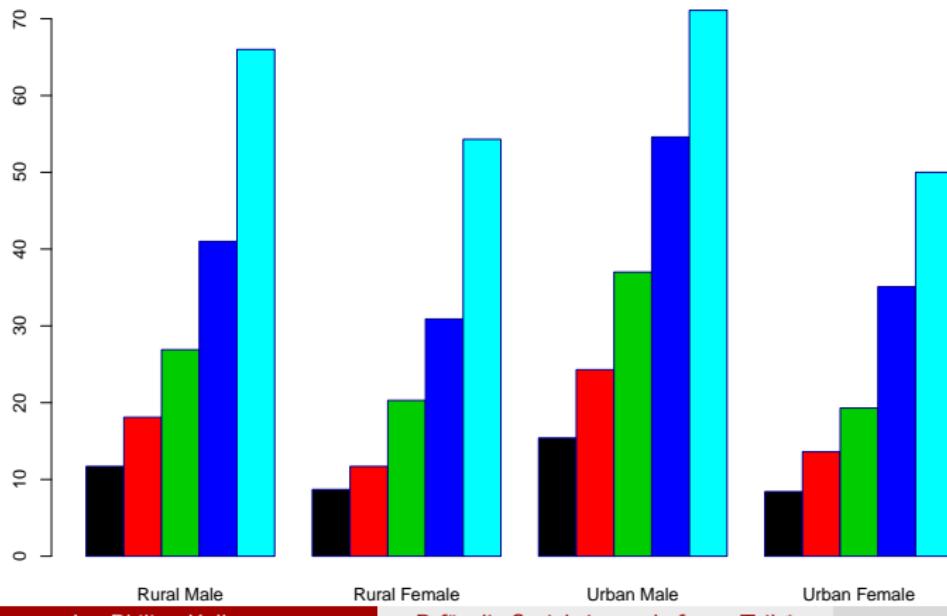


# CMYK Farbschema

```
pdf("test.cmyk.pdf", colormodel='cmyk')
pie(1:10, col=1:10)
dev.off()
```

# Aufgabe - einfache Grafiken

- Laden Sie den Datensatz VADeaths und erzeugen Sie den folgenden plot:



# Datenanalyse

# Den Datensatz laden

```
## Warning: NAs durch Umwandlung erzeugt

library(foreign)
dat <- read.dta(
  "https://github.com/Japhilko/RSocialScience/blob/master/data/
  GPanel.dta?raw=true")
dat$bazq020a <- as.numeric(dat$bazq020a)
```

# Streuungsmaße

- Varianz: `var()`
- Standardabweichung: `sd()`
- Minimum und Maximum: `min()` und `max()`
- Range: `range()`

```
var(dat$bazq020a)  
## [1] NA  
  
var(dat$bazq020a,na.rm=T)  
## [1] 476.8859  
  
sd(dat$bazq020a,na.rm=T)  
## [1] 21.83772  
  
range(dat$bazq020a,na.rm=T)
```

# Häufigkeiten und gruppierte Kennwerte

- Eine Auszählung der Häufigkeiten der Merkmale einer Variable liefert `table()`
- Mit `table()` sind auch Kreuztabellierungen möglich indem zwei Variablen durch Komma getrennt werden: `table(x,y)` liefert Häufigkeiten von y für gegebene Ausprägungen von x

```
table(dat$a11d054a)
```

```
##  
## Männlich Weiblich  
##          43        57
```

## Tabellieren - weiteres Beispiel

```
?table
```

```
table(dat$a11d054a)
```

```
##
```

```
## Männlich Weiblich
```

```
##      43       57
```

```
table(dat$a11d054a,dat$a11d056z)
```

```
##
```

```
##          Ambiguous answer Item nonresponse Not reached Unknown
```

```
##  Männlich                      0                      0                      0
```

```
##  Weiblich                      0                      0                      0
```

```
##
```

```
##          Not in panel 18 bis unter 20 Jahre 20 bis unter 25 Jahre
```

```
##  Männlich                      0                      2
```

```
##  Weiblich
```

# Häufigkeitstabellen

- `prop.table()` liefert die relativen Häufigkeiten
- Wird die Funktion außerhalb einer `table()` Funktion geschrieben erhält man die relativen Häufigkeiten bezogen auf alle Zellen

## Die Funktion prop.table

```
?prop.table
```

```
prop.table(table(dat$a11d054a,dat$a11d056z),1)
```

```
##  
##          Ambiguous answer Item nonresponse Not reached Un  
##  Männlich      0.000000000 0.000000000 0.000000000  
##  Weiblich      0.000000000 0.000000000 0.000000000  
##  
##          Not in panel 18 bis unter 20 Jahre 20 bis unter  
##  Männlich      0.000000000 0.046511630 0.0  
##  Weiblich      0.000000000 0.035087720 0.  
##  
##          25 bis unter 30 Jahre 30 bis unter 35 Jahre  
##  Männlich      0.046511630 0.069767440  
##  Weiblich      0.087719300 0.035087720  
##
```

# Die aggregate Funktion

- Mit der aggregate() Funktion können Kennwerte für Untergruppen erstellt werden
- aggregate(x,by,FUN) müssen mindestens drei Argumente übergeben werden:

```
aggregate(dat$bazq020a, by=list(dat$a11d054a), mean, na.rm=T)
```

```
##      Group.1          x
## 1 Männlich 13.534884
## 2 Weiblich  8.773585
```

x: ein oder mehrere Beobachtungsvektor(en) für den der Kennwert berechnet werden soll

by: eine oder mehrere bedingende Variable(n)

FUN: die Funktion welche den Kennwert berechnet (z.B. mean oder sd)

## Beispieldatensatz - apply Funktion

```
ApplyDat <- cbind(1:4,runif(4),rnorm(4))
```

1	0.4537133	0.0455947
2	0.7668416	0.1840527
3	0.1274953	1.4282038
4	0.5330915	0.8005702

## Argumente der Funktion apply

?apply

- Für margin=1 die Funktion mean auf die Reihen angewendet,
- Für margin=2 die Funktion mean auf die Spalten angewendet,
- Anstatt mean können auch andere Funktionen wie var, sd oder length verwendet werden.

# Die apply Funktion anwenden

```
apply(ApplyDat, 1, mean)
```

```
## [1] 0.4997693 0.9836314 1.5185664 1.7778872
```

```
apply(ApplyDat, 2, mean)
```

```
## [1] 2.5000000 0.4702854 0.6146054
```

## Die Funktion apply

```
apply(ApplyDat, 1, var)

## [1] 0.2293132 0.8596645 2.0689448 3.7212251

apply(ApplyDat, 1, sd)

## [1] 0.4788666 0.9271810 1.4383827 1.9290477

apply(ApplyDat, 1, range)

## [,1]      [,2]      [,3]      [,4]
## [1,] 0.04559471 0.1840527 0.1274953 0.5330915
## [2,] 1.00000000 2.0000000 3.0000000 4.0000000

apply(ApplyDat, 1, length)

## [1] 3 3 3 3
```

# Die Funktion tapply

?tapply

- Auch andere Funktionen können eingesetzt werden. . . - Auch selbst programmierte Funktionen
- Im Beispiel wird die einfachste eigene Funktion angewendet.

## Beispiel Funktion tapply

```
tapply(dat$a11d054a,dat$a11d056z,mean)

## Warning in mean.default(X[[i]], ...): argument is not numeric
## returning NA

## Warning in mean.default(X[[i]], ...): argument is not numeric
## returning NA

## Warning in mean.default(X[[i]], ...): argument is not numeric
## returning NA

## Warning in mean.default(X[[i]], ...): argument is not numeric
## returning NA

## Warning in mean.default(X[[i]], ...): argument is not numeric
## returning NA
```

# Links Datenanalyse

- Die Benutzung von `apply`, `tapply`, etc. (Artikel bei R-bloggers)
- Quick-R zu deskriptiver Statistik
- Quick-R zur Funktion `aggregate`

# Grafiken und Zusammenhang

# Ein Plot sagt mehr als 1000 Worte

- Grafisch gestützte Datenanalyse ist toll
- Gute Plots können zu einem besseren Verständnis beitragen
- Einen Plot zu generieren geht schnell
- Einen guten Plot zu machen kann sehr lange dauern
- Mit R Plots zu generieren macht Spaß
- Mit R erstellte Plots haben hohe Qualität
- Fast jeder Plottyp wird von R unterstützt
- R kennt eine große Menge an Exportformaten für Grafiken

# Plot ist nicht gleich Plot

- Bereits das base Package bringt eine große Menge von Plot Funktionen mit
- Das lattice Packet erweitert dessen Funktionalität
- Eine weit über diese Einführung hinausgehende Übersicht findet sich in Murrell, P (2006): R Graphics.

# Task View zu Thema Graphiken

CRAN Task View: Graphic Displays & Dynamic Graphics & Graphic Devices & Visualization

**Maintainer:** Nicholas Lewin-Koh

**Contact:** nikko at hailmail.net

**Version:** 2015-01-07

**URL:** <https://CRAN.R-project.org/view=Graphics>

R is rich with facilities for creating and developing interesting graphics. Base R contains functionality for many plot types including coplots, mosaic plots, biplots, and the list goes on. There are devices such as postscript, png, jpeg and pdf for outputting graphics as well as device drivers for all platforms running R. [lattice](#) and grid are supplied with R's recommended packages and are included in every binary distribution. [lattice](#) is an R implementation of William Cleveland's trellis graphics, while grid defines a much more flexible graphics environment than the base R graphics.

# Datensatz

```
library(mlmRev)  
data(Chem97)
```

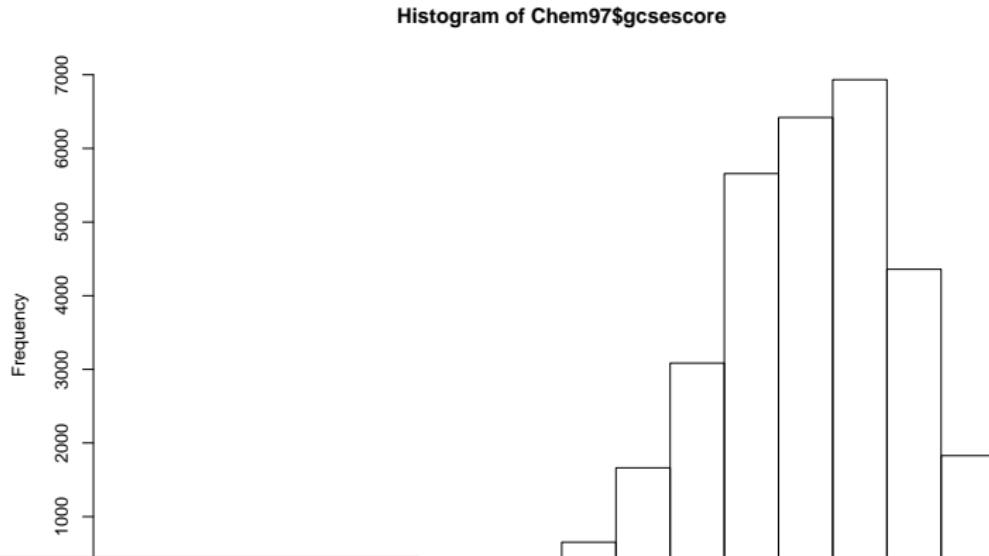
- [lea] Local Education Authority - a factor
- [school] School identifier - a factor
- [student] Student identifier - a factor
- [score] Point score on A-level Chemistry in 1997
- [gender] Student's gender
- [age] Age in month, centred at 222 months or 18.5 years
- [gcsescore] Average GCSE score of individual.
- [gcsecnt] Average GCSE score of individual, centered at mean.

# Histogramm - Die Funktion hist()

Wir erstellen ein Histogramm der Variable gcsescore:

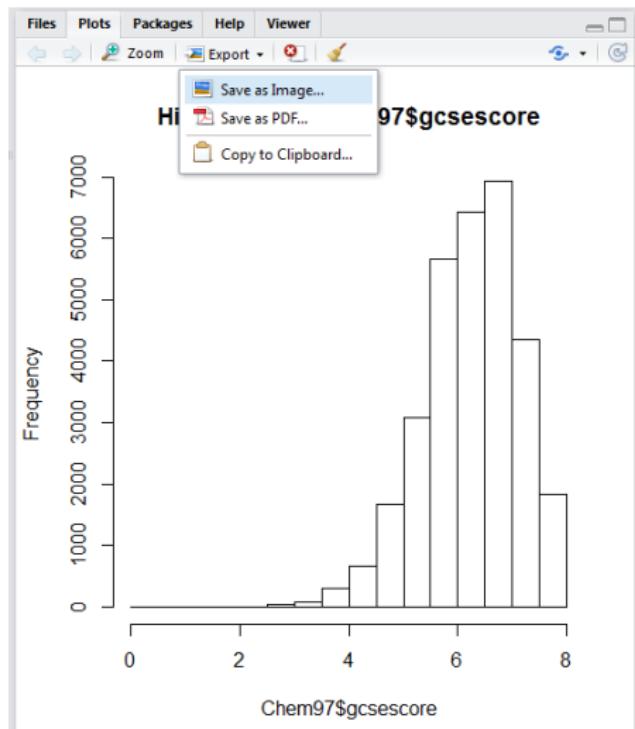
?hist

`hist(Chem97$gcsescore)`



# Graphik speichern

- Mit dem button Export in Rstudio kann man die Grafik speichern.



# Befehl um Graphik zu speichern

- Alternativ auch bspw. mit den Befehlen png, pdf oder jpeg

```
png("Histogramm.png")
hist(Chem97$gcsescore)
dev.off()
```

# Histogramme

- Die Funktion `hist()` plottet ein Histogramm der Daten
- Der Funktion muss mindestens ein Beobachtungsvektor übergeben werden
- `hist()` hat noch sehr viel mehr Argumente, die alle (sinnvolle) default values haben

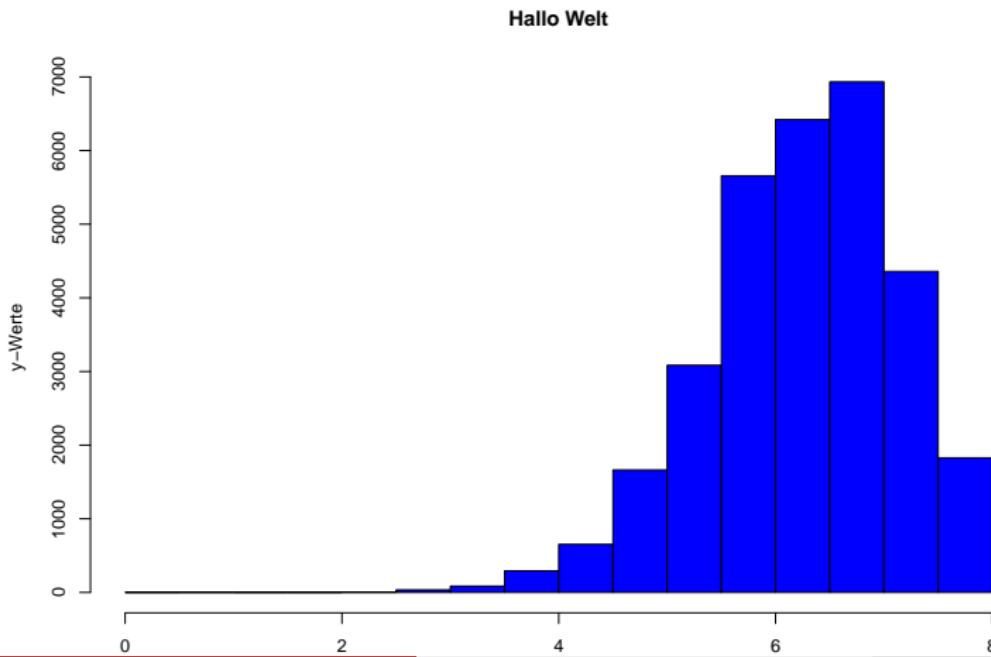
---

Argument	Bedeutung	Beispiel
main	Überschrift	main="Hallo Welt"
xlab	x-Achsenbeschriftung	xlab="x-Werte"
ylab	y-Achsenbeschriftung	ylab="y-Werte"
col	Farbe	col="blue"

---

# Histogramm

```
hist(Chem97$gcsescore, col="blue",  
main="Hallo Welt", ylab="y-Werte", xlab="x-Werte")
```



# Barplot

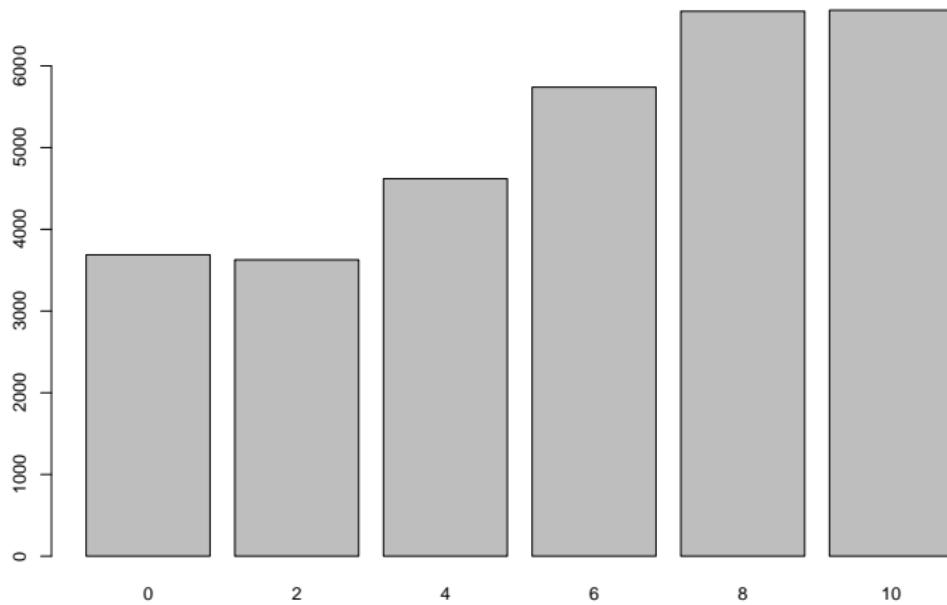
- Die Funktion `barplot()` erzeugt aus einer Häufigkeitstabelle einen Barplot
- Ist das übergebene Tabellen-Objekt zweidimensional wird ein bedingter Barplot erstellt

```
tabScore <- table(Chem97$score)
```

```
barplot(tabScore)
```

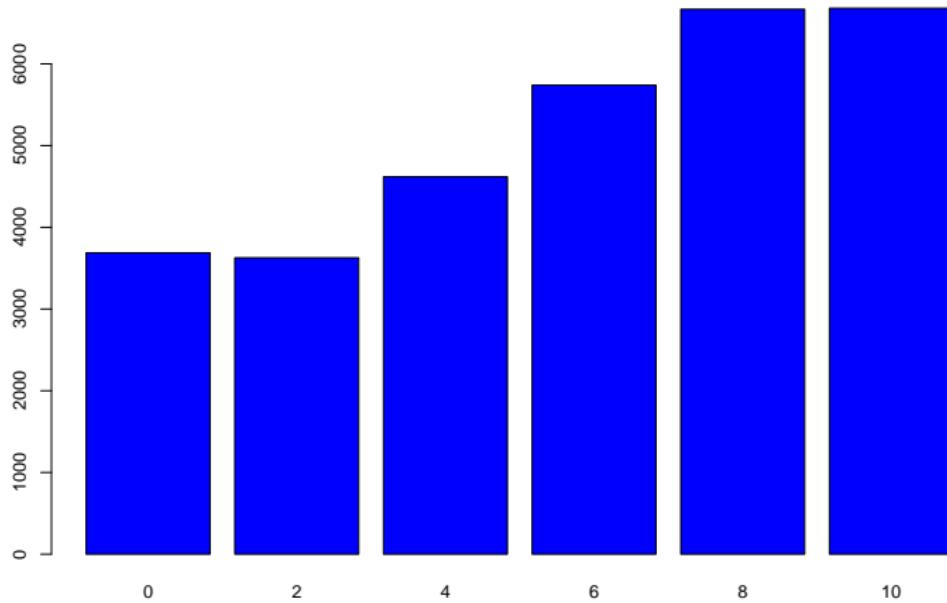
# Barplots und barcharts

```
barplot(tabScore)
```



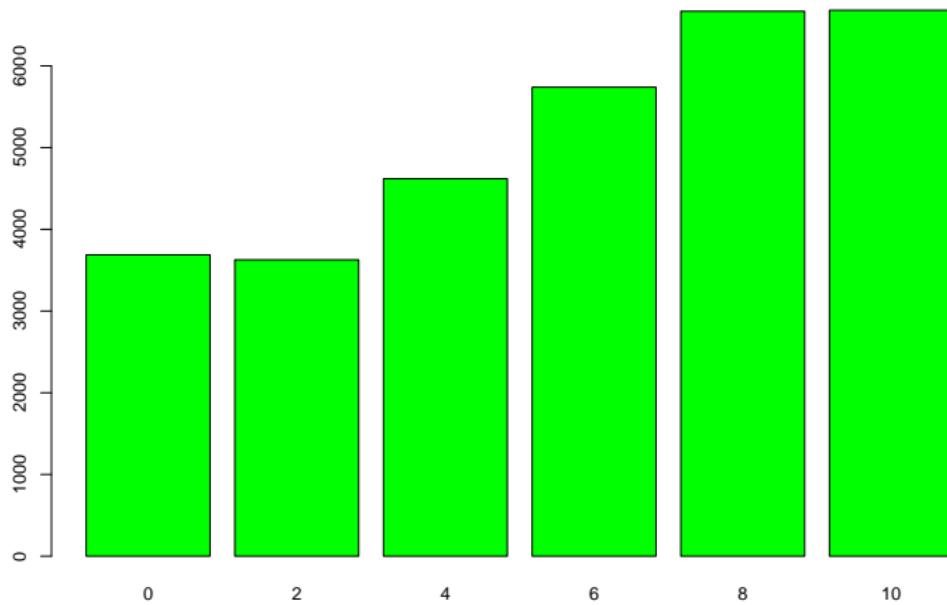
## Mehr Farben:

```
barplot(tabScore, col=rgb(0,0,1))
```



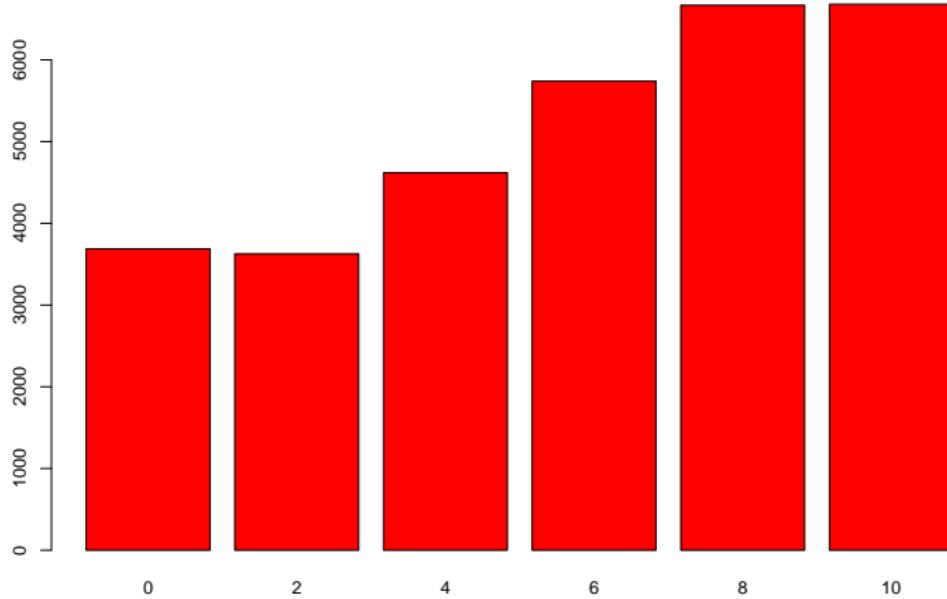
# Grüne Farbe

```
barplot(tabScore, col=rgb(0,1,0))
```



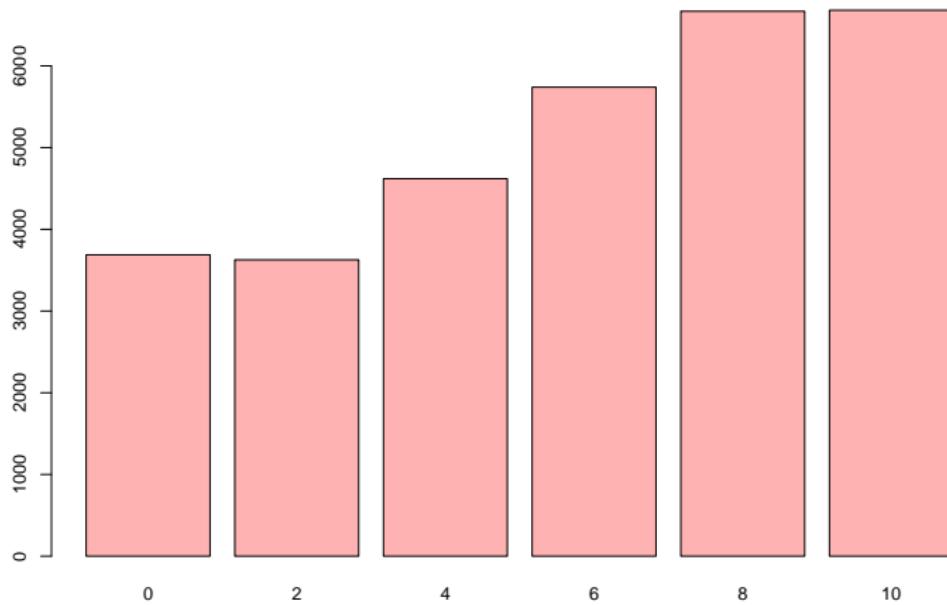
# Rote Farbe

```
barplot(tabScore, col=rgb(1,0,0))
```



# Transparent

```
barplot(tabScore, col=rgb(1,0,0,.3))
```



# Scatterplots

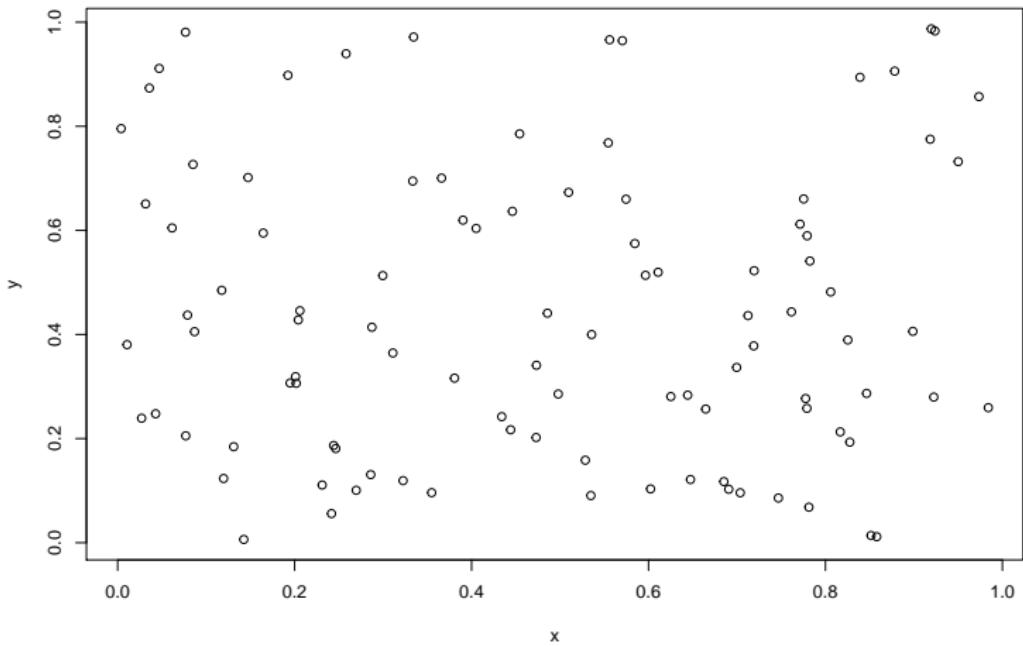
- Ein einfacher two-way Scatterplot kann mit der Funktion `plot()` erstellt werden
- `plot()` muss mindestens ein `x` und ein `y` Beobachtungsvektor übergeben werden
- Um die Farbe der Plot-Symbole anzupassen gibt es die Option `col` (Farbe als character oder numerisch)
- Die Plot-Symbole selbst können mit `pch` (plotting character) angepasst werden (character oder numerisch)
- Die Achsenbeschriftungen (`labels`) werden mit `xlab` und `ylab` definiert

# Beispieldaten für Scatterplot

```
x <- runif(100)  
y <- runif(100)
```

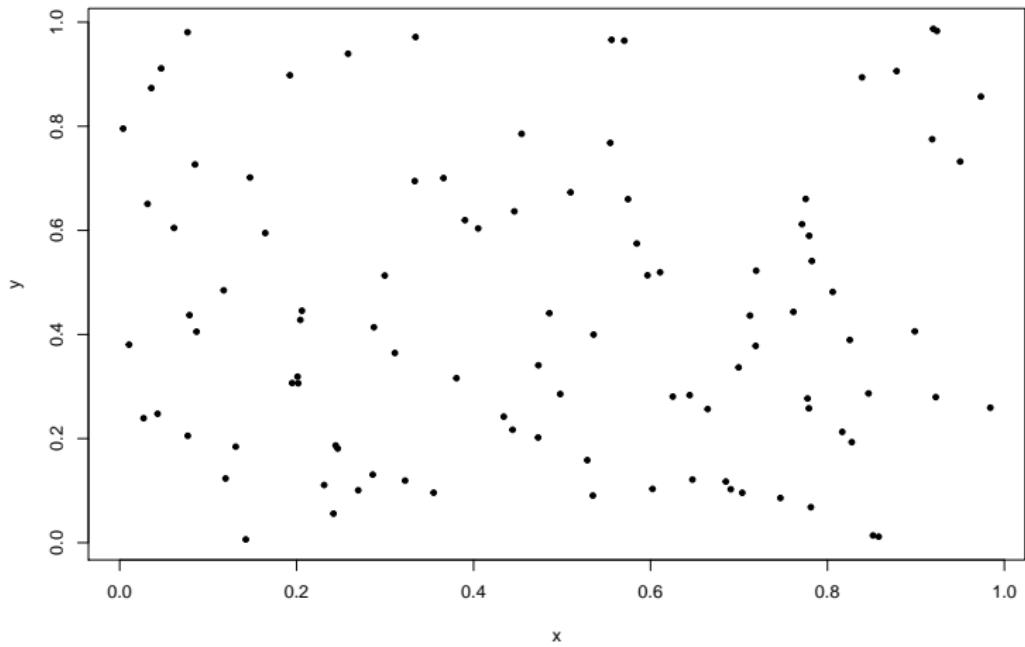
# Einfacher Scatterplot

`plot(x, y)`



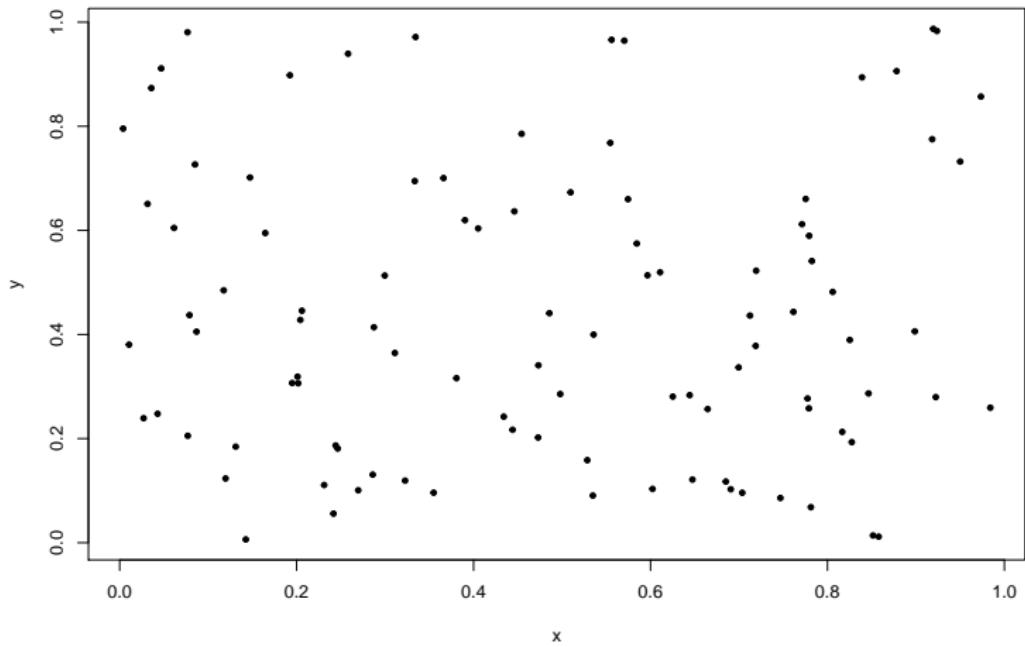
# Einfacher Scatterplot II

```
plot(x,y,pch=20)
```



# Einfacher Scatterplot III

```
plot(x,y,pch=20)
```



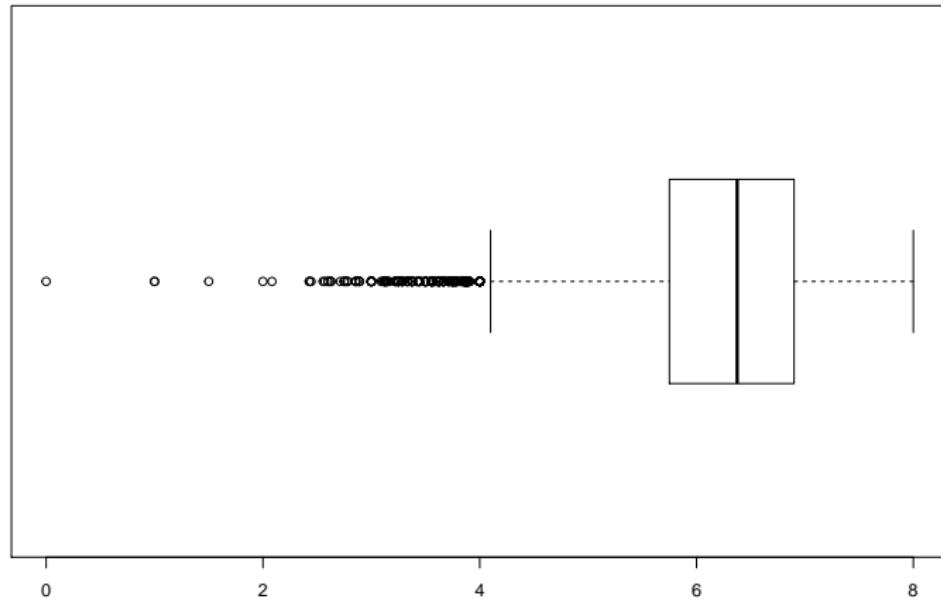
# Boxplot

- Einen einfachen Boxplot erstellt man mit `boxplot()`
- Auch `boxplot()` muss mindestens ein Beobachtungsvektor übergeben werden

?`boxplot`

# Horizontaler Boxplot

```
boxplot(Chem97$gcsescore,  
horizontal=TRUE)
```

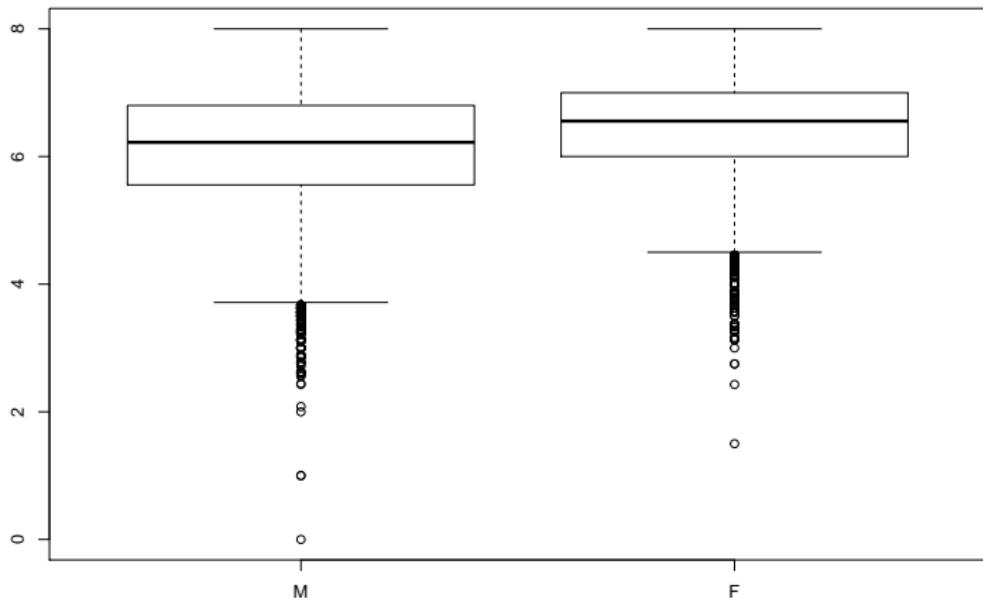


# Gruppierte Boxplots

- Ein sehr einfacher Weg, einen ersten Eindruck über bedingte Verteilungen zu bekommen ist über sog. Gruppierte notched Boxplots
- Dazu muss der Funktion `boxplot()` ein sog. Formel-Objekt übergeben werden
- Die bedingende Variable steht dabei auf der rechten Seite einer Tilde

# Beispiel grupierter Boxplot

```
boxplot(Chem97$gcsescore~Chem97$gender)
```



# Alternativen zu Boxplot

## Violinplot

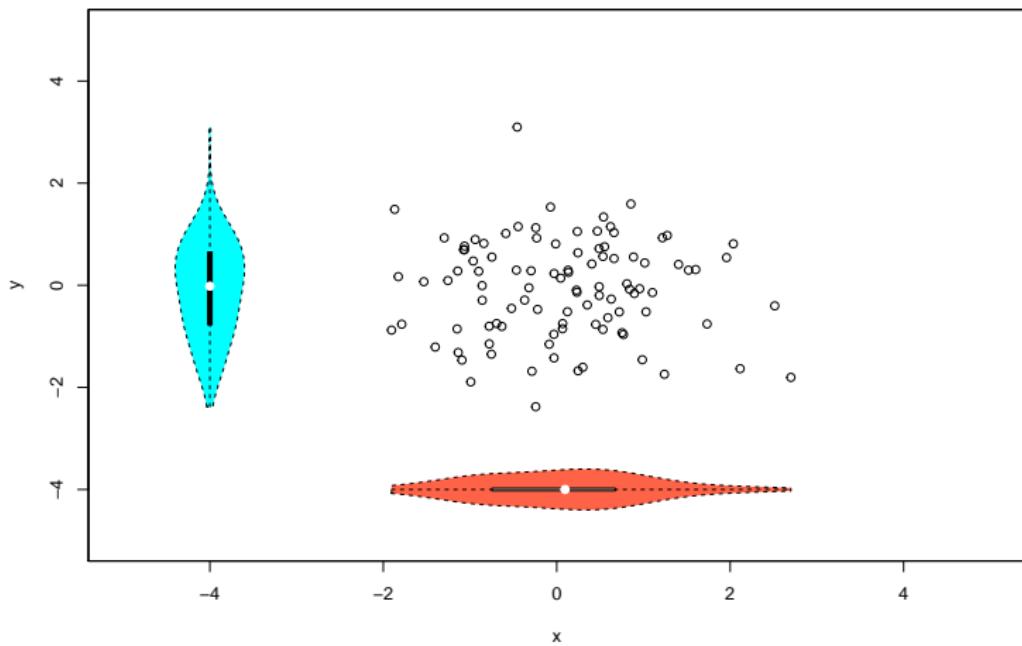
- Baut auf Boxplot auf
- Zusätzlich Informationen über Dichte der Daten
- Dichte wird über Kernel Methode berechnet.
- weißer Punkt - Median
- Je weiter die Ausdehnung, desto größer ist die Dichte an dieser Stelle.

```
# Beispieldaten erzeugen
x <- rnorm(100)
y <- rnorm(100)
```

# Die Bibliothek vioplot

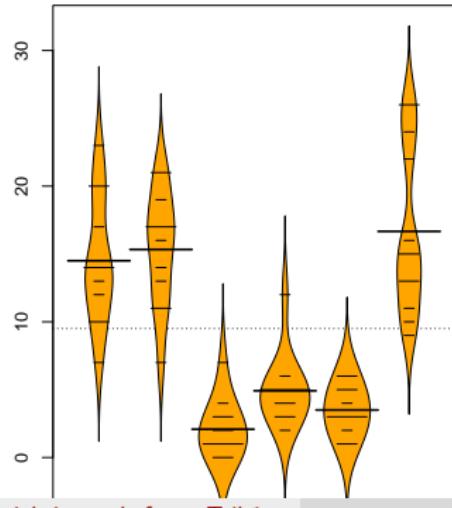
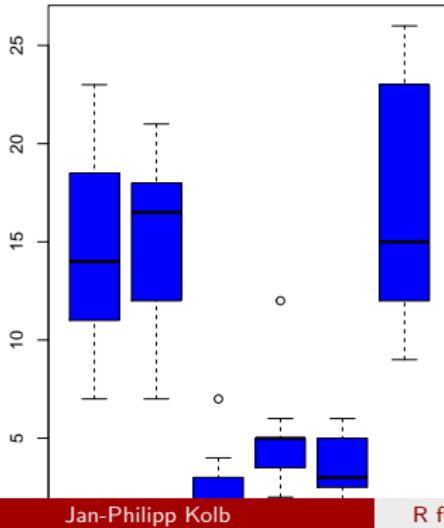
```
library(vioplot)
plot(x, y, xlim=c(-5,5), ylim=c(-5,5))
vioplot(x, col="tomato", horizontal=TRUE, at=-4,
        add=TRUE, lty=2, rectCol="gray")
vioplot(y, col="cyan", horizontal=FALSE, at=-4,
        add=TRUE, lty=2)
```

# vioplot - Das Ergebnis



## Alternativen zum Boxplot

```
library(beanplot)
par(mfrow = c(1,2))
boxplot(count~spray,data=InsectSprays,col="blue")
beanplot(count~spray,data=InsectSprays,col="orange")
```

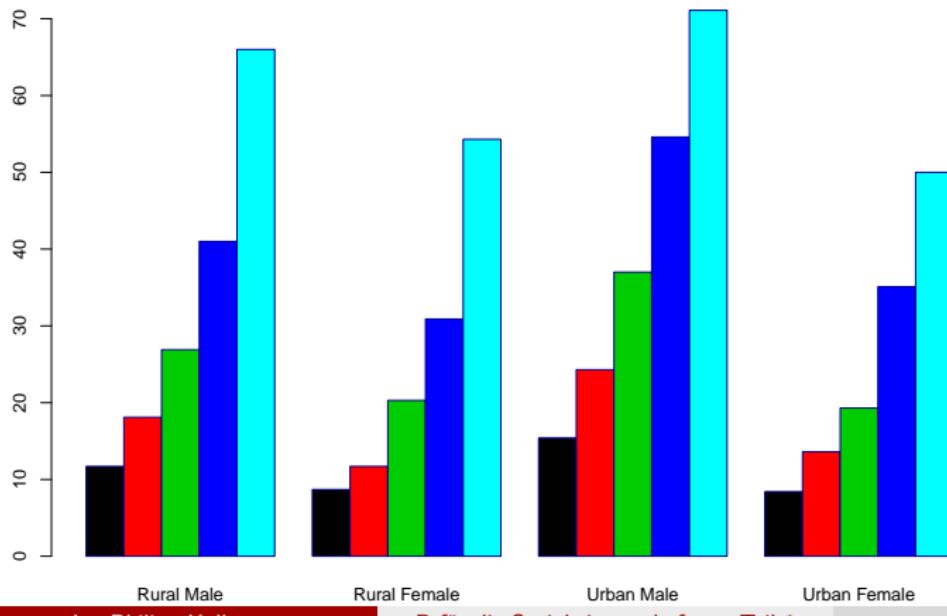


# CMYK Farbschema

```
pdf("test.cmyk.pdf", colormodel='cmyk')
pie(1:10, col=1:10)
dev.off()
```

# Aufgabe - einfache Grafiken

- Laden Sie den Datensatz VADeaths und erzeugen Sie den folgenden plot:



# Das lattice Paket

# Das lattice-Paket

*It is designed to meet most typical graphics needs with minimal tuning, but can also be easily extended to handle most nonstandard requirements.*

[http://stat.ethz.ch/R-manual/R-devel/library/lattice/html/  
Lattice.html](http://stat.ethz.ch/R-manual/R-devel/library/lattice/html/Lattice.html)

# Der Datensatz - Scores on A-level Chemistry in 1997

```
library("mlmRev")
data(Chem97)
```

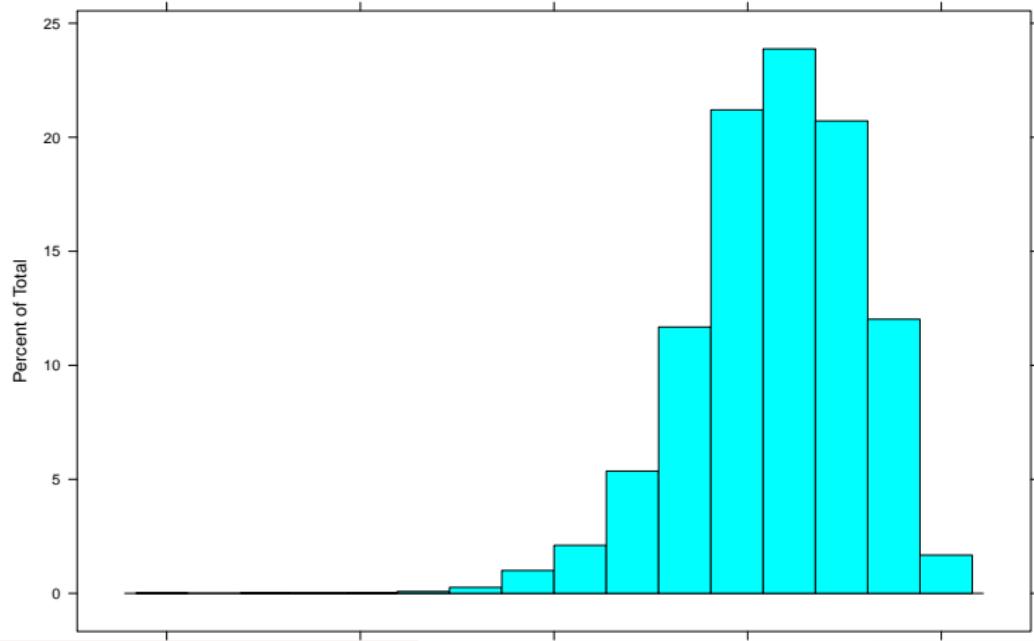
---

variables	categories
lea	Local Education Authority
school	School identifier
student	Student identifier
score	Point score on A-level Chemistry in 1997
gender	Student's gender
age	Age in month, centred at 222 months or 18.5 years
gcsescore	Average GCSE score of individual
gcsecnt	Average GCSE score of individual, centered at mean

---

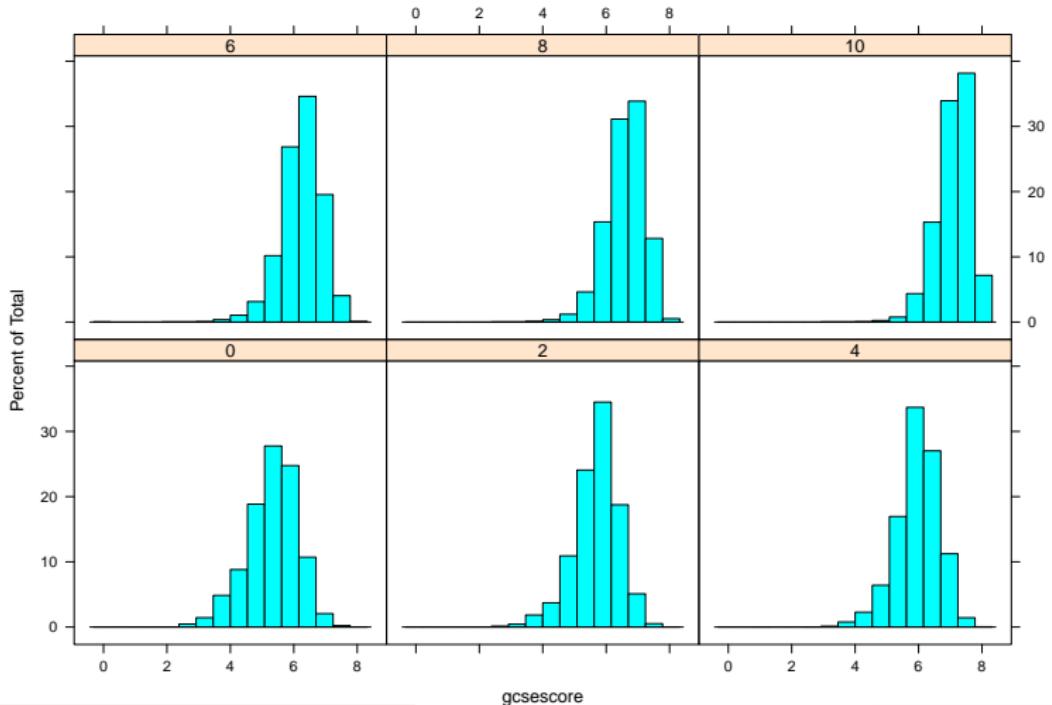
# Histogramm mit Lattice

```
library("lattice")
histogram(~ gcsescore, data = Chem97)
```



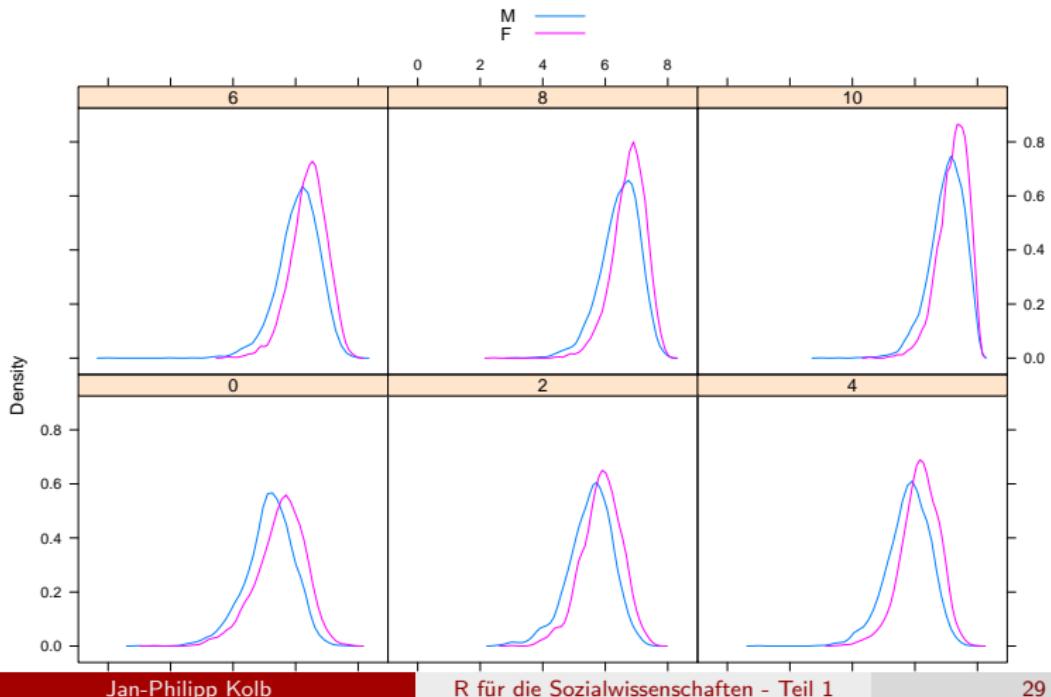
# Histogramm mit Lattice

```
histogram(~ gcsescore | factor(score), data = Chem97)
```



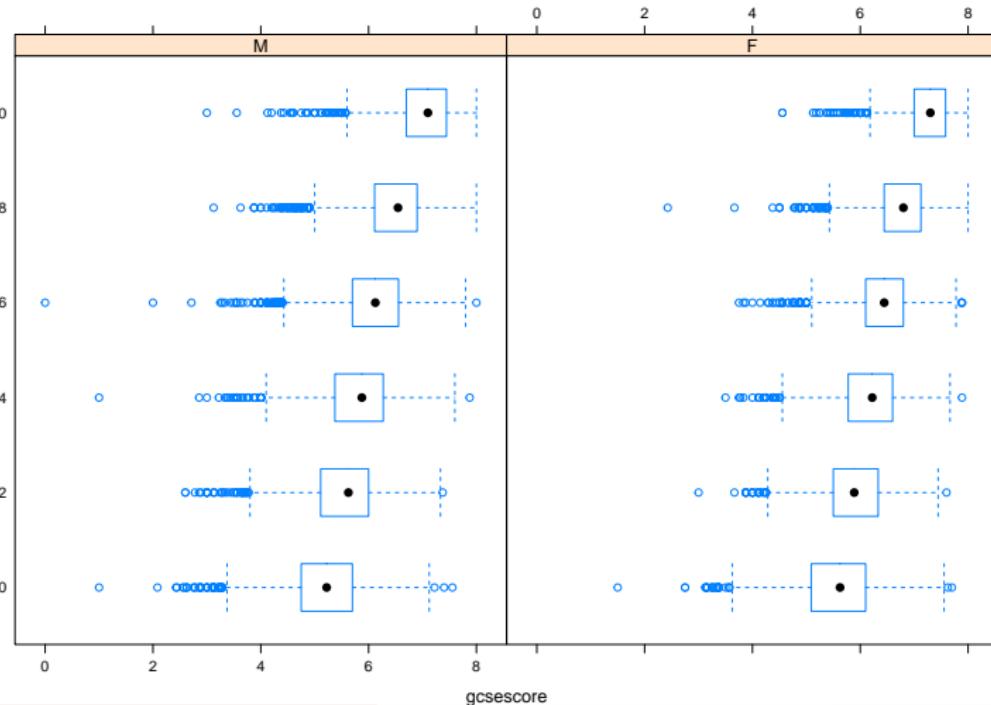
# Die Dichte mit Lattice zeichnen

```
densityplot(~ gcsescore | factor(score), Chem97,  
groups=gender, plot.points=FALSE, auto.key=TRUE)
```



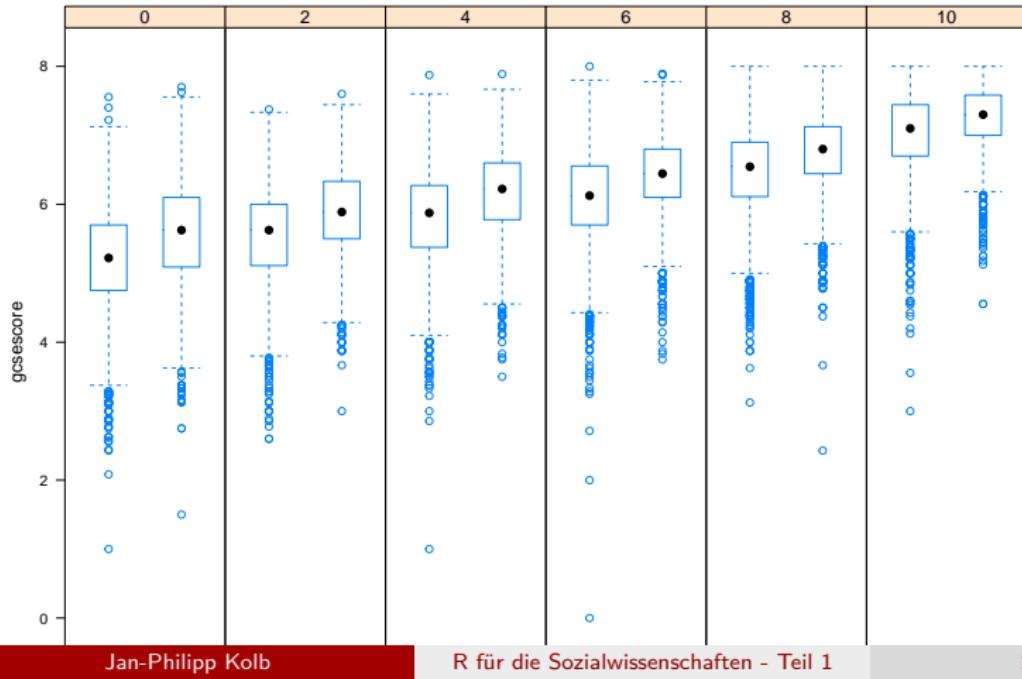
# Boxplot mit Lattice zeichnen

```
bwplot(factor(score) ~ gcsescore | gender, Chem97)
```



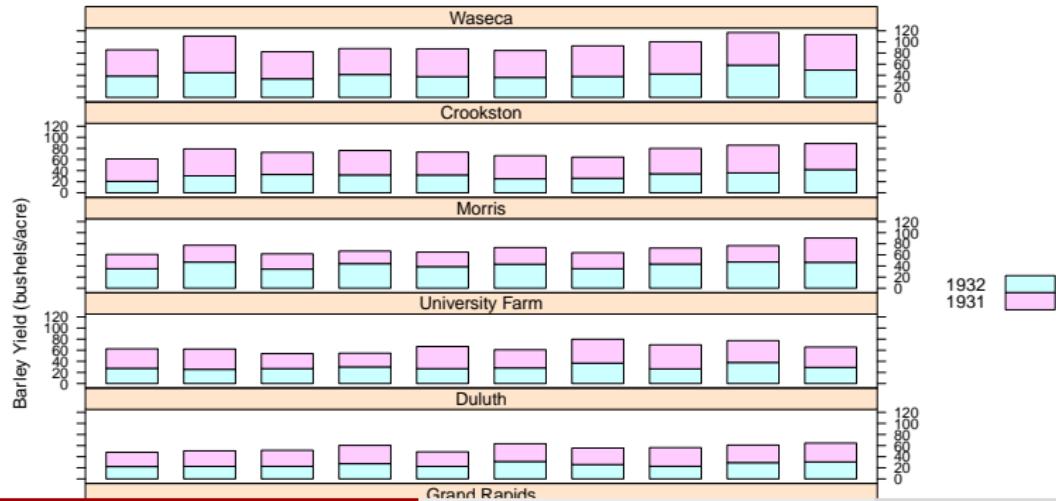
# Boxplot mit Lattice zeichnen

```
bwplot(gcsescore ~ gender | factor(score), Chem97,  
       layout = c(6, 1))
```



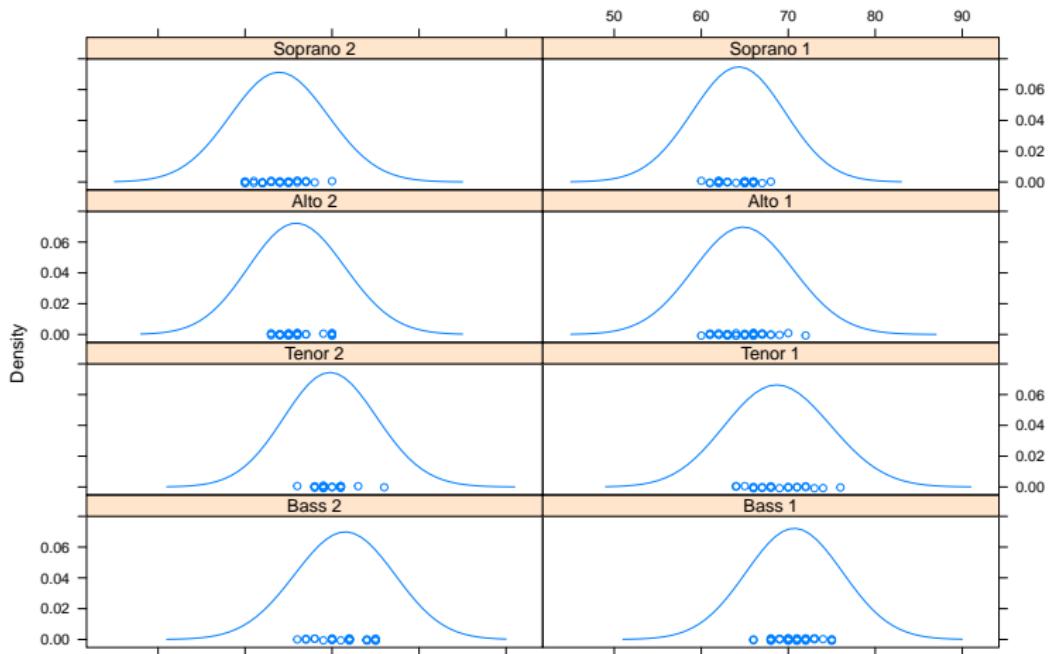
# Univariate Plots

```
barchart(yield ~ variety | site, data = barley,
          groups = year, layout = c(1,6), stack = TRUE,
          auto.key = list(space = "right"),
          ylab = "Barley Yield (bushels/acre)",
          scales = list(x = list(rot = 45)))
```



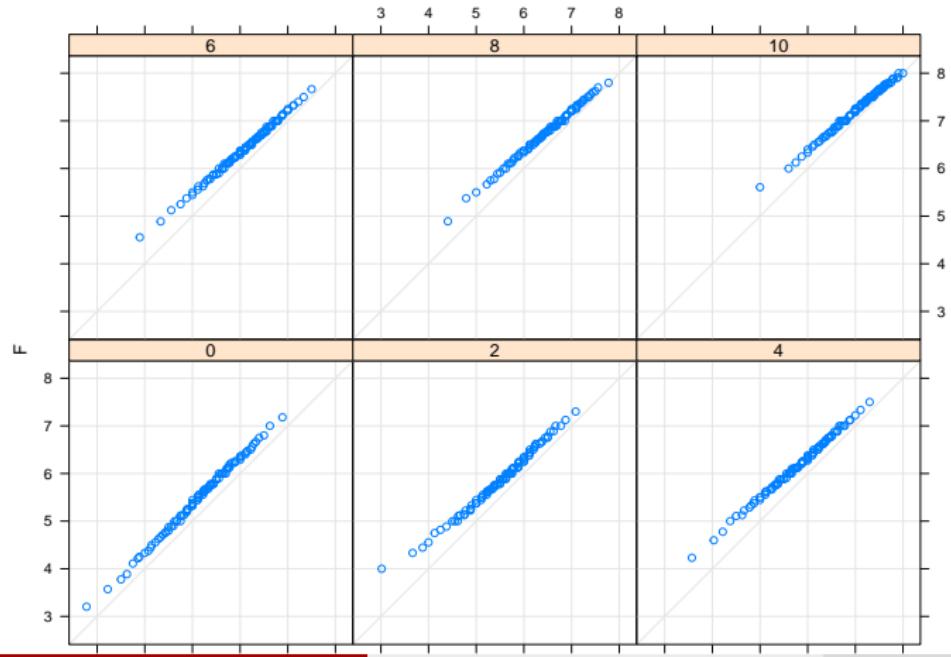
# Densityplot

```
densityplot(~height|voice.part,data=singer,layout = c(2,4),  
           xlab = "Height (inches)",bw = 5)
```



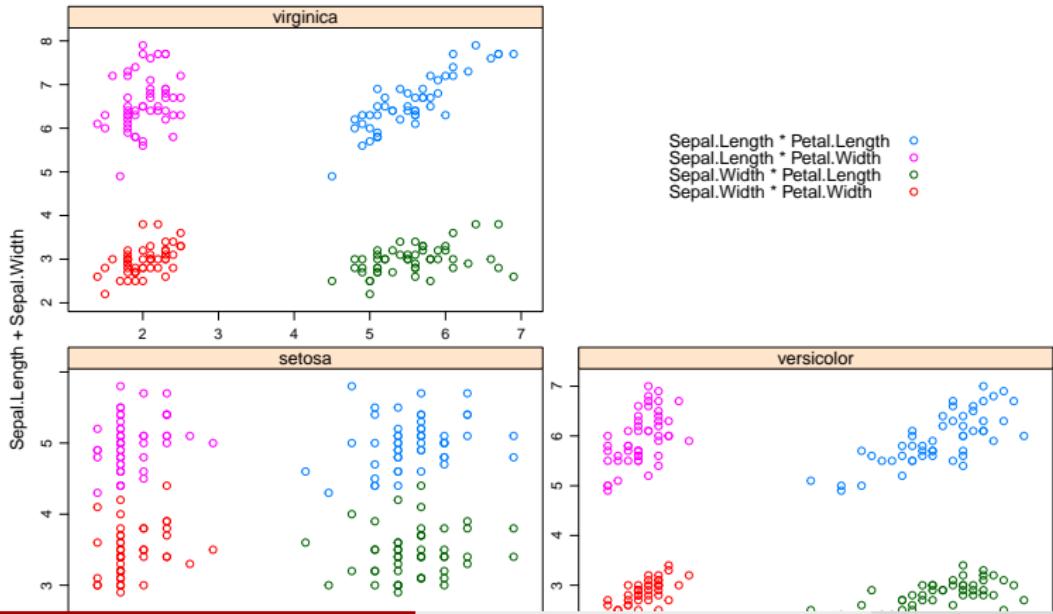
# Bivariate Plots

```
qq(gender ~ gcsescore | factor(score), Chem97,  
f.value = ppoints(100), type = c("p", "g"), aspect = 1)
```



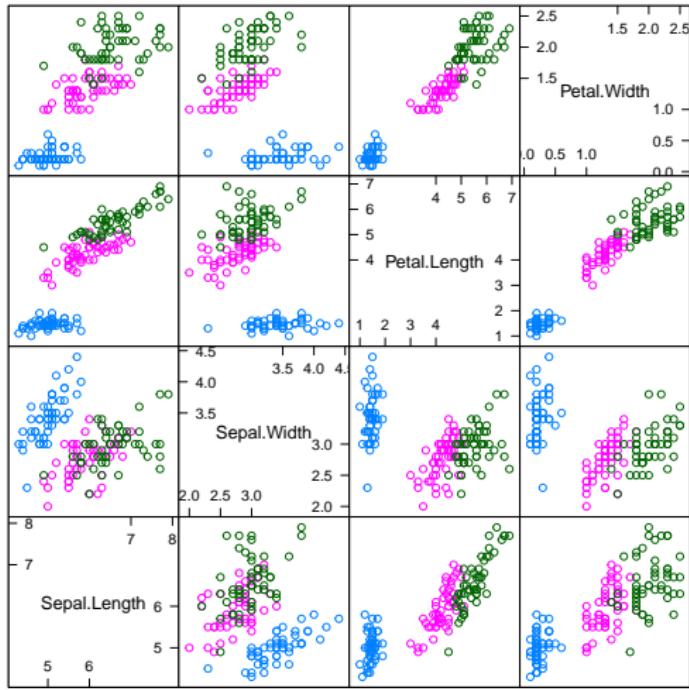
# xyplot

```
xyplot(Sepal.Length + Sepal.Width ~ Petal.Length + Petal.Width  
       data = iris, scales = "free", layout = c(2, 2),  
       auto.key = list(x = .6, y = .7, corner = c(0, 0)))
```



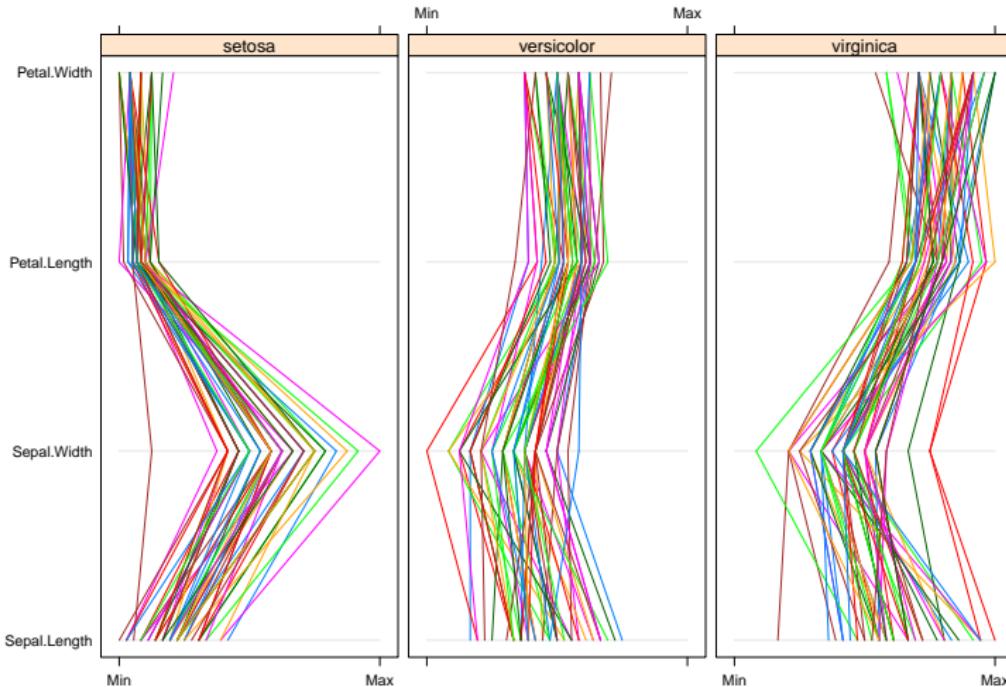
# Multivariate Plots

```
splom(~iris[1:4], groups = Species, data = iris)
```



# parallelplot

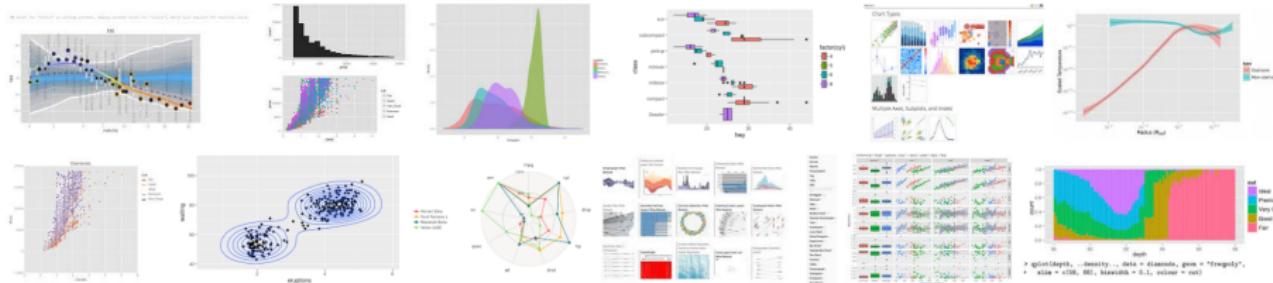
```
parallelplot(~iris[1:4] | Species, iris)
```



# ggplot und ggmap

# Das Paket ggplot2

- Entwickelt von Hadley Wickham
- Viele Informationen unter:  
<http://ggplot2.org/>
- Den Graphiken liegt eine eigene Grammatik zu Grunde



# Das Paket `ggplot2` installieren und laden

- Basiseinführung `ggplot2`

```
install.packages("ggplot2")
```

```
library(ggplot2)
```

# Der diamonds Datensatz

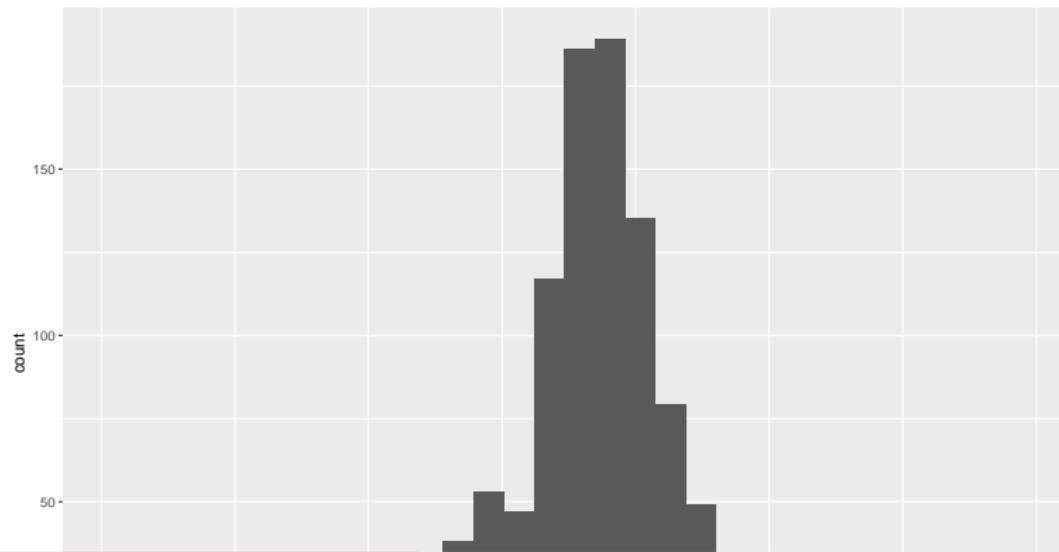
```
head(diamonds)
```

carat	cut	color	clarity	depth	table	price	x	y	z
0.23	Ideal	E	SI2	61.5	55	326	3.95	3.98	2.43
0.21	Premium	E	SI1	59.8	61	326	3.89	3.84	2.31
0.23	Good	E	VS1	56.9	65	327	4.05	4.07	2.31
0.29	Premium	I	VS2	62.4	58	334	4.20	4.23	2.63
0.31	Good	J	SI2	63.3	58	335	4.34	4.35	2.75
0.24	Very Good	J	VVS2	62.8	57	336	3.94	3.96	2.48

# Wie nutzt man qplot

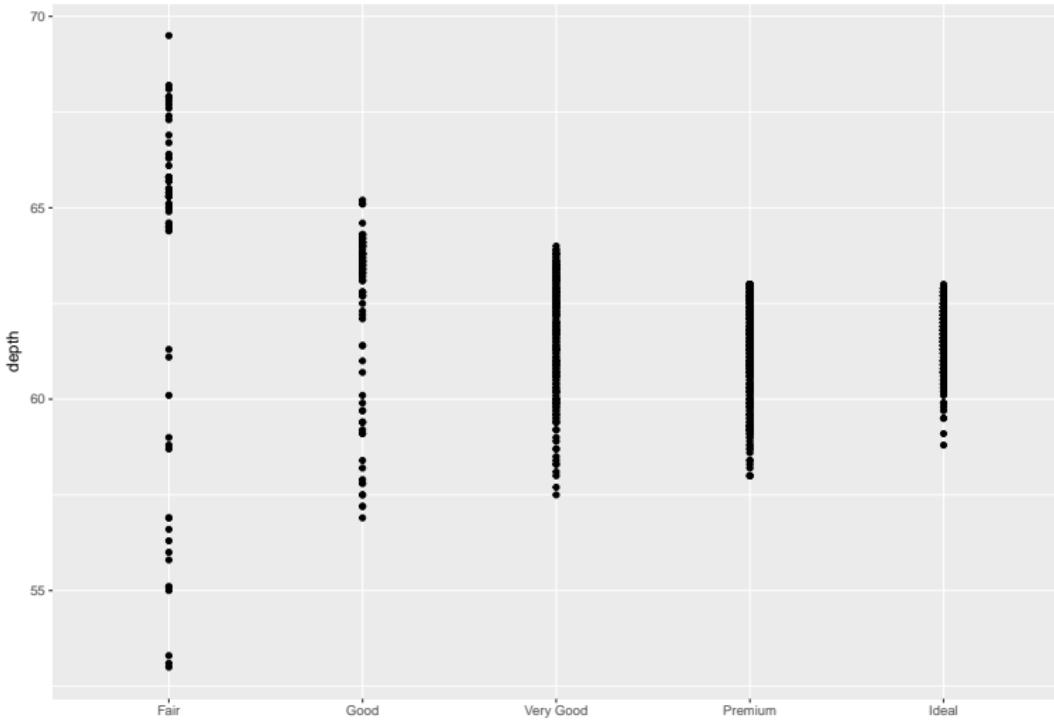
- qplot wird für schnelle Graphiken verwendet (quick plots)
- bei ggplot kann man alles bis ins Detail kontrollieren

```
# histogram  
qplot(depth, data=diamonds2)
```



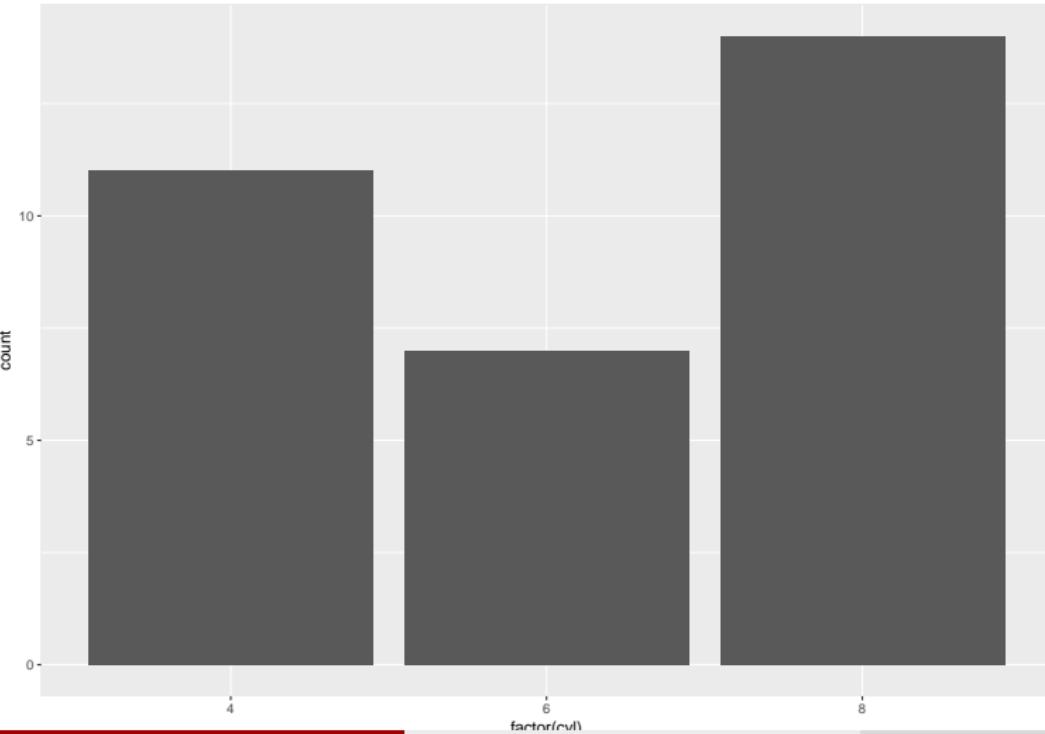
# Ein Balkendiagramm

```
qplot(cut, depth, data=diamonds2)
```



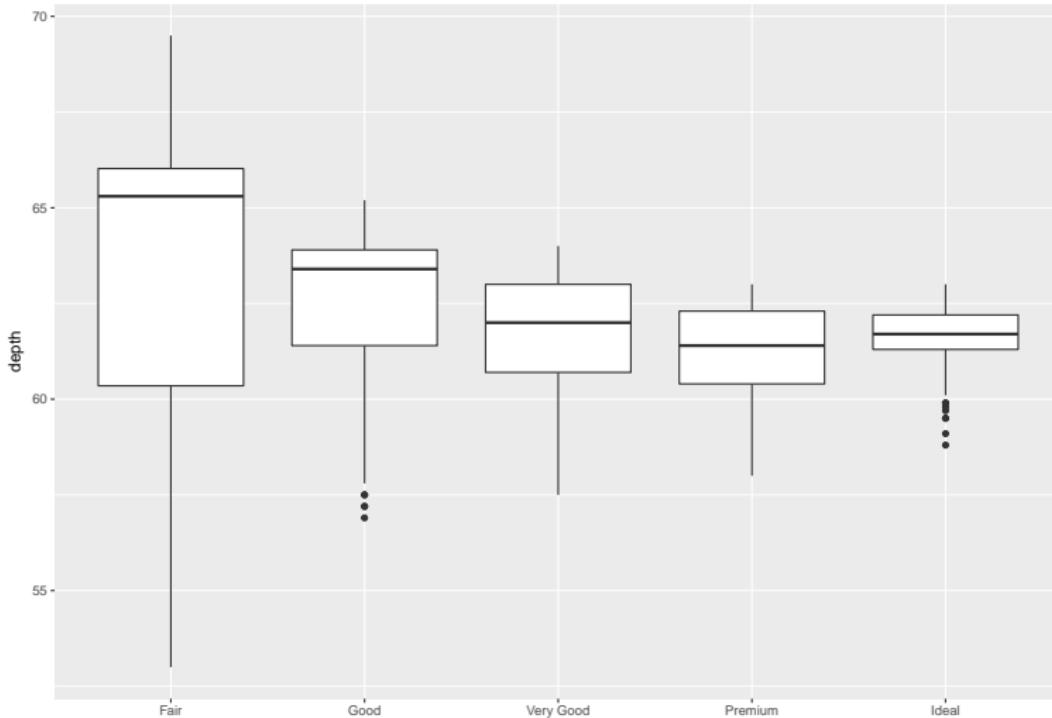
# Ein weiteres Balkendiagramm

```
qplot(factor(cyl), data=mtcars, geom="bar")
```



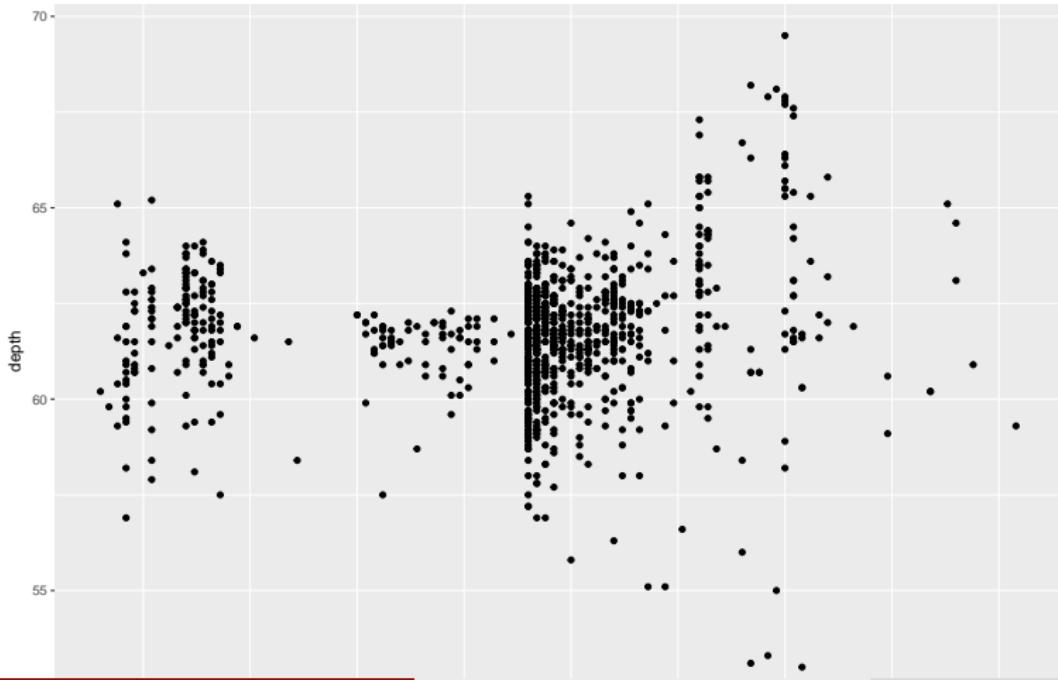
# Boxplot

```
qplot(data=diamonds2, x=cut, y=depth, geom="boxplot")
```



# Scatterplot

```
# scatterplot  
qplot(carat, depth, data=diamonds2)
```



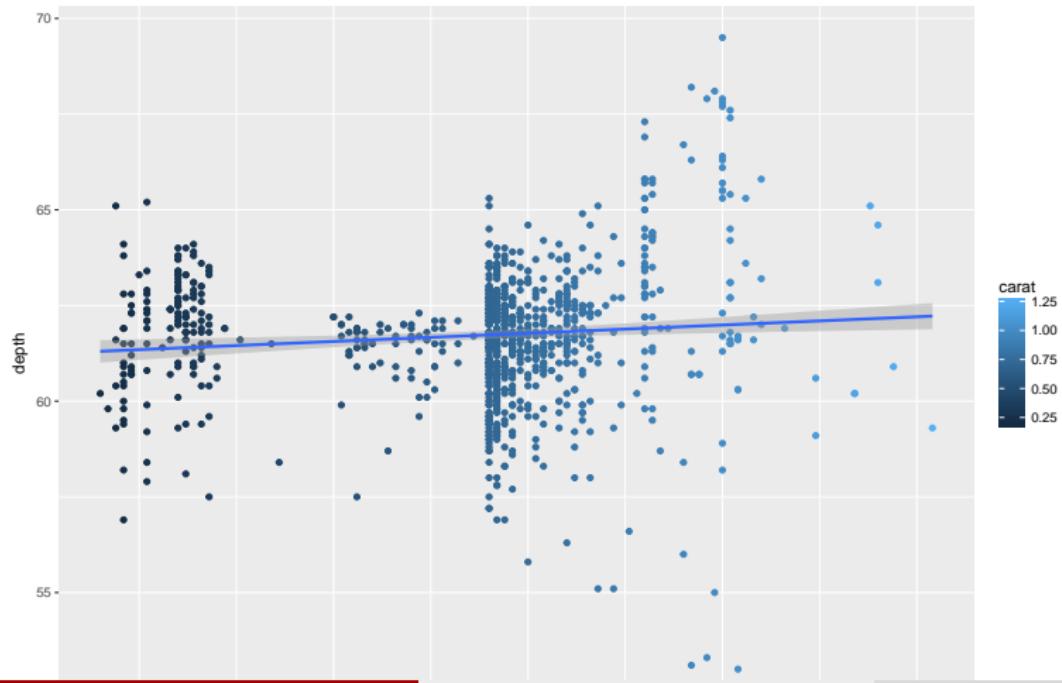
Farbe hinzufügen:

```
qplot(carat, depth, data=diamonds2, color=cut)
```



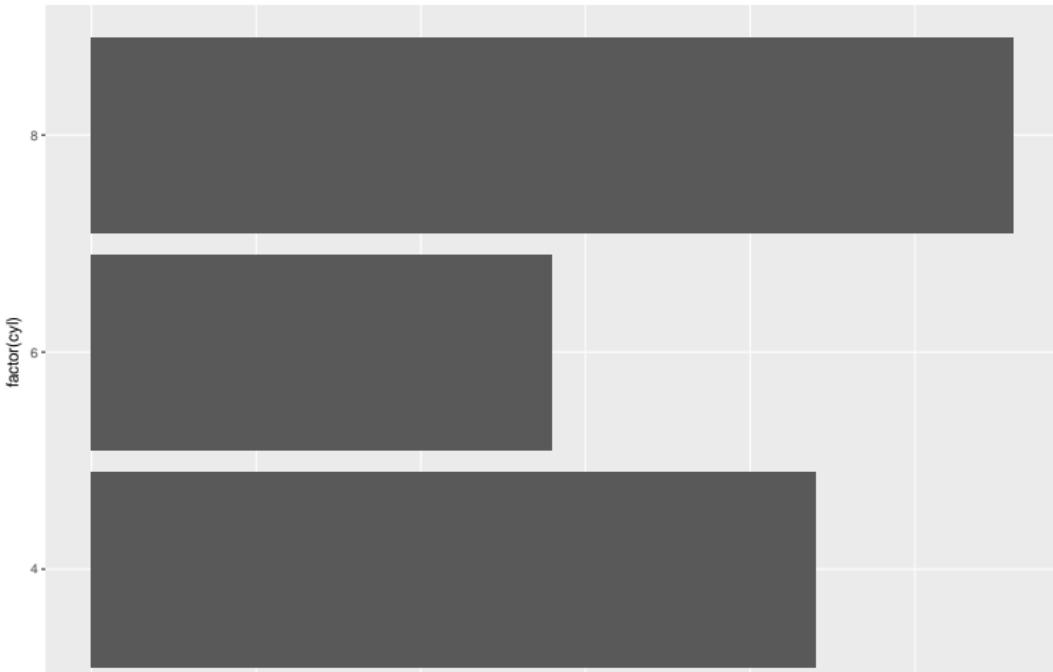
# Trendlinie hinzufügen

```
myGG<-qplot(data=diamonds2, x=carat, y=depth, color=carat)  
myGG + stat_smooth(method="lm")
```



# Graphik drehen

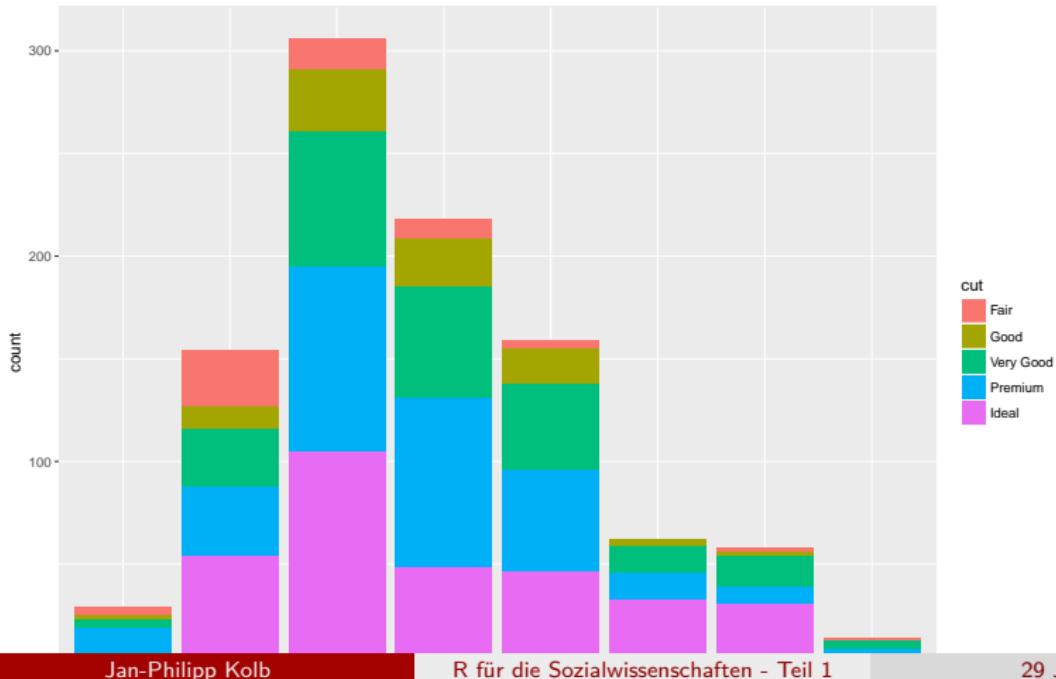
```
qplot(factor(cyl), data=mtcars, geom="bar") +  
coord_flip()
```



# Wie nutzt man ggplot

- die aesthetics:

```
ggplot(diamonds2, aes(clarity, fill=cut)) + geom_bar()
```



## Farben selber wählen

Es wird das Paket RColorBrewer verwendet um die Farbpalette zu ändern

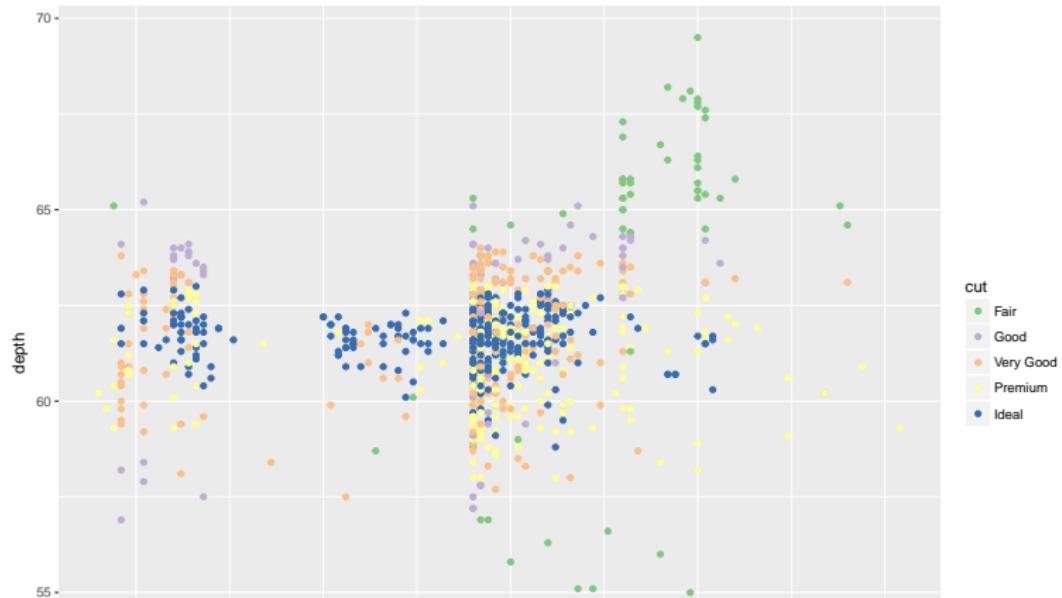
```
install.packages("RColorBrewer")

library(RColorBrewer)
myColors <- brewer.pal(5, "Accent")
names(myColors) <- levels(diamonds2$cut)
colScale <- scale_colour_manual(name = "cut",
                                  values = myColors)
```

<http://stackoverflow.com/questions/6919025/>

# Eine Graphik mit den gewählten Farben

```
p <- ggplot(diamonds2,aes(carat, depth, colour = cut)) +  
  geom_point()  
p + colScale
```



# Speichern mit ggsave

```
ggsave("Graphik.jpg")
```

## Links

- Warum man ggplot2 für einfache Grafiken nutzen sollte

# Why I use ggplot2

February 12, 2016

By David Robinson

 Like 585

 Share

 Share 69

(This article was first published on [Variance Explained](#), and kindly contributed to [R-bloggers](#))

590  
SHARES

 Share

 Tweet

- Einführung in ggplot2

# Installieren des Paketes

- Zur Erstellung der Karten brauchen wir das Paket `ggmap`:

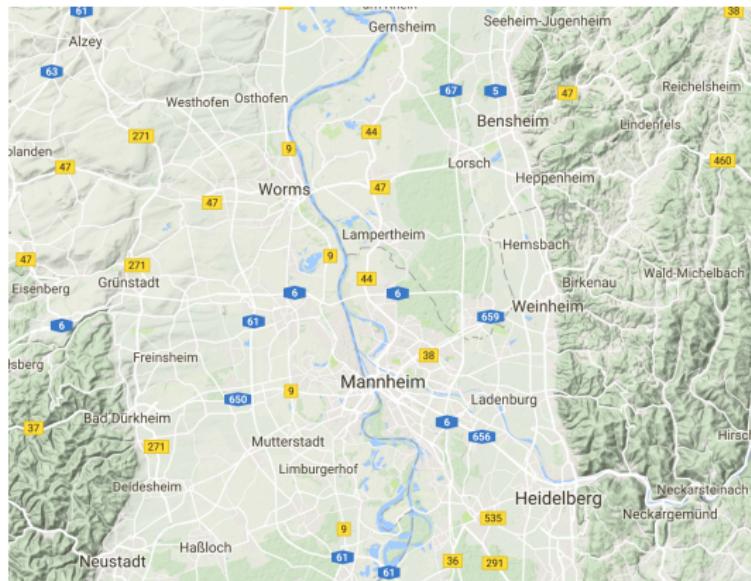
```
install.packages("ggmap")
```

# Paket ggmap - Hallo Welt

```
library(ggmap)
```

Und schon kann die erste Karte erstellt werden:

```
qmap("Mannheim")
```



# *Zoom level bei ggmap*

- level 3 - Kontinent
- level 10 - Stadt
- level 21 - Gebäude

```
qmap("Germany", zoom = 6)
```



# Hilfe bekommen wir mit dem Fragezeichen

?qmap

Verschiedene Abschnitte in der Hilfe:

- Description
- Usage
- Arguments
- Value
- Author(s)
- See Also
- Examples

# Die Beispiele in der Hilfe

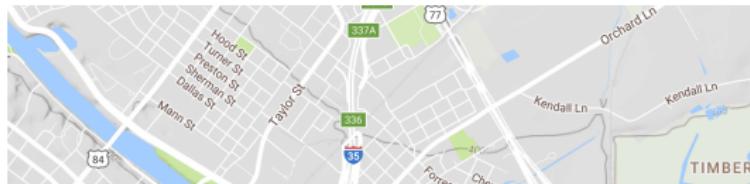
Ausschnitt aus der Hilfe Seite zum Befehl qmap:

## Examples

```
## Not run:  
# these examples have been excluded for checking efficiency  
  
qmap(location = "baylor university")  
qmap(location = "baylor university", zoom = 14)  
qmap(location = "baylor university", zoom = 14, source = "osm")
```

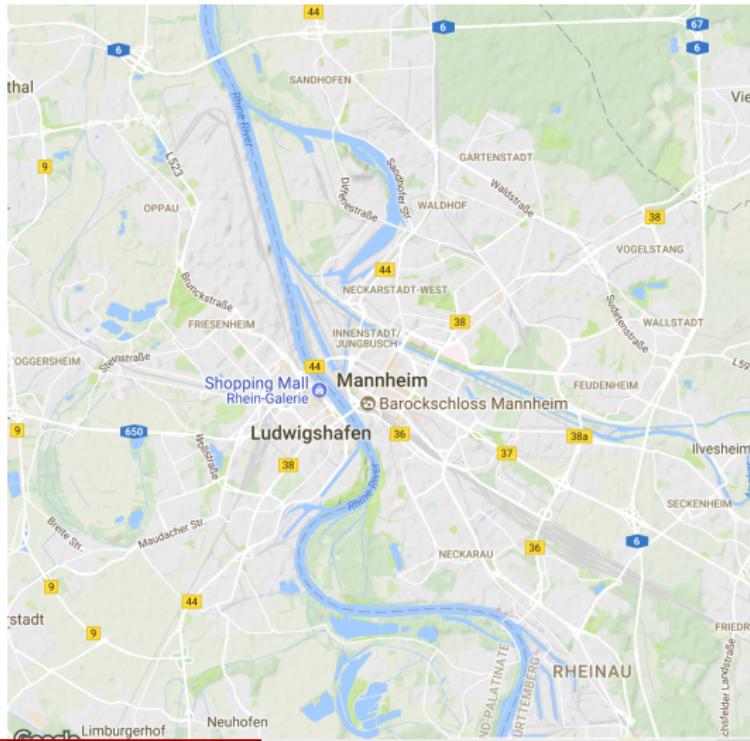
Das Beispiel kann man direkt in die Konsole kopieren:

```
# qmap("baylor university")  
qmap("baylor university", zoom = 14)
```



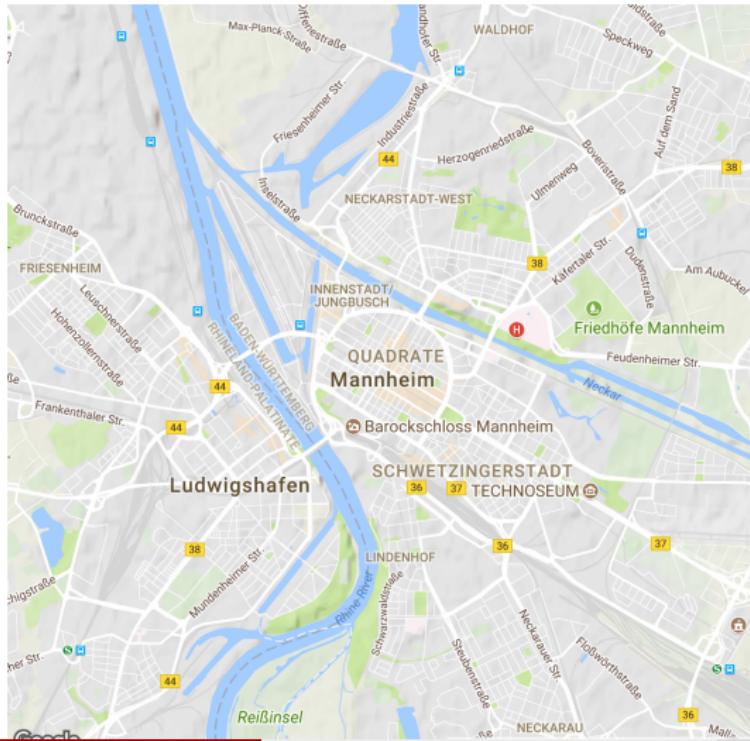
# Ein anderes *zoom level*

`qmap("Mannheim", zoom = 12)`



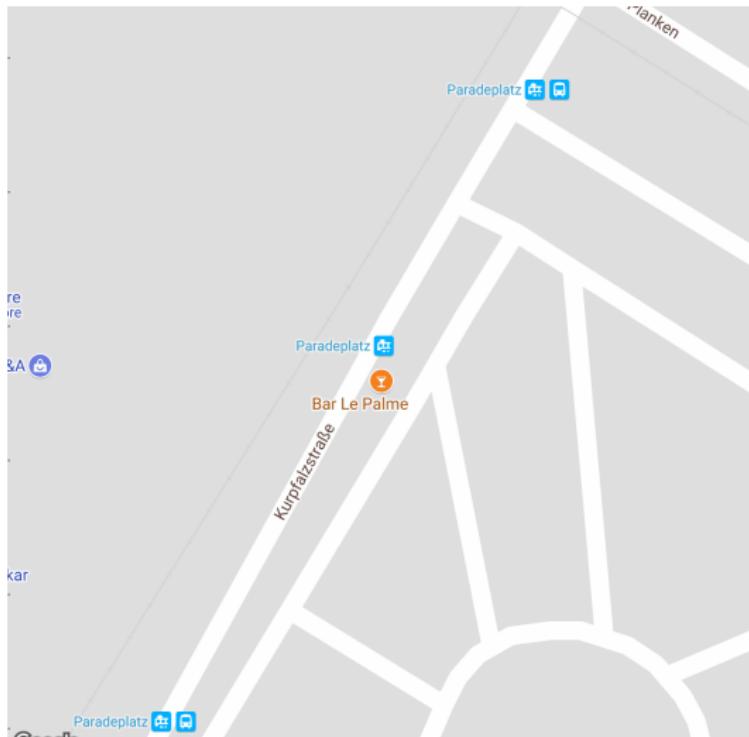
# Näher rankommen

```
qmap('Mannheim', zoom = 13)
```



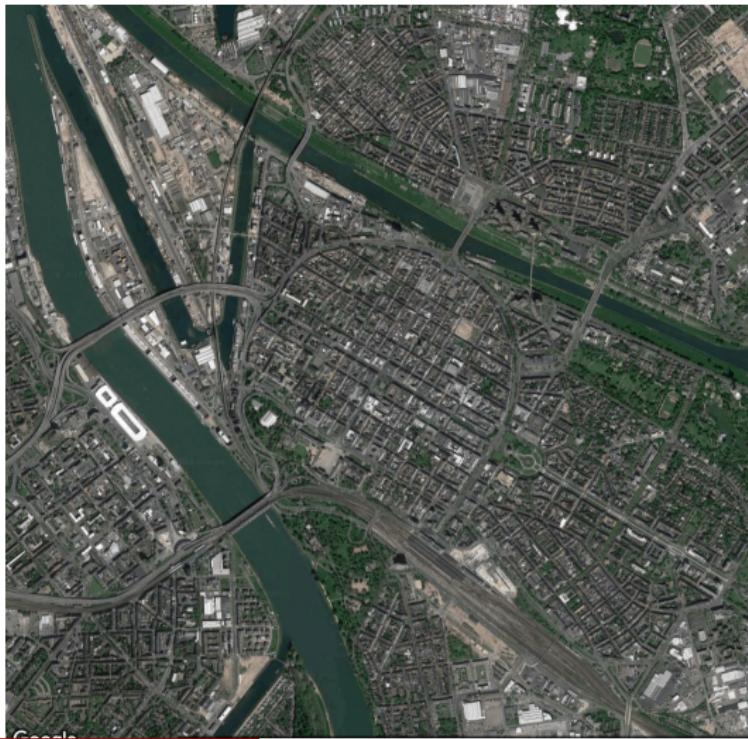
# Ganz nah dran

```
qmap('Mannheim', zoom = 20)
```



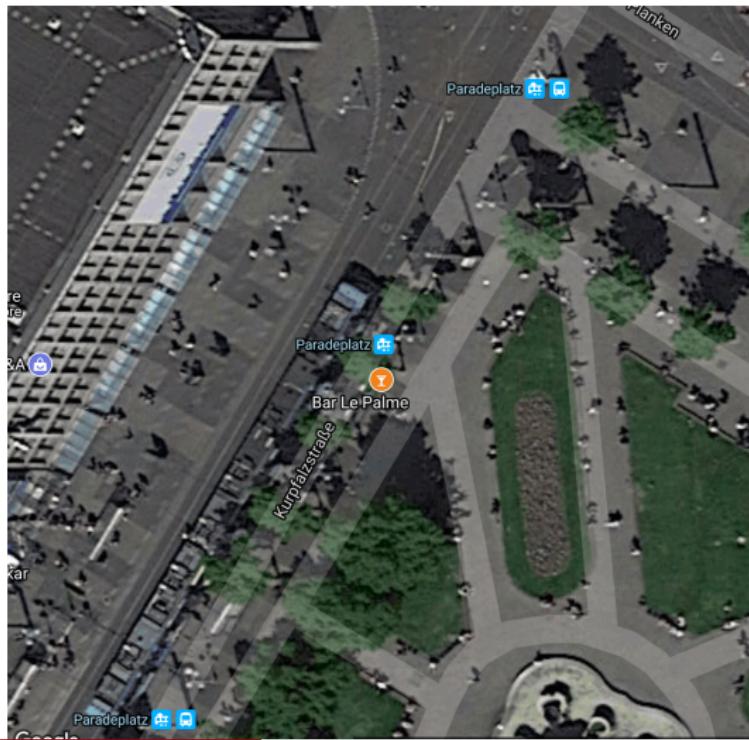
## ggmap - maptype satellite

```
qmap('Mannheim', zoom = 14, maptype="satellite")
```



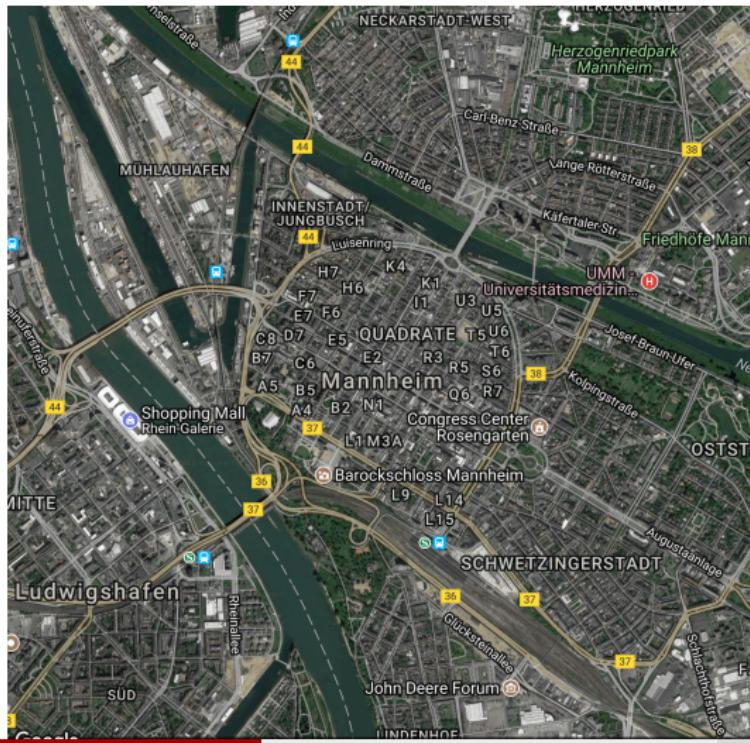
## ggmap - maptype satellite zoom 20

```
qmap('Mannheim', zoom = 20, maptype="hybrid")
```



# ggmap - maptype hybrid

```
qmap("Mannheim", zoom = 14, maptype="hybrid")
```

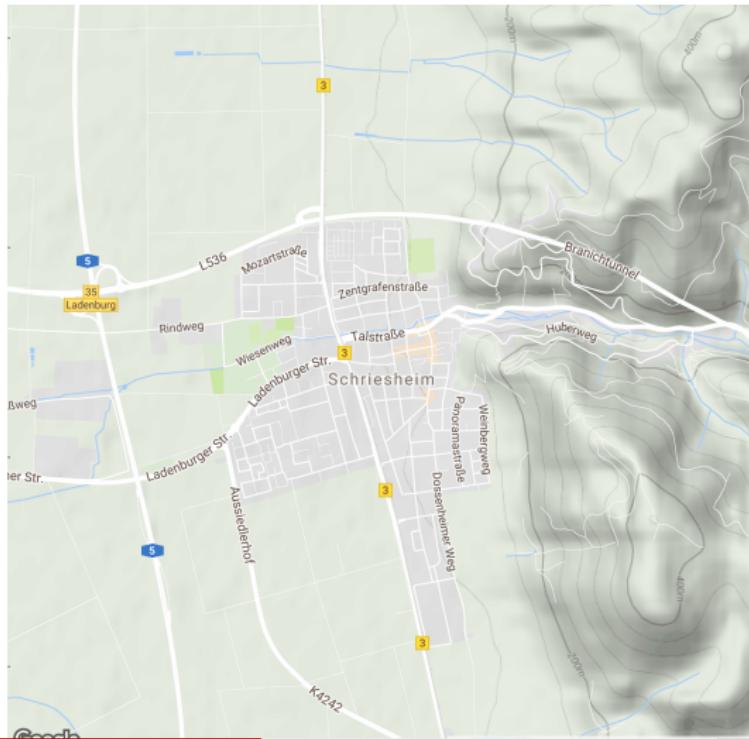


# Terrain/physical maps

- Aus Physischen Karten kann man Informationen über Berge, Flüsse und Seen ablesen.
- Farben werden oft genutzt um Höhenunterschiede zu visualisieren

# ggmap - terrain map

```
qmap('Schriesheim', zoom = 14, maptype="terrain")
```



## Abstrahierte Karten)



- Abstraktion wird genutzt um nur essentielle Informationen zu zeigen.
- Bsp. U-Bahn Karten - wichtig sind Richtungen und wenig Infos zur Orientierung
- Nun kommen Karten, die sich als Hintergrund eignen.

## ggmap - maptype watercolor

```
qmap('Mannheim', zoom = 14, maptype="watercolor", source="stamen")
```



## ggmap - source stamen

```
qmap('Mannheim', zoom = 14,  
      maptype="toner",source="stamen")
```



## ggmap - maptype toner-lite

```
qmap('Mannheim', zoom = 14,  
      maptype="toner-lite", source="stamen")
```



## ggmap - maptype toner-hybrid

```
qmap('Mannheim', zoom = 14,  
      maptype="toner-hybrid",source="stamen")
```

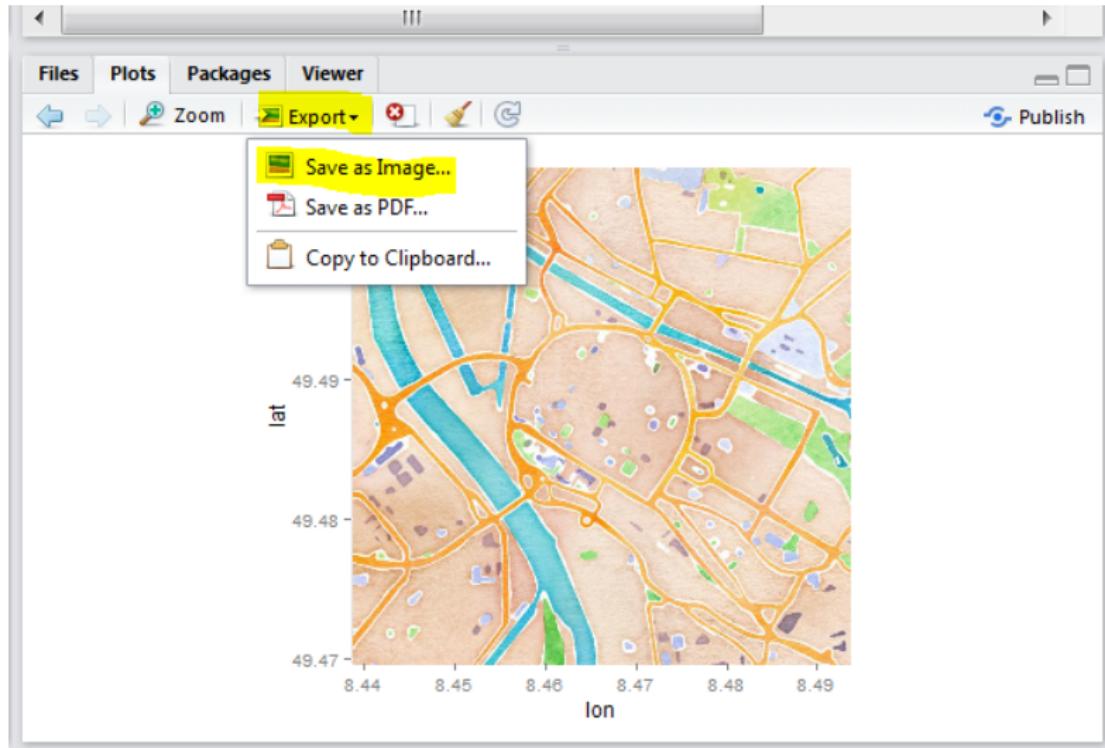


## ggmap - maptype terrain-lines

```
qmap('Mannheim', zoom = 14,  
      maptype="terrain-lines", source="stamen")
```



# Graphiken speichern



## ggmap - ein Objekt erzeugen

- <- ist der Zuweisungspfeil um ein Objekt zu erzeugen
- Dieses Vorgehen macht bspw. Sinn, wenn mehrere Karten nebeneinander gebraucht werden.

```
MA_map <- qmap('Mannheim',
                 zoom = 14,
                 maptype="toner",
                 source="stamen")
```

# Geokodierung

*Geocoding ( . . . ) uses a description of a location, most typically a postal address or place name, to find geographic coordinates from spatial reference data . . .*

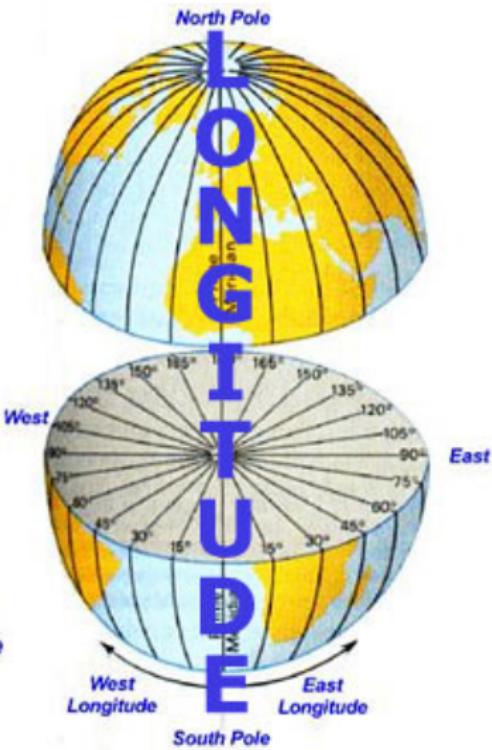
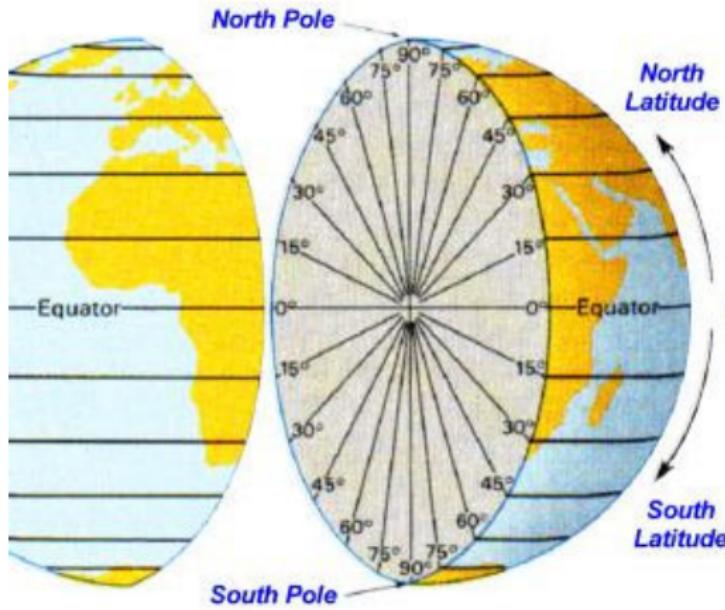
Wikipedia - Geocoding

```
library(ggmap)  
geocode("Mannheim", source="google")
```

lon	lat
8.463182	49.48608

# Latitude und Longitude

## LATITUDE



# Koordinaten verschiedener Orte in Deutschland

cities	lon	lat
Hamburg	9.993682	53.55108
Koeln	6.960279	50.93753
Dresden	13.737262	51.05041
Muenchen	11.581981	48.13513

# Reverse Geokodierung

*Reverse geocoding is the process of back (reverse) coding of a point location (latitude, longitude) to a readable address or place name. This permits the identification of nearby street addresses, places, and/or areal subdivisions such as neighbourhoods, county, state, or country.*

Quelle: Wikipedia

```
revgeocode(c(48,8))
```

```
## [1] "Unnamed Road, Somalia"
```

# Die Distanz zwischen zwei Punkten

```
mapdist("Q1, 4 Mannheim","B2, 1 Mannheim")
```

```
##                 from                  to      m      km      miles seconds
## 1 Q1, 4 Mannheim B2, 1 Mannheim 746 0.746 0.4635644      209
##                 hours
## 1 0.05805556
```

```
mapdist("Q1, 4 Mannheim","B2, 1 Mannheim",mode="walking")
```

```
##                 from                  to      m      km      miles seconds
## 1 Q1, 4 Mannheim B2, 1 Mannheim 546 0.546 0.3392844      423
```

# Eine andere Distanz bekommen

```
mapdist("Q1, 4 Mannheim","B2, 1 Mannheim",mode="bicycling")  
  
##                 from                  to    m      km    miles seconds  
## 1 Q1, 4 Mannheim B2, 1 Mannheim 555 0.555 0.344877     215  
##          hours  
## 1 0.05972222
```

## Geokodierung - verschiedene Punkte von Interesse

```
POI1 <- geocode("B2, 1 Mannheim",source="google")
POI2 <- geocode("Hbf Mannheim",source="google")
POI3 <- geocode("Mannheim, Friedrichsplatz",source="google")
ListPOI <- rbind(POI1,POI2,POI3)
POI1;POI2;POI3

##           lon      lat
## 1 8.462844 49.48569

##           lon      lat
## 1 8.469879 49.47972

##           lon      lat
## 1 8.475754 49.48304
```

# Punkte in der Karte

```
MA_map +  
  geom_point(aes(x = lon, y = lat),  
  data = ListPOI)
```



# Punkte in der Karte

```
MA_map +  
  geom_point(aes(x = lon, y = lat), col="red",  
  data = ListPOI)
```



## ggmap - verschiedene Farben

```
ListPOI$color <- c("A", "B", "C")  
MA_map +  
  geom_point(aes(x = lon, y = lat, col=color),  
  data = ListPOI)
```



## ggmap - größere Punkte

```
ListPOI$size <- c(10,20,30)  
MA_map +  
  geom_point(aes(x = lon, y = lat, col=color, size=size),  
  data = ListPOI)
```



# Eine Route von Google maps bekommen

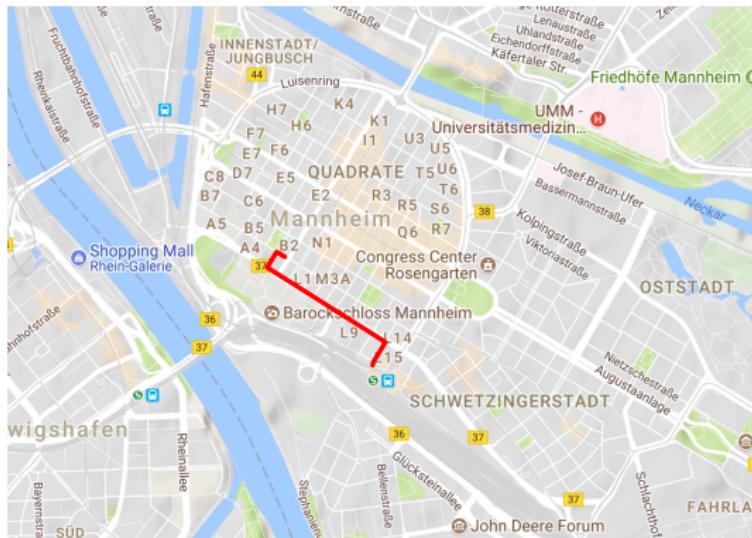
```
from <- "Mannheim Hbf"  
to <- "Mannheim B2 , 1"  
route_df <- route(from, to, structure = "route")
```

Mehr Information

<http://rpackages.ianhowson.com/cran/ggmap/man/route.html>

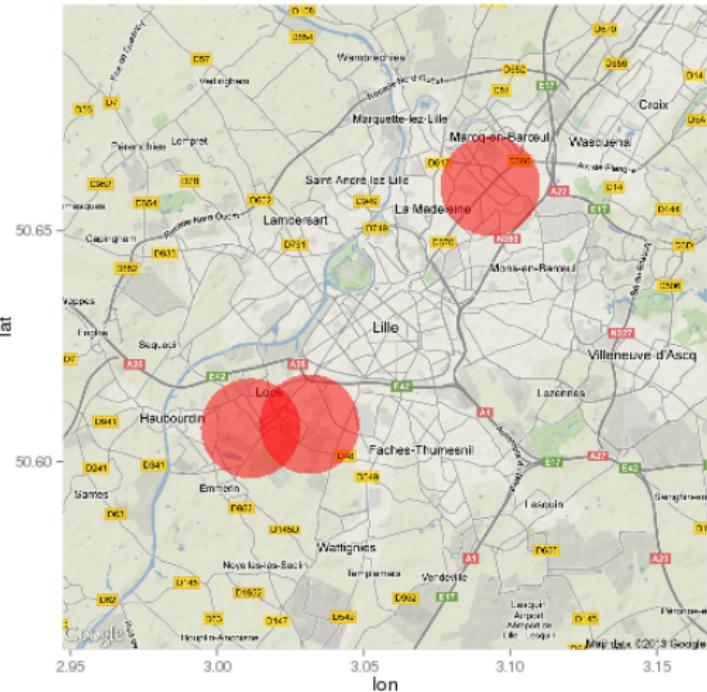
# Eine Karte mit dieser Information zeichnen

```
qmap("Mannheim Hbf", zoom = 14) +  
  geom_path(  
    aes(x = lon, y = lat), colour = "red", size = 1.5,  
    data = route_df, lineend = "round"  
)
```



# Bubbles auf einer Karte

<http://i.stack.imgur.com>



# Cheatsheet

## • Cheatsheet zu data visualisation

<https://www.rstudio.com/>

### Data Visualization with ggplot2

Cheat Sheet

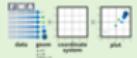


#### Basics

ggplot2 is based on the **grammar of graphics**, the idea that you can build every graph from the same few components: a set of **geom**-visual marks that represent data points, and a **coordinate system**.

#### Coordinate Systems

To display data values, map variables in the data set to aesthetic properties of the geom like **size**, **color**, and **x** & **y** locations.



Build a graph with **ggplot()** or **ggplot2()**

ggplot(mapping = ..., data = ..., geom = ...,   
 ggplot2(mapping = ..., data = ..., stat = ...,   
 geom = ..., position = ...))

Creates a complete plot with given data, geom, and mapping. Supports many visual details.

ggplot(data = mpg, aes(x = cyl, y = hwy))

Begin a plot that you finish by adding layers to. No defaults, but provides more control than ggplot().

ggplot(mpg, aes(hwy, cyl)) +   
 geom\_point() +   
 geom\_smooth(method = "lm") +   
 geom\_text(aes(label = scales::label\_percent(),   
 color = color\_gradient("blue", "red"),   
 size = 3))

Add a new layer to a plot with a **geom**, **stat**, or **stat**, **Q**-function. Each provides a geom, a set of aesthetic mappings, and a default stat and position adjustment.

last\_plot()

Returns the last plot

ggplot("plot.png", width = 5, height = 5)

Jan-Philipp Kolb

**Geoms** - Use a geom to represent data points, use the geom's aesthetic properties to represent variables. Each function returns a layer.

#### One Variable

##### Continuous

a <- ggplot(mpg, aes(hwy))

x, y, alpha, color, fill, linetype, size

b <- geom\_area(stat = "bin")

x, y, alpha, color, fill, linetype, size, weight

c <- geom\_dotplot()

x, y, alpha, color, fill

d <- geom\_freqpoly()

x, y, alpha, color, linetype, size

e <- geom\_hex()

x, y, alpha, color, fill, linetype, size, weight

f <- geom\_histogram()

x, y, alpha, color, fill, linetype, size, weight

g <- geom\_jitter()

x, y, alpha, color, fill, shape, size

h <- geom\_line()

x, y, alpha, color, fill, linetype, size

i <- geom\_point()

x, y, alpha, color, fill, shape, size

j <- geom\_quantile()

x, y, alpha, color, linetype, size, weight

k <- geom\_rug()

alpha, color, linetype, size

l <- geom\_smooth(method = lm)

x, y, alpha, color, fill, linetype, size, weight

m <- geom\_text(label = "y")

x, y, alpha, color, angle, color, family, fontface, fontsize, label, size

n <- geom\_bar(stat = "identity")

x, y, alpha, color, fill, linetype, size, weight

o <- geom\_boxplot()

lower, middle, upper, x, ymin, ymax, alpha, color, fill, linetype, size, weight

p <- geom\_label()

stacked, x, y, alpha, color, fill

q <- geom\_linerange()

x, y, alpha, color, fill, linetype, size

r <- geom\_pointrange()

x, y, alpha, color, fill, linetype, size, weight

s <- geom\_rect()

alpha, color, fill, linetype, size

t <- geom\_segment()

xend, yend, x, y, alpha, color, fill, linetype, size

u <- geom\_vline()

x, y, alpha, color, fill, linetype, size

##### Continuous X, Continuous Y

f <- ggplot(mpg, aes(cyl, hwy))

x, y, alpha, color, fill, linetype, size, weight

##### Two Variables

Continuous Bivariate Distribution

i <- ggplot(movies, aes(year, rating))

x, y, alpha, min, year, rating, alpha, color, fill, linetype, size, weight

j <- geom\_hex()

x, y, alpha, color, fill, size

k <- geom\_hex2d()

x, y, alpha, color, fill, size

##### Continuous Function

l <- ggplot(economics, aes(date, unemploy))

x, y, alpha, color, fill, linetype, size

m <- geom\_line()

x, y, alpha, color, fill, linetype, size

n <- geom\_step(direction = "in")

x, y, alpha, color, fill, linetype, size

##### Discrete X, Continuous Y

o <- ggplot(mpg, aes(carbon, hwy))

x, y, alpha, color, fill, linetype, size, weight

##### Discrete X, Discrete Y

p <- ggplot(diamonds, aes(carat, color))

x, y, alpha, color, fill, linetype, size

q <- geom\_bar(stat = "identity")

x, y, alpha, color, fill, linetype, size, weight

r <- geom\_boxplot()

lower, middle, upper, x, ymin, ymax, alpha, color, fill, linetype, size, weight

s <- geom\_label()

stacked, x, y, alpha, color, fill

t <- geom\_linerange()

x, y, alpha, color, fill, linetype, size

u <- geom\_pointrange()

x, y, alpha, color, fill, linetype, size, weight

v <- geom\_rect()

alpha, color, fill, linetype, size

w <- geom\_segment()

xend, yend, x, y, alpha, color, fill, linetype, size

xend, yend, x, y, alpha, color, fill, linetype, size

xend, yend, x, y, alpha, color, fill, linetype, size

##### Discrete Function

x <- data.frame(state = USArrests\$Murder,

state = USArrests\$Name[USArrests\$ID])

y <- data.frame(y = USArrests\$Murder)

z <- expand\_limits(y = y\$Murder)

map <- ggplot(x, aes(state, map = map))

map <- map + geom\_map(map = map, fill = "white", color = "black")

map <- map + geom\_rect(state, fill = "white", color = "black")

map <- map + geom\_label(state, fill = "white", color = "black")

map <- map + geom\_vline(x0 = 0, x1 = 1, y0 = 0, y1 = 1)

map <- map + geom\_hline(y0 = 0, y1 = 1, x0 = 0, x1 = 1)

map <- map + geom\_point(state, fill = "white", color = "black")

map <- map + geom\_text(state, fill = "white", color = "black")

map <- map + geom\_text(state, fill = "white", color = "black")

map <- map + geom\_text(state, fill = "white", color = "black")

map <- map + geom\_text(state, fill = "white", color = "black")

map <- map + geom\_text(state, fill = "white", color = "black")

map <- map + geom\_text(state, fill = "white", color = "black")

map <- map + geom\_text(state, fill = "white", color = "black")

map <- map + geom\_text(state, fill = "white", color = "black")

map <- map + geom\_text(state, fill = "white", color = "black")

map <- map + geom\_text(state, fill = "white", color = "black")

##### Three Variables

seats <- withouts, script\_seats\_long %>% delta\_set %>%

m <- ggplot(seats, aes(seats, lat))

x, y, alpha, color, fill, linetype, size

n <- geom\_contour(bands = 5, alpha = 0.5,   
 fill = "white", color = "black")

x, y, alpha, fill, linetype, size

R für die Sozialwissenschaften - Teil 1

29 Juli, 2017

262 / 463

# Resourcen und Literatur

- Artikel von David Kahle und Hadley Wickham zur Nutzung von ggmap.
- Schnell eine Karte bekommen
- Karten machen mit R
- Problem mit der Installation von ggmap

# Take Home Message

Was klar sein sollte:

- Wie man eine schnelle Karte erzeugt
- Wie man geokodiert
- Wie man eine Distanz berechnet

# Die lineare Regression

# Die lineare Regression

Maindonald - Data Analysis

- Einführung in R
- Datenanalyse
- Statistische Modelle
- Inferenzkonzepte
- Regression mit einem Prädiktor
- Multiple lineare Regression
- Ausweitung des linearen Modells
- ...

# Lineare Regression in R - Beispieldatensatz

John H. Maindonald and W. John Braun

DAAG - Data Analysis and Graphics Data and Functions

```
install.packages("DAAG")
```

```
library("DAAG")
data(roller)
```

help on roller data:

```
?roller
```

# Das lineare Regressionsmodell in R

Schätzen eines Regressionsmodells:

```
roller.lm <- lm(depression ~ weight, data = roller)
```

So bekommt man die Schätzwerte:

```
summary(roller.lm)
```

```
##  
## Call:  
## lm(formula = depression ~ weight, data = roller)  
##  
## Residuals:  
##      Min       1Q   Median       3Q      Max  
## -8.180 -5.580 -1.346  5.920  8.020  
##  
## Coefficients:  
##              Estimate Std. Error t value Pr(>|t|)  
##
```

# Summary des Modells

```
summary(roller.lm)

##
## Call:
## lm(formula = depression ~ weight, data = roller)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -8.180  -5.580  -1.346   5.920   8.020
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.0871     4.7543  -0.439  0.67227
## weight       2.6667     0.7002   3.808  0.00518 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 '
```

# R arbeitet mit Objekten

- `roller.lm` ist nun ein spezielles Regressions-Objekt
- Auf dieses Objekt können nun verschiedene Funktionen angewendet werden

```
predict(roller.lm) # Vorhersage
```

```
##          1          2          3          4          5          6
## 2.979669 6.179765 6.713114 10.713233 12.046606 14.180002
##          8          9         10
## 18.180121 24.046962 30.980502
```

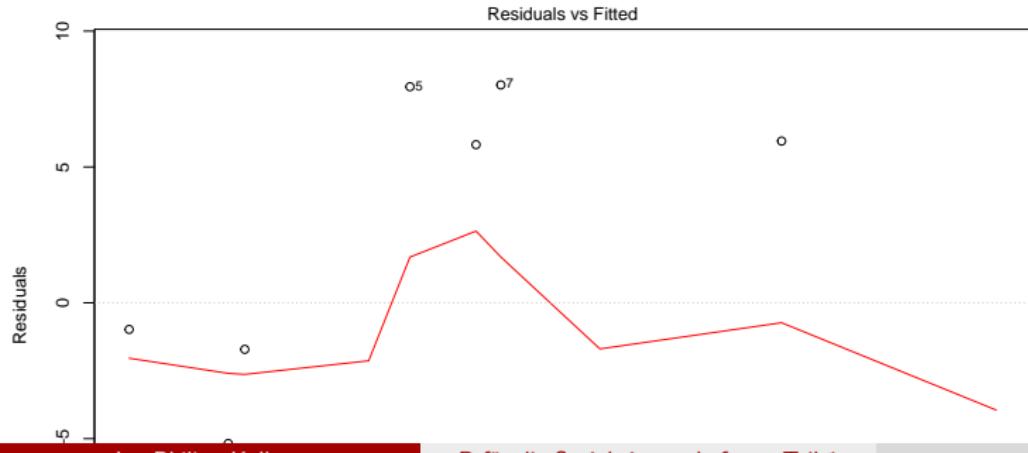
```
resid(roller.lm) # Residuen
```

```
##          1          2          3          4          5          6
## -0.9796695 -5.1797646 -1.7131138 -5.7132327 7.9533944 5.8
##          7          8          9         10
## 8.0199738 -8.1801213 5.9530377 -5.9805017
```

# Residuenplot

- Sind Annahmen des linearen Regressionsmodells verletzt?
- Dies ist der Fall, wenn ein Muster abweichend von einer Linie zu erkennen ist.
- Hier ist der Datensatz sehr klein

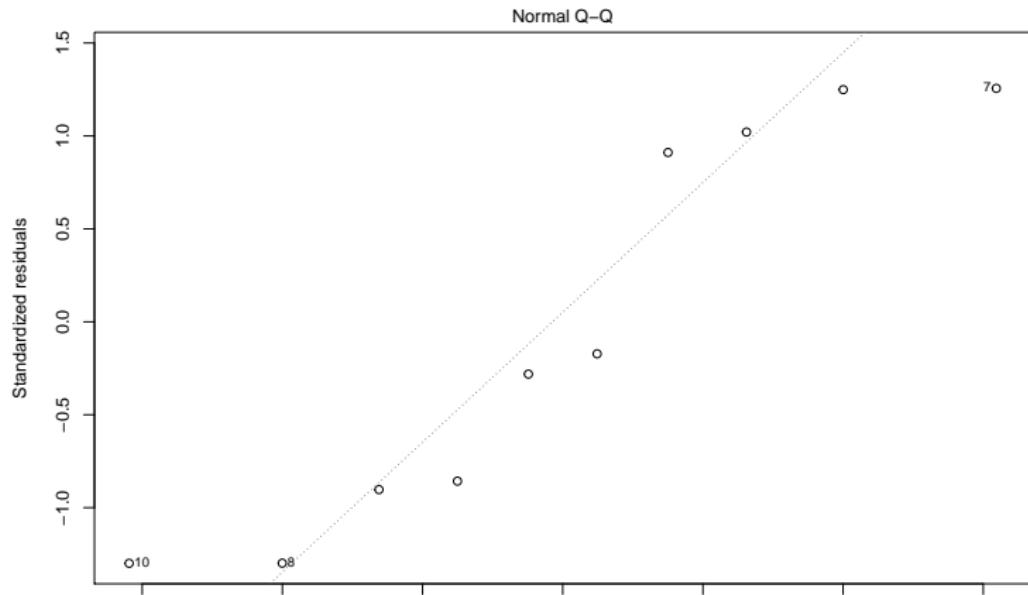
```
plot(roller.lm, 1)
```



# Residuenplot

- Wenn die Residuen normalverteilt sind sollten sie auf einer Linie liegen.

```
plot(roller.lm, 2)
```



# Regressionsdiagnostik mit Basis-R

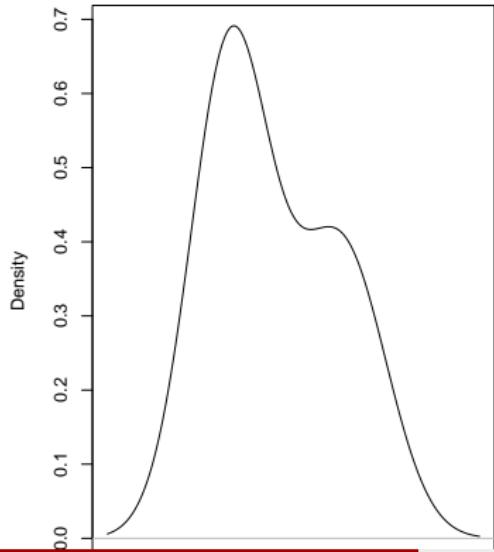
## Ein einfaches Modell

```
N <- 5  
x1 <- rnorm(N)  
y <- runif(N)
```

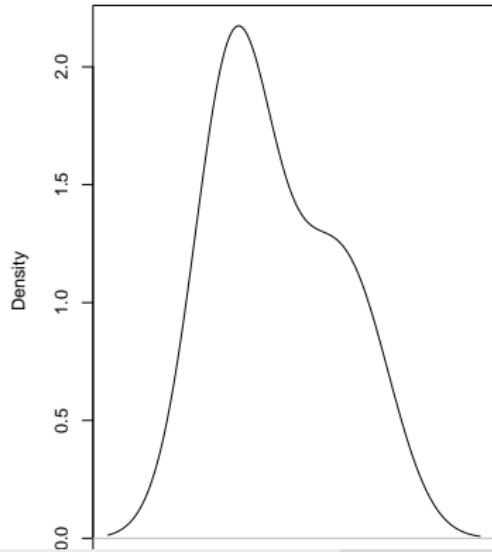
# Die Dichte der beiden Vektoren

```
par(mfrow=c(1,2))  
plot(density(x1))  
plot(density(y))
```

`density.default(x = x1)`



`density.default(x = y)`



# Modellvorhersage machen

```
mod1 <- lm(y~x1)
pre <- predict(mod1)
y

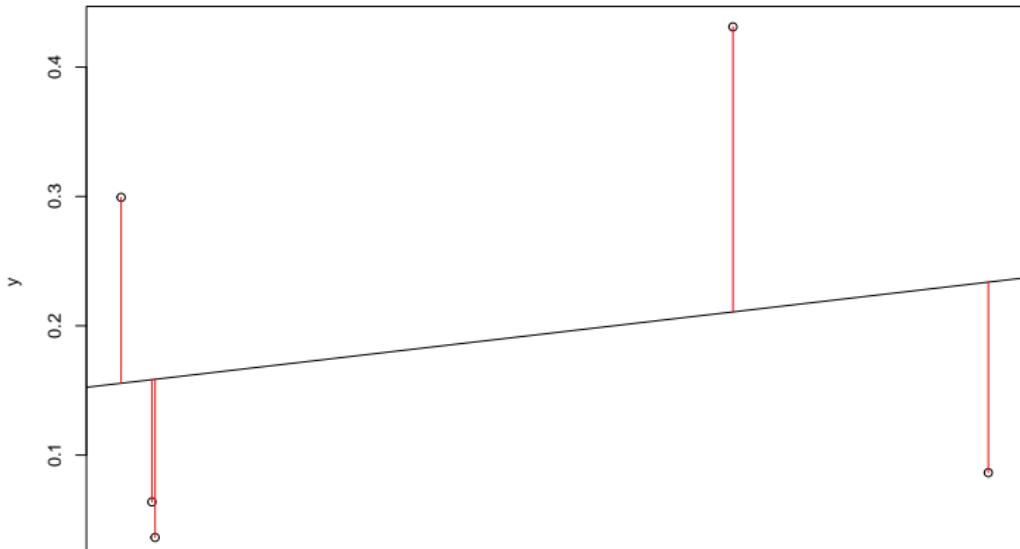
## [1] 0.03633879 0.08638181 0.29930752 0.43115604 0.06377895

pre

##          1           2           3           4           5
## 0.1586083 0.2337576 0.1555483 0.2107252 0.1583237
```

# Regressionsdiagnostik mit Basis-R

```
plot(x1,y)
abline(mod1)
segments(x1, y, x1, pre, col="red")
```



# Beispieldaten Luftqualität

```
library(datasets)
?airquality
```

```
airquality {datasets}
```

## New York Air Quality Measurements

### Description

Daily air quality measurements in New York, May to September 1973.

### Usage

```
airquality
```

### Format

A data frame with 154 observations on 6 variables.

```
[,1] Ozone    numeric Ozone (ppb)
[,2] Solar.R  numeric Solar R (lang)
[,3] Wind     numeric Wind (mph)
[,4] Temp     numeric Temperature (degrees F)
[,5:12] Month   numeric Month (1-12)
```

# Das visreg-Paket

Ein Modell wird auf dem airquality Datensatz geschätzt

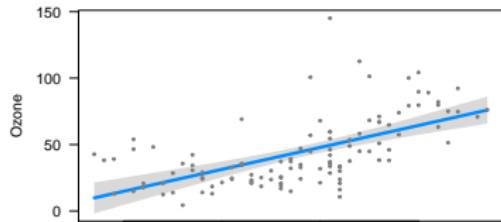
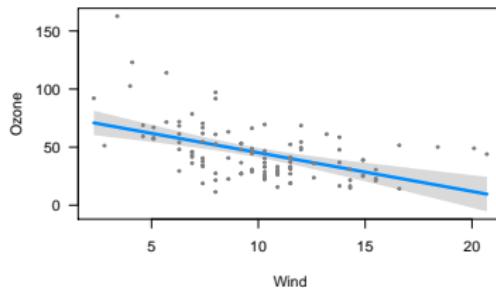
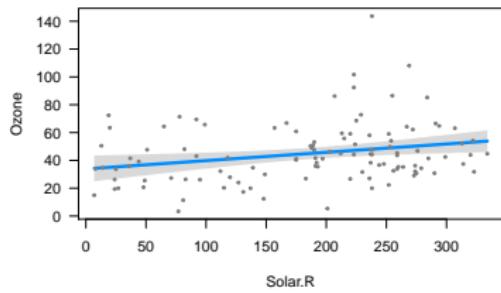
```
install.packages("visreg")

library(visreg)
fit <- lm(Ozone ~ Solar.R + Wind + Temp, data = airquality)
summary(fit)

##
## Call:
## lm(formula = Ozone ~ Solar.R + Wind + Temp, data = airquality)
##
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -40.485  -14.219   -3.551   10.097   95.619 
##
## Coefficients:
## (Intercept)  Solar.R    Wind     Temp  
##            52.9358     0.1239    -0.0087    0.0085
```

# Visualisierung

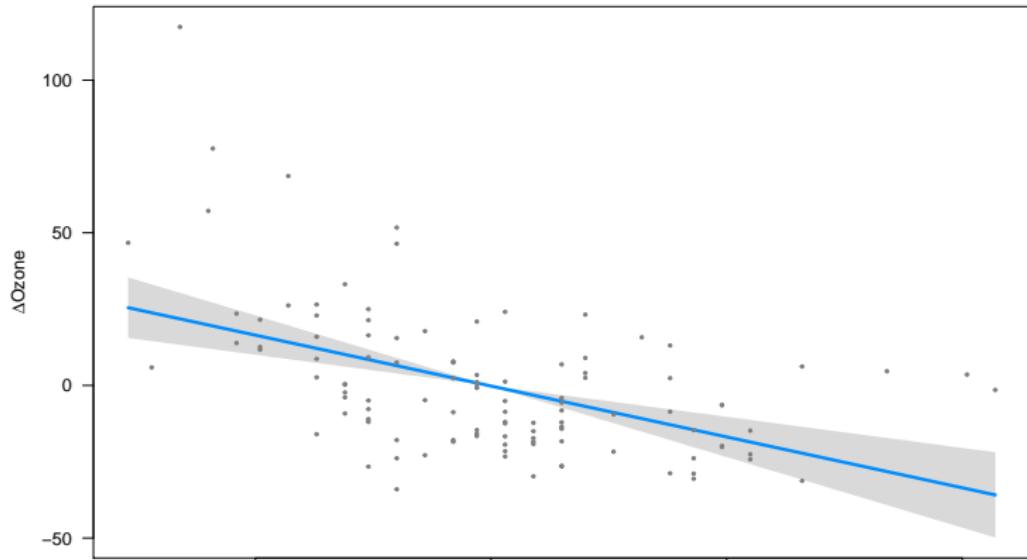
```
par(mfrow=c(2,2))  
visreg(fit)
```



# Und dann mit visreg visualisiert.

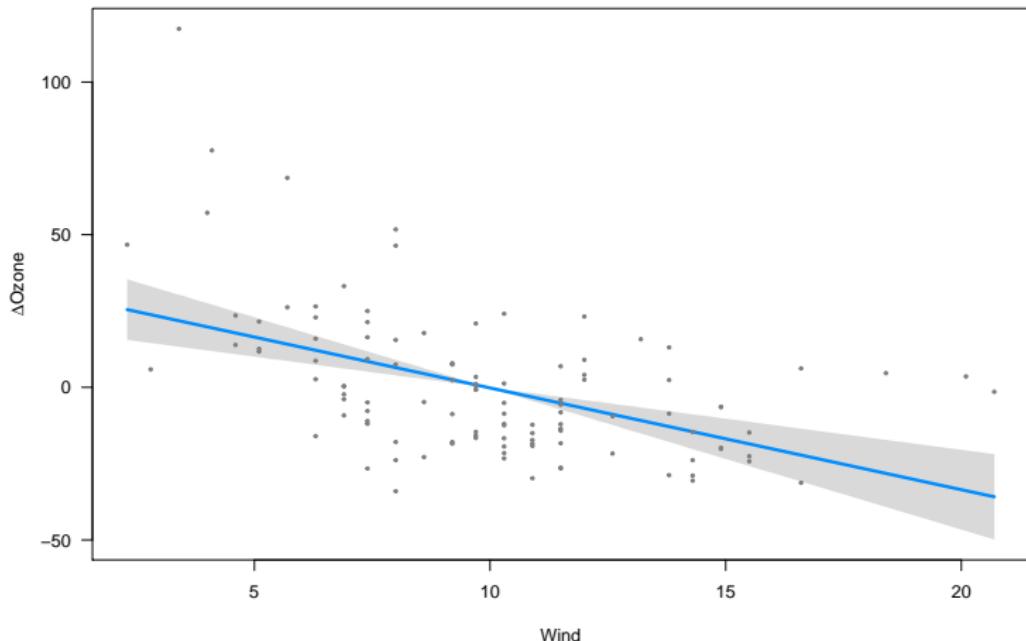
- Zweites Argument - Spezifikation erklärende Variable für Visualisierung

```
visreg(fit, "Wind", type = "contrast")
```



# Visualisierung mit dem Paket visreg

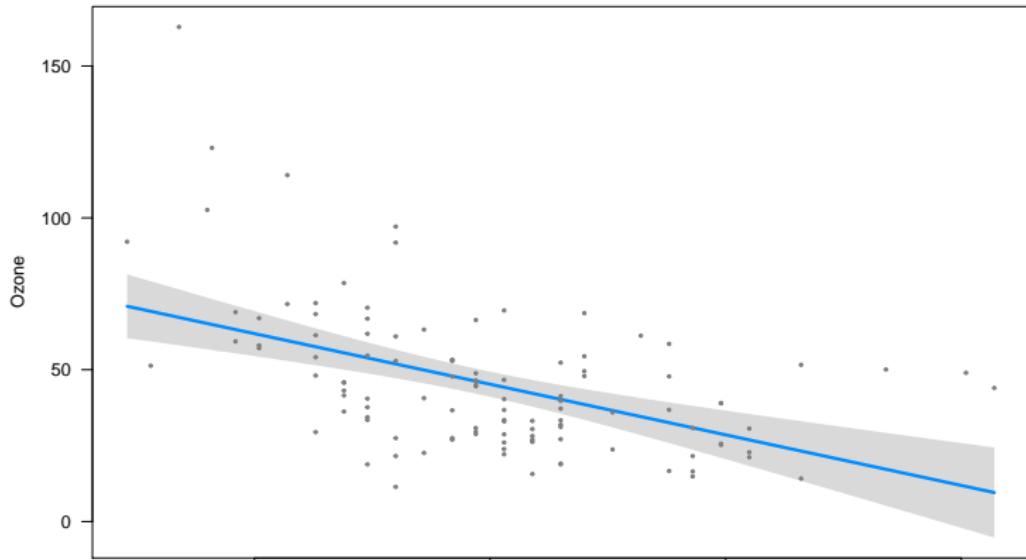
```
visreg(fit, "Wind", type = "contrast")
```



# Das visreg-Paket

- Das Default-Argument für type ist conditional.

```
visreg(fit, "Wind", type = "conditional")
```



# Regression mit Faktoren

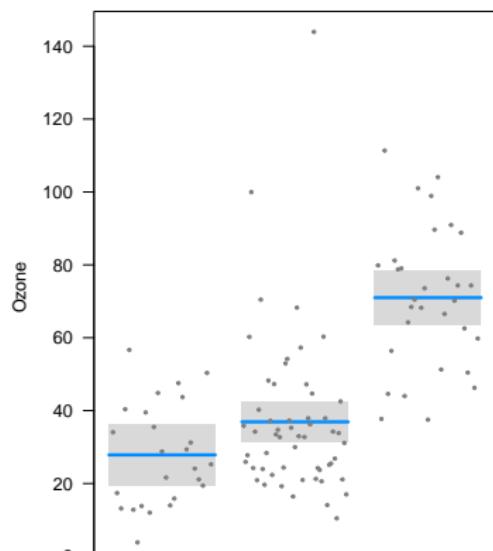
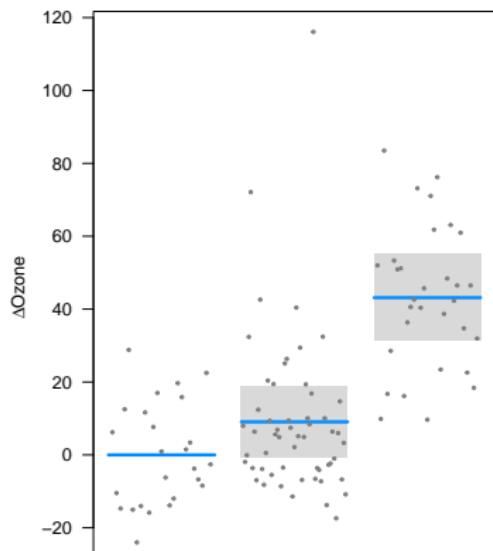
Mit visreg können die Effekte bei Faktoren visualisiert werden.

```
airquality$Heat <- cut(airquality$Temp, 3,
  labels=c("Cool", "Mild", "Hot"))
fit.heat <- lm(Ozone ~ Solar.R + Wind + Heat,
  data = airquality)
summary(fit.heat)

##
## Call:
## lm(formula = Ozone ~ Solar.R + Wind + Heat, data = airquality)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -33.473 -12.794  -2.686   8.461 107.035
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 37.8851    1.8778  20.158  <2e-16 ***
## Solar.R     -0.1239    0.0172 -7.218  1.51e-09 ***
## Wind        -0.0094    0.0015 -6.275  1.02e-09 ***
## Heat        -0.7726    0.1465 -5.300  1.44e-05 ***
```

# Effekte von Faktoren

```
par(mfrow=c(1,2))
visreg(fit.heat, "Heat", type = "contrast")
visreg(fit.heat, "Heat", type = "conditional")
```



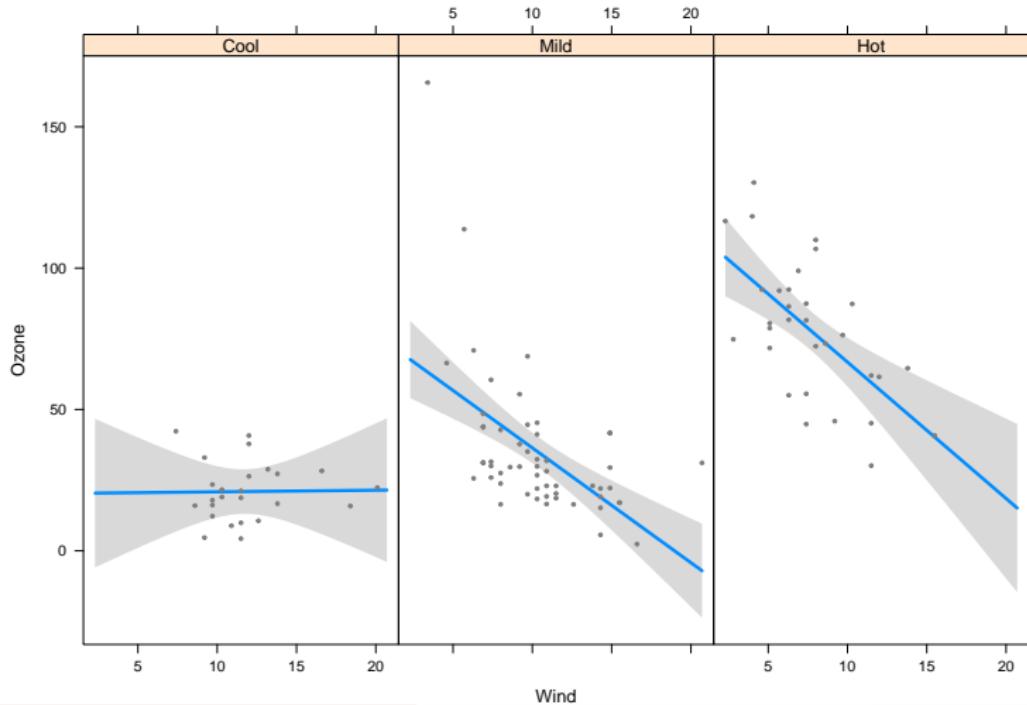
## Das Paket visreg - Interaktionen

```
airquality$Heat <- cut(airquality$Temp, 3,
labels=c("Cool", "Mild", "Hot"))
fit <- lm(Ozone ~ Solar.R + Wind * Heat, data = airquality)
summary(fit)

##
## Call:
## lm(formula = Ozone ~ Solar.R + Wind * Heat, data = airquality)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -34.472 -11.640  -1.919   7.403 102.428
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 4.48042   17.38219   0.258 0.797102
```

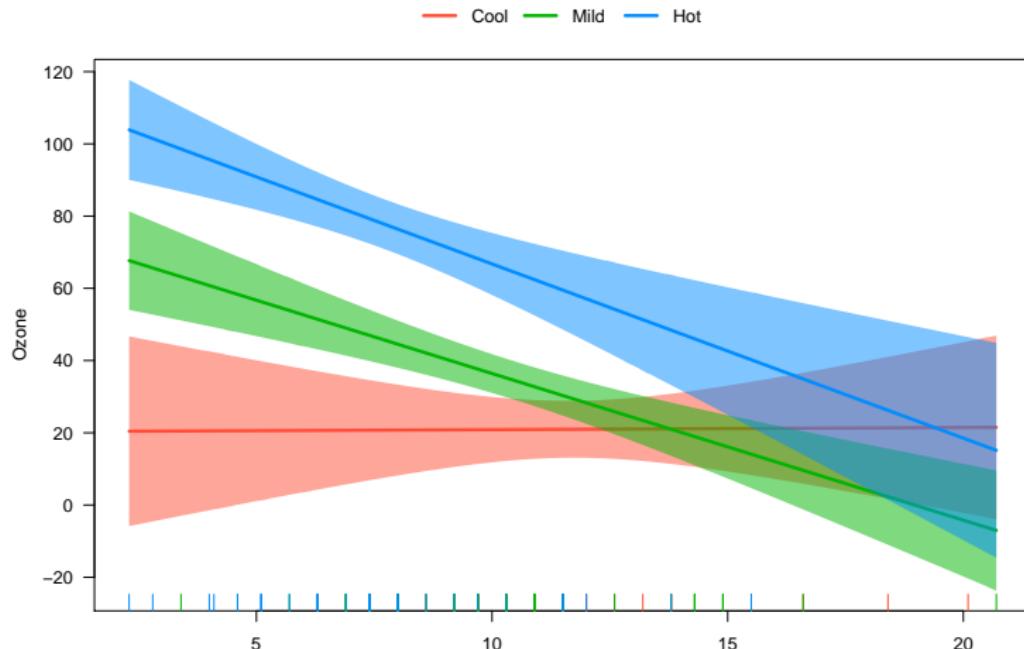
# Steuern der Graphikausgabe mittels layout

```
visreg(fit, "Wind", by = "Heat", layout=c(3,1))
```



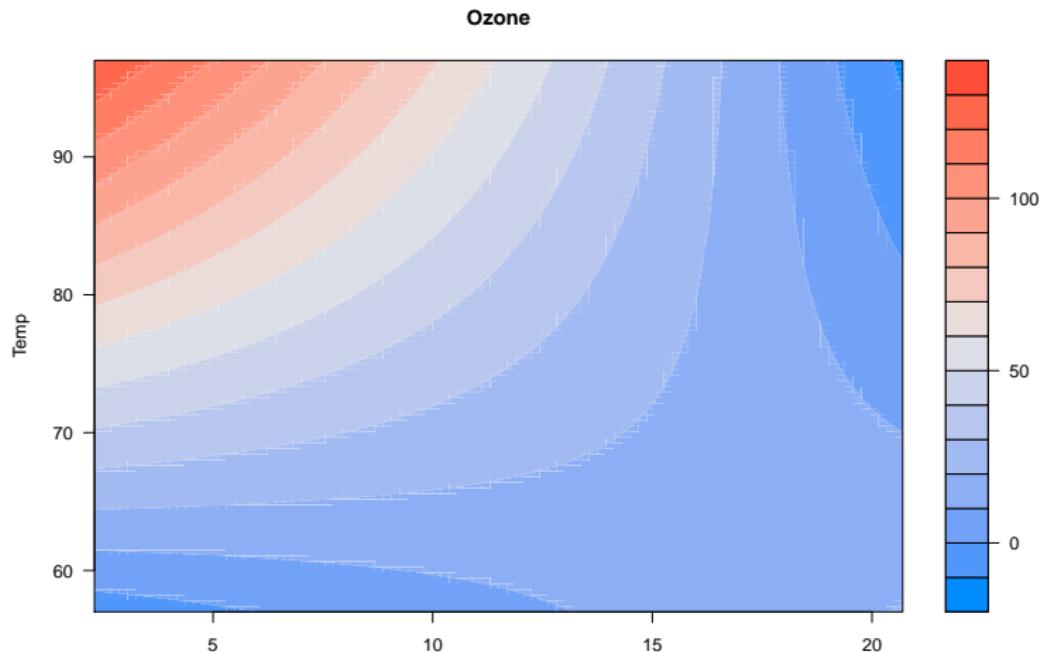
# Das Paket visreg - Interaktionen overlay

```
fit<-lm(Ozone~Solar.R+Wind*Heat,data=airquality)
visreg(fit,"Wind",by="Heat",overlay=TRUE,partial=FALSE)
```



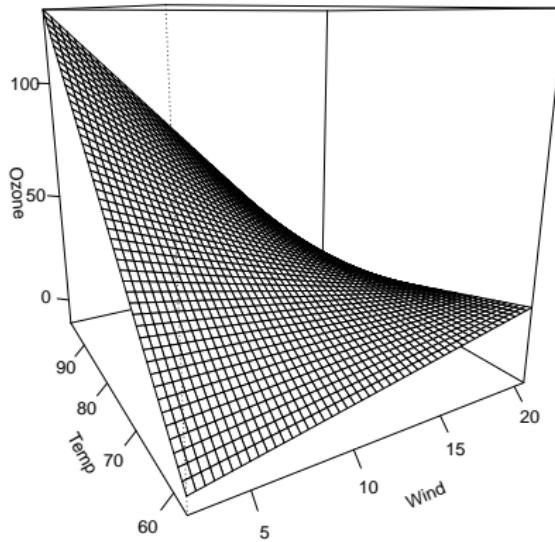
# Das Paket visreg - visreg2d

```
fit2<-lm(Ozone~Solar.R+Wind*Temp,data=airquality)  
visreg2d(fit2,"Wind","Temp",plot.type="image")
```



## Das Paket visreg - surface

```
visreg2d(fit2, "Wind", "Temp", plot.type = "persp")
```



# Regression mit dem survey Paket

```
library(survey)
```

```
data(api)
```

```
head(api.pop)
```

##	cds	stype	name		##	dname	dnum	cname	cnum	flag	pcttest	api00
## 1	01611190130229	H	Alameda	High		Alameda						
## 2	01611190132878	H	Encinal	High		Encinal						
## 3	01611196000004	M	Chipman	Middle		Chipman						
## 4	01611196090005	E	Lum (Donald D.)	Lum (Donald D.)	Elem							
## 5	01611196090013	E	Edison	Elementa	Edison	Edison						
## 6	01611196090021	E	Otis (Frank)	El	Otis (Frank)	Otis (Frank)						
##												
## 1	Alameda City Unified	6	Alameda	1	NA	96	731					
## 2	Alameda City Unified	6	Alameda	1	NA	99	622					
## 3	Alameda City Unified	6	Alameda	1	NA	99	622					
## 4	Alameda City Unified	6	Alameda	1	NA	99	774					

# Das Survey Design spezifizieren

```
dstrat<-svydesign(id=~1,strata=~stype, weights=~pw,
                    data=apistrat, fpc=~fpc)
```

Die Regression rechnen

```
summary(svyglm(api00~ell+meals+mobility,
                design=dstrat))
```

```
##  
## Call:  
## svyglm(formula = api00 ~ ell + meals + mobility, design = dstrat)  
##  
## Survey design:  
## svydesign(id = ~1, strata = ~stype, weights = ~pw, data = apistrat,  
##           fpc = ~fpc)  
##  
## Coefficients:
```

# Linkliste - lineare Regression

- Regression - r-bloggers
- Das Komplette Buch von Faraway- sehr intuitiv geschrieben.
- Gute Einführung auf Quick-R
- Multiple Regression
- Basis Regression - How to go about interpreting regression coefficients

## Aufgabe - lineare Regression

Beschrieben wird Wegstrecke, dreier Spielzeugautos die in unterschiedlichen Winkeln Rampe herunterfuhren.

- angle: Winkel der Rampe
  - distance: Zurückgelegte Strecke des Spielzeugautos
  - car: Autotyp (1, 2 oder 3)
- ① Lesen Sie den Datensatz `toycars` (Paket DAAG) in einen `dataframe` `data` ein und wandeln Sie die Variable `car` des Datensatzes in einen Faktor (`as.factor`) um.
  - ② Erstellen Sie drei Boxplots, die die zurückgelegte Strecke getrennt nach dem Faktor `car` darstellen.
  - ③ Schätzen Sie für die Autos die Parameter des folgenden linearen Modells mit Hilfe der Funktion `lm()`

$$distance_i = \beta_0 + \beta_1 \cdot angle_i + \epsilon_i$$

# Die logistische Regression



- Sehr intuitiv geschriebenes Buch
- Sehr ausführliches begleitendes Skript von Thompson
- Das Skript eignet sich um die kategoriale Datenanalyse nachzuvollziehen

# Faraway Bücher zu Regression in R

Texts in Statistical Science

## Linear Models with R

- Logistische Regressionen gut erklärt
- Beispiele mit R-code
  - Faraway - Extending the linear model with r
  - Faraway - Practical Regression and Anova using R

# Binäre AVs mit `glm`

- Die logistische Regression gehört zur Klasse der generalisierten linearen Modelle (GLM)
- Die Funktion zur Schätzung eines Modells dieser Klasse in heißt `glm()`
- `glm()` muss 1. ein Formel-Objekt mitgegeben werden und 2. die Klasse (binomial, gaussian, Gamma) samt link-Funktion (logit, probit, cauchit, log, cloglog)

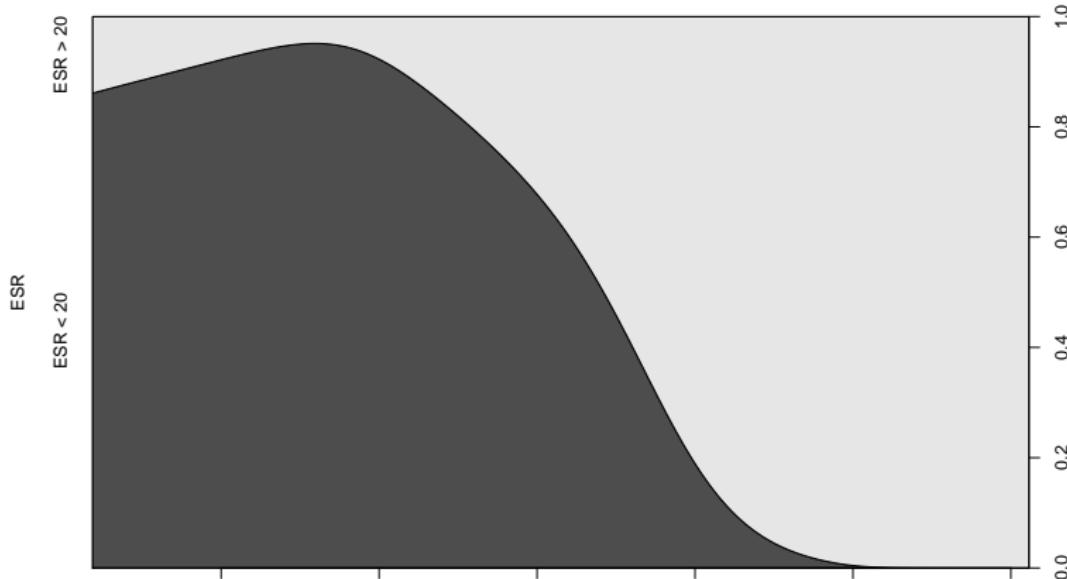
# Beispieldaten für die logistische Regression

```
install.packages("HSAUR")  
  
library("HSAUR")  
data("plasma", package = "HSAUR")
```

# Logistische Regression mit R

- Kategoriale Daten:

```
cdplot(ESR ~ fibrinogen, data = plasma)
```



# Logistische Regression mit R

```
plasma_glm_1 <- glm(ESR ~ fibrinogen, data = plasma,  
                      family = binomial())
```

## Weitere Beispieldaten

```
install.packages("faraway")
```

```
library("faraway")
```

```
data(orings)
```

temp	damage
53	5
57	1
58	1

# Generalisierte Regression mit R - weitere Funktionen

- Logistisches Modell mit Probit-Link:

```
probitmod <- glm(cbind(damage, 6-damage) ~ temp,  
                  family=binomial(link=probit), orings)
```

- Regression mit Zähldaten:

```
modp <- glm(Species ~ ., family=poisson, gala)
```

- Proportional odds logistic regression im Paket MASS:

```
library("MASS")  
house.plr<-polr(Sat~Infl, weights=Freq, data=housing)
```

# Linkliste - logistische Regression

- Einführung in logistische Regression



- Code zum Buch von Faraway



```
library(faraway)
data(orings)
plot(damage/6 ~ temp, orings, xlim=c(25,85), ylim = c(0,1), xlab="Temperature", ylab="Prob of damage")
lmod <- lm(damage/6 ~ temp, orings)
abline(lmod)
logitmod <- glm(cbind(damage,6-damage) ~ temp, family=binomial, orings)
summary(logitmod)
plot(damage/6 ~ temp, orings, xlim=c(25,85), ylim = c(0,1), xlab="Temperature", ylab="Prob of damage")
```

# Mehrebenenmodelle

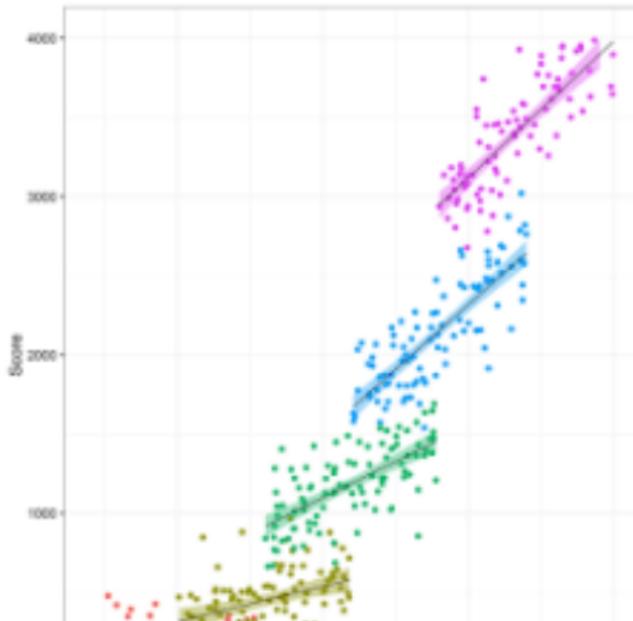
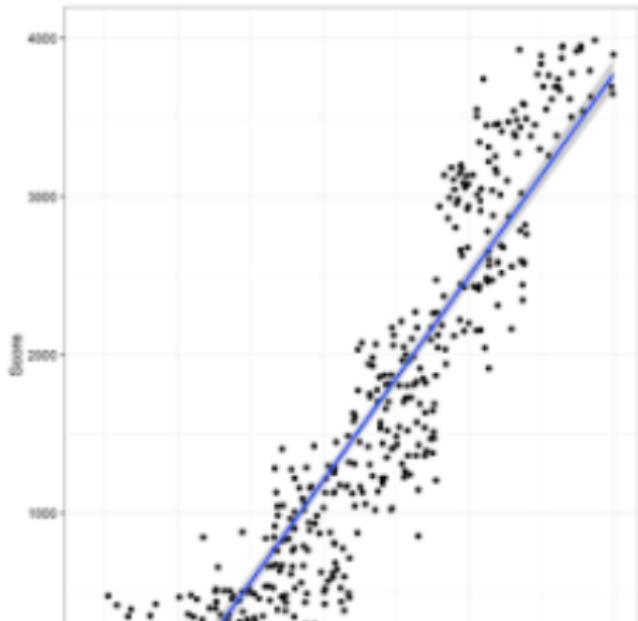
# Wie sehen die Daten aus?

- Beispiel Mehrebenenstruktur der Daten



# Andres Gutierrez - Multilevel Modeling of Educational Data using R

- Lineare Modelle erkennen den Cluster-Effekt aufgrund der Intraklassen Korrelation nicht



# Beispiel Mehrebenenmodelle

## Untersuchungsgegenstand

- Es sollen die Kenntnisse (Fähigkeiten) von Grundschülern in Mathematik gemessen werden.
- Dazu werden in einem Schulbezirk zunächst Schulen ausgewählt und anschließend Klassen.
- Innerhalb der Klassen soll schließlich jeweils eine Stichprobe gezogen und diese getestet werden.
- Geht man zunächst von einer zufälligen Auswahl von Klassen aus, dann ist die Level-1-Variation durch die Schüler und die Level-2-Variation durch die Klassen bestimmt.

## Fragen hierzu

- Wie wäre die Auswahl der Schulen zu berücksichtigen?
- Wie kann zusätzlich eine Unterscheidung nach privaten und staatlichen Schulen in die Modellierung eingebracht werden?

# Beispiel in Goldstein (2010), Kapitel 1.2

Evaluierung der Effektivität von Schulen

Mehrebenen-Modelle:

- Schüler
- Klassenverbände
- Schulamtsbezirke oder Bundesländer

Unterscheidung

- Modelle mit vielen Parametern, die wiederum modelliert werden können
- Regressionen mit Koeffizienten, die zwischen Gruppen variieren können

# Bibliotheken

```
# Linear Mixed-Effects Models using 'Eigen' and S4  
install.packages("lme4")  
  
# Data Visualization for Statistics in Social Science  
install.packages("sjPlot")
```

- Nötige Pakete werden geladen

```
library(ggplot2)  
# Miscellaneous Functions for "Grid" Graphics  
library(gridExtra)  
library(lme4)  
library(sjPlot)  
# A Grammar of Data Manipulation  
library(dplyr)
```

# Beispieldaten

```
mlexdat <- read.csv(  
  "https://github.com/Japhilko/RSocialScience/  
  raw/master/data/mlexdat.csv")
```

X	SES	Score	ID
1	18.62733	-55.120574	A
2	33.64915	-92.375273	A
3	22.26931	-48.783447	A
4	36.49052	38.099329	A
5	38.21402	339.701754	A
6	11.36669	2.286978	A

# Formalistisch

- Bei der Analyse von Daten mit diesen hierarchischen Strukturen, sollte man immer zunächst ein Null-Modell anpassen
- Somit kann man die Variation erfassen, die auf die Schulen zurückzuführen ist.
- Das passende Modell sieht folgendermaßen aus:

$$y_{ij} = \alpha_j + \varepsilon_{ij}$$
$$\alpha_j = \alpha_0 + u_j$$

Die Gesamtvariation wird in zwei Teile zerlegt:

- Variation zwischen Schülern (innerhalb der Schulen) und
- zwischen den Schulen (zwischen den Schulen).

# Der R-code für dieses Nullmodell

- das einfachste Multilevel Modell
- nach dem vertikalen Strich wird die Gruppen Variable spezifiziert
- die Default Schätzmethode ist restricted maximum likelihood (REML)
- Man kann aber auch maximum likelihood Schätzung spezifizieren (ML)

```
HLM0 <- lmer(Score ~ (1 | ID), data = mlexdat)
```

# Nullmodell Ergebnis - Koeffizienten

`coef(HLMO)`

```
## $ID
##   (Intercept)
## A     45.7893
## B     430.7218
## C    1182.1760
## D    2145.2329
## E    3489.1408
##
## attr(,"class")
## [1] "coef.mer"
```

# Nullmodell Ergebnis - Zusammenfassung

`summary(HLMO)`

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: Score ~ (1 | ID)
##   Data: mlexdat
##
## REML criterion at convergence: 7130.6
##
## Scaled residuals:
##       Min      1Q  Median      3Q     Max
## -2.74559 -0.69317 -0.00757  0.68337  2.96511
##
## Random effects:
##   Groups   Name        Variance Std.Dev.
##   ID       (Intercept) 1931758  1389.9
##   Residual           87346    295.5
```

# Interpretation des Nullmodells

- 96 Prozent Variation zwischen den Schulen
- 4 Prozent Variation innerhalb der Schulen

$100 * 87346 / (87346 + 1931757)$

## [1] 4.32598

- Die Schätzung der zufälligen Effekte zeigt, dass die Variation zwischen den Schulen (Intraklassen Korrelation) fast 96 Prozent beträgt
- Während der Anteil der Variation zwischen den Studierenden nur etwas mehr als 4 Prozent ausmacht.
- Das Null-Modell behauptet also , dass Leistungsträger zu bestimmten Schulen gehen und Studierende mit geringerem Leistungsniveau nicht diese Schulen besuchen.
- Mit anderen Worten, die Schule bestimmt das Testergebnis.

## Ein weiteres Modell

- Das Null Modell schließt keine erklärenden Variablen ein.
- Allerdings könnte der sozioökonomischen Status (SES) der Schüler auch eine Rolle spielen.
- Die folgenden Ausdrücke geben ein verfeinertes Modell mit zufälligen Achsenabschnitten und Steigung für jede der Schulen.

## Rcode für dieses Modell

```
HLM1 <- lmer(Score ~ SES + (SES | ID), data = mlexdat)
coef(HLM1)

## $ID
##   (Intercept)      SES
## A   36.46401  0.3798185
## B   37.21549  9.7596237
## C   38.10719 20.8897245
## D   38.85566 30.2320132
## E   39.70159 40.7907386
##
## attr(,"class")
## [1] "coef.mer"
```

## Zusammenfassung zweites Modell

`summary(HLM1)`

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: Score ~ SES + (SES | ID)
##   Data: mlexdat
##
## REML criterion at convergence: 6742.1
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -2.83274 -0.64740  0.02662  0.69063  2.67309
##
## Random effects:
##   Groups   Name        Variance Std.Dev. Corr
##   ID       (Intercept)  1.65     1.285
##           SES         257.09   16.034  1.00
```

```
# 1% - BS variance  
# 99% - WS variance  
100 * 40400.24 / (40400.24 + 257.09 + 1.65)  
  
## [1] 99.36363  
  
# Percentage of variation explained by SES between schools  
1 - ((257.09 + 1.65) / 1931757)  
  
## [1] 0.9998661  
  
# Percentage of variation explained by SES within schools  
1 - (40400.24 / 87346)  
  
## [1] 0.5374689
```

- die Variable SES erklärt 99 Prozent der Unterschiede zwischen den Schulen
- diese Variable SES erklärt 53 Prozent der Abweichungen innerhalb der Schulen.

# Was heißt das? - Schulsegregation

- wohlhabende Studenten gehören zu reichen Schulen
- arme Studenten gehören zu armen Schulen.
- Die Leistung der wohlhabenden Studenten ist höher als die der armen Studenten.

# Ein weiteres Beispiel zur Spezifikation von Multilevel Modellen

- benötigte Bibliotheken:

```
library(lme4)
library(mlmRev)
```

# Der Datensatz

```
data(Exam)  
# names(Exam)
```

school	normexam	schgend	schavg	vr	intake	star
1	0.2613242	mixed	0.1661752	mid 50%	bottom 25%	0.61
1	0.1340672	mixed	0.1661752	mid 50%	mid 50%	0.20
1	-1.7238820	mixed	0.1661752	mid 50%	top 25%	-1.36
1	0.9675862	mixed	0.1661752	mid 50%	mid 50%	0.20
1	0.5443412	mixed	0.1661752	mid 50%	mid 50%	0.37
1	1.7348992	mixed	0.1661752	mid 50%	bottom 25%	2.18

## Zufälliger Intercept und fixed predictor auf individueller Ebene

- Ein Prädiktor wird auf jeder Ebene hinzugefügt
- Dazu wird die '1' im Nullmodell durch den Prädiktor (hier: standLRT) ersetzen.
- Es wird immer ein Intercept angenommen
- Da wir nicht wollen, dass der Effekt des Prädiktors zwischen den Gruppen variiert, bleibt die Spezifikation des zufälligen Teils des Modells mit dem vorherigen Modell identisch.

```
lmer(normexam ~ standLRT + (1 | school), data=Exam)
```

# Random intercept, Random slope

- Modell mit zufälligen Intercept auf individueller Ebene und
- einem Prädiktor, der zwischen Gruppen variieren darf.
- Mit anderen Worten: die Wirkung der Hausaufgaben auf das Ergebnis der Klausur (Mathe-Test) variiert zwischen den Schulen.
- Zur Schätzung wird '1' - der Intercept im zufälligen Teil der Modellspezifikation
- ... durch die Variable ersetzt, von der wir den Effekt zwischen den Gruppen variieren wollen.

## Varying intercept model

```
MLexamp.6<-lmer(extro~open+agree+ social + (1 | school),  
                   data = lmm.data)
```

# Varying slope model

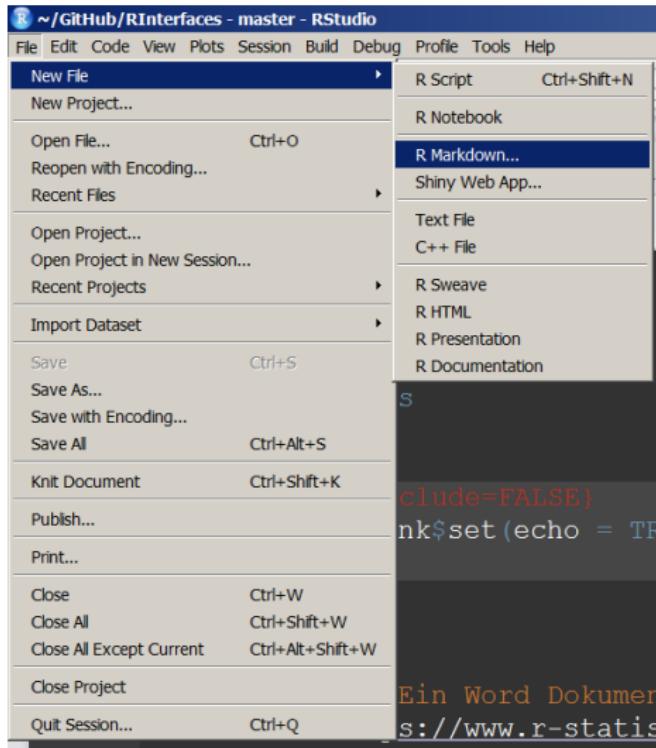
```
MLexamp.9<-lmer(extro~open + agree + social +  
                   (1 + open | school/class),  
                   data = lmm.data)
```

# Links

- Paket lmer
- Uncertainty in parameter estimates using multilevel models
- Multilevel models with R
- Ein Beispieldatensatz
- Multilevel Modeling of Educational Data using R (Part 1)
- Vignette für lme4
- Mixed model guide

# Word Dokumente mit R erstellen

# Ein Markdown Dokument mit Rstudio erzeugen



# Mein erstes mit R erzeugtes Word Dokument

New R Markdown

Document

Presentation

Shiny

From Template

**Title:** Mein Word Dokument

**Author:** Jan-Philipp Kolb

**Default Output Format:**

HTML  
Recommended format for authoring (you can switch to PDF or Word output anytime).

PDF  
PDF output requires TeX (MiKTeX on Windows, MacTeX 2013+ on OS X, TeX Live 2013+ on Linux).

Word  
Previewing Word documents requires an installation of MS Word (or Libre/Open Office on Linux).

# Erstes Beispiel

R2word.docx [Schreibgeschützt] [Kompatibilitätsmodus] - Microsoft Word

AaBbCc AaBbCc

Formatvorlagen ändern \* Markieren \* Bearbeiten

## R2word

Jan-Philipp Kolb

11 April 2017

### Table of Contents

Word mit Pander.....	2
Der Start.....	2
Mein erstes mit R erzeugtes Word Dokument.....	3
Optionen.....	3
Ressourcen.....	3

# Das Arbeiten mit Rmarkdown - erste Schritte

Markdown ist eine sehr einfache Syntax, die es Benutzern erlaubt, aus einfachen Textdateien gut gelayoutete Dokumente zu erstellen.

**\*\*fettes Beispiel\*\***

**\*kursives Beispiel\***

**~~durchgestrichen~~**

**- Aufzählungspunkt**

**fettes Beispiel**

*kursives Beispiel*

*durchgestrichen*

- Aufzählungspunkt

# Weitere Markdown Befehle

### Überschrift Ebene 3

#### Überschrift Ebene 4

[Meine Github Seite] (<https://github.com/Japhilko>)

Überschrift Ebene 3

Überschrift Ebene 4

Meine Github Seite

## Weitere Markdown Befehle

- So kann man Bilder einbinden:
- Man kann entweder einen Link angeben:

! [BSP] ([http://e-scientifics.de/content/example\\_kinderbild.jpg](http://e-scientifics.de/content/example_kinderbild.jpg))

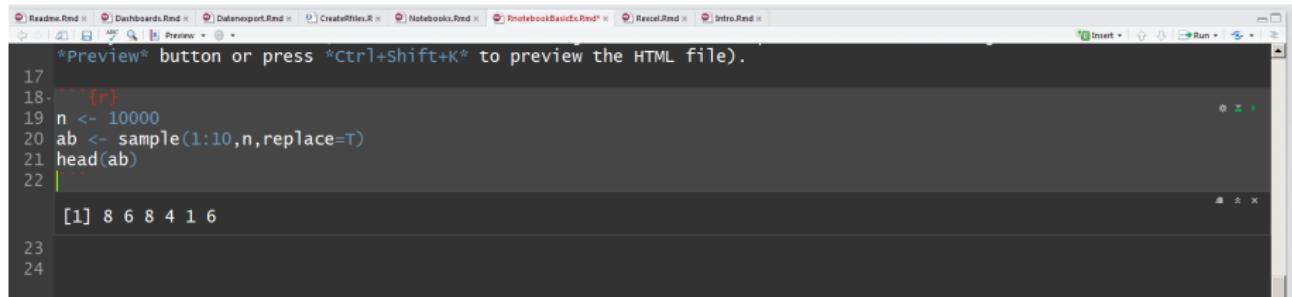
- oder einen (Unterordner) in dem das Bild liegt:

! [BSP 2] (figure/example.png)

- in den eckigen Klammern steht die Bildunterschrift
- alle gängigen Formate (.png, .jpeg,.gif) können so eingebunden werden
- Man kann auch noch weitere Optionen spezifizieren (Größe, Breite etc.) - dazu später mehr

# Chunks - Erste Schritte

- Es lassen sich so genannte Chunks einfügen
- In diesen Chunks wird ganz normaler R-code geschrieben



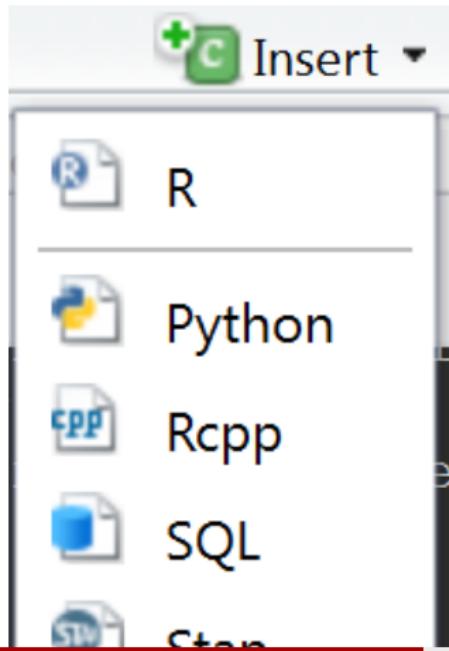
The screenshot shows a RStudio interface with multiple tabs at the top. The active tab is titled "NotebookBasic.Rnw". The main pane displays R code and its output:

```
*Preview* button or press *Ctrl+Shift+K* to preview the HTML file.
17
18 ~~~[r]
19 n <- 10000
20 ab <- sample(1:10,n,replace=T)
21 head(ab)
22
[1] 8 6 8 4 1 6
23
24
```

The code creates a vector 'ab' of length 10000 by sampling from 1:10 with replacement, then prints the first six elements of 'ab'. The RStudio interface includes standard toolbar icons and a menu bar.

## Button um Chunks einzufügen

- Die default Version eines Chunks ist R
- Man hat aber auch die Möglichkeit andere Programmiersprachen einzubinden



# Inline Code

```
n = 100
```

```
# Ein inline Codeblock: `r n`
```

```
n=100
```

Ein inline Codeblock: 100

# Chunk Optionen

- Man kann den Chunks Optionen mitgeben:

Argument	Beschreibung
eval	Soll Rcode evaluiert werden?
warning	Sollen Warnings angezeigt werden?
cache	Soll der Output gespeichert werden?

- Bei eval kann ein logischer Wert angegeben werden oder eine/mehrere Nummer(n)

# Optionen

The screenshot shows the RStudio interface with several R Markdown files listed in the sidebar: Readme.Rmd\*, R2word.Rmd, AddOn.Rmd\*, Kolb\_PubKonzept.Rmd, and Re... . The main area displays the R2word.Rmd file's code:

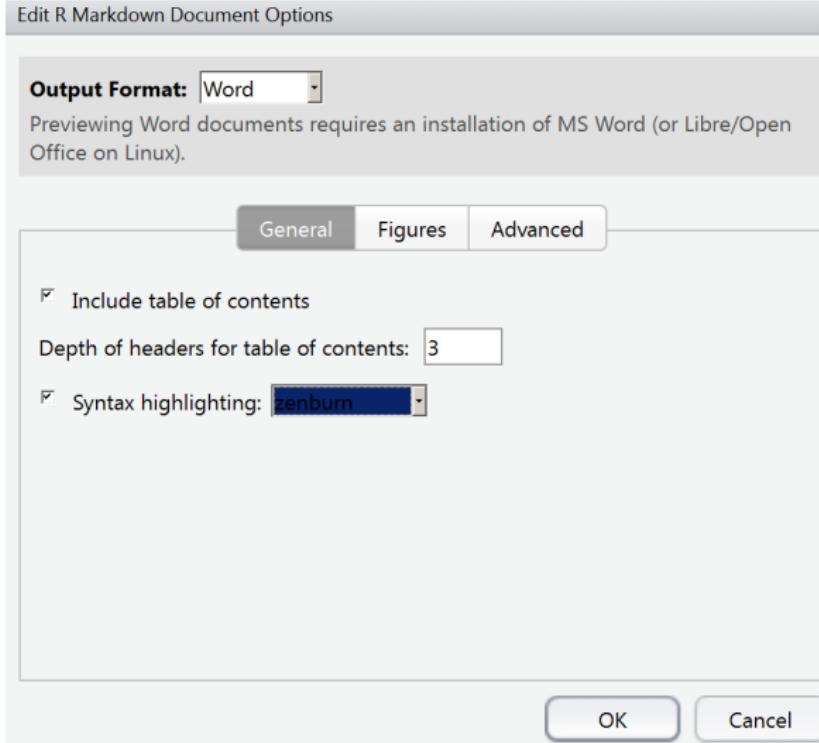
```
1 ---  
2 title: "R2word"  
3 author: "Jan-Philipp Kolb"  
4 date: "11 April 2017"  
5 output:  
6   html_document:  
7     keep_md: yes  
8 ---  
9  
10 ````{r setup, include=FALSE}  
11 knitr::opts_chunk$set(echo = TRUE)  
12 ````  
13  
14 ## Der Start  
15  
16 
```

A context menu is open over the code editor, specifically over the line `knitr::opts\_chunk\$set(echo = TRUE)`. The menu items are:

- Preview in Window
- Preview in Viewer Pane
- (No Preview)
- Preview Images and Equations
- Show Previews Inline

At the bottom of the menu, there is an "Output Options..." button and a tooltip that says "Edit the R Markdown format options for the current file".

# Optionen für Word Output



# Code Hervorhebung

- pygments Hervorhebung

```
# Beispiel für Code
ab <- sample(1:10,5,replace=T)
summary(ab)
```

##	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
##	2.0	3.0	5.0	5.4	8.0	9.0

- tango

```
# Beispiel für Code
ab <- sample(1:10,5,replace=T)
summary(ab)
```

##	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
##	5.0	6.0	6.0	6.6	7.0	9.0

# Das Paket knitr

```
install.packages("knitr")
```

```
library("knitr")
```

- Das Paket knitr enthält zahlreiche wichtige Funktionen
- Beispiel: Befehl `kable` um Tabellen zu erzeugen

## Eine Tabelle mit `kable` erzeugen

```
a <- runif(10)  
b <- rnorm(10)  
ab <- cbind(a,b)  
kable(ab)
```

a	b
0.3637329	-0.9653774
0.4897318	1.0985992
0.2881170	-0.2019373
0.7549531	1.3957739
0.1267248	-0.8987117
0.9710641	-0.2482414
0.9870349	-0.6015906
0.0973259	0.0221077
0.9970510	-0.2603437
0.5756800	-0.3695140

# Vorlagen verwenden

- Formatvorlagen können verändert werden
- ① Ein Word Dokument mit Rmarkdown erstellen
  - ② Das Dokument in Word öffnen und Format verändern
  - ③ Vorlage als Referenz angeben

```
1 ---  
2 title: "R2word"  
3 author: "Jan-Philipp Kolb"  
4 date: "11 April 2017"  
5 output:  
6   word_document:  
7     reference_docx: RefDoc.docx  
8     highlight: zenburn  
9     toc: yes
```

# Immer das aktuelle Datum im Kopf

```
date: "29 Juli, 2017"
```

```
1 ---  
2 title: "RPostgreSQL"  
3 author: "Jan-Philipp Kolb"  
4 date: `r format(sys.time(), '%d %B, %Y')`"  
5 output:  
6   slidify_presentation:  
7     keep_md: yes  
8 ---
```

# RPostgreSQL

# Ein Schummelzettel

## R Markdown

Schummelzettel

Mehr auf [rmarkdown.rstudio.com](http://rmarkdown.rstudio.com)

rmarkdown 0.2.60 Update: 8/14

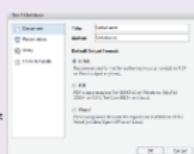


**1. Arbeitsablauf** Die Auszeichnungssprache „R Markdown“ erlaubt die Erstellung von reproduzierbaren und dynamisch anpassbaren Protokollen in R-R-Quellcode und dessen Ergebnisse können in Präsentationen, PDF-Dateien, HTML-Seiten, Word-Dateien etc. eingebettet werden. Um solch ein Protokoll zu erstellen:



**2. Datei erstellen** Zuerst wird eine Textdatei mit der Extension „.Rmd“ erstellt oder ein RStudio Rmd-Template geöffnet.

- Zum Erstellen eines „R Markdown“ Protokolls im Menü folgendes auswählen:
  - Dessel... > Neue Datei > R Markdown...
  - Oder im englischen Menü:
    - File > New File > R Markdown...
- Im neuen Fenster den Typ des Dokuments auswählen, das erstellt werden soll. Das entsprechende Optionfeld für die Dateiauswahl kann nachfolgend gekündigt werden.
- OK klicken.



**3. Markdown schreiben** Anschließend wird das Protokoll in Klarertext geschrieben. Zur Formatierung wird Markdown-Syntax verwendet.

### Syntax

KlarTEXT  
Alle die 8 Zeichenketten beginnen mit einem Zeichen zu beginnen.  
# Überschrift 1  
## Überschrift 2  
### Überschrift 3  
#### Überschrift 4  
##### Überschrift 5  
##### Überschrift 6

Abstand (empty, entweder): --  
Zeichenketten (empty, entweder): ---  
Akkusativspur (empty, entweder): ...  
einzelne Zeichen: a c r { } [ ] { }

weiterer Strich:

---

3. Stilzeichen

• ungestrichene Liste

• gestrichene Liste

1. gestrichene Liste

2. gestrichene Liste

3. gestrichene Liste

4. gestrichene Liste

5. gestrichene Liste

6. gestrichene Liste

7. gestrichene Liste

8. gestrichene Liste

9. gestrichene Liste

10. gestrichene Liste

### Resultat

KlarTEXT  
Ziel mit 2 Zeichenketten um neuen Absatz zu beginnen.  
# Überschrift 1  
## Überschrift 2  
### Überschrift 3  
#### Überschrift 4  
##### Überschrift 5  
##### Überschrift 6

## Überschrift 1

## Überschrift 2

### Überschrift 3

#### Überschrift 4

#### Überschrift 5

Abstand (empty, entweder): --  
Zeichenketten (empty, entweder): ---  
Akkusativspur (empty, entweder): ...  
einzelne Zeichen: a c r { } [ ] { }

weiterer Strich:

---

### Blocktext

#### ungestrichene Liste

#### gestrichene Liste

#### 1. gestrichene Liste

#### 2. gestrichene Liste

#### 3. gestrichene Liste

#### 4. gestrichene Liste

#### 5. gestrichene Liste

#### 6. gestrichene Liste

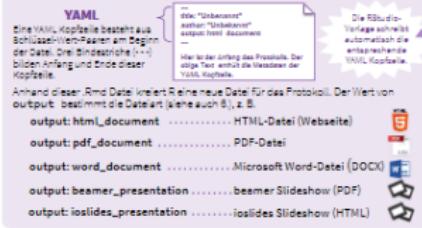
#### 7. gestrichene Liste

#### 8. gestrichene Liste

#### 9. gestrichene Liste

#### 10. gestrichene Liste

**4. Ausgabeformat auswählen** Eine YAML Kopfzeile wird erstellt. Sie verdeutlicht, welchen Dokumenttyp die R Markdown Datei erzeugen soll.



RStudio-Daten-Imports/Verknüpfungen/RStudio.Rproj - 00\_RStudio - http://rstudio.com - 244-442-1212 - 1048x608

- Interview - Ein Word Dokument mit wenig Aufwand schreiben
- pander: Ein R Pandoc Wrapper
- Einführung in Markdown
- Warum TeX besser als Word ist

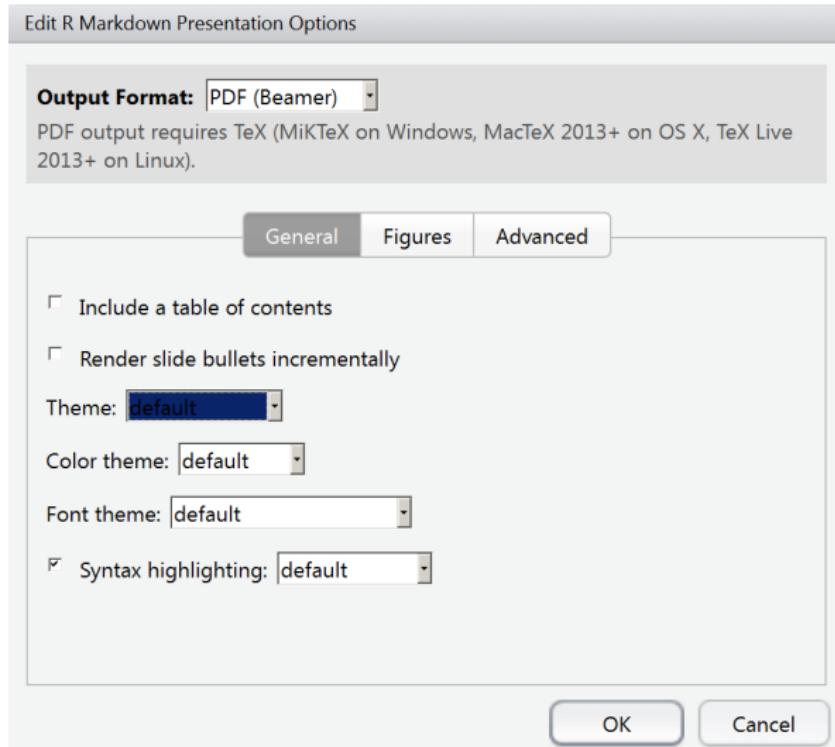
# PDF Dokumente und Präsentationen mit LaTeX, Beamer und Sweave

# Präsentationen mit Rmarkdown - beamer Präsentationen

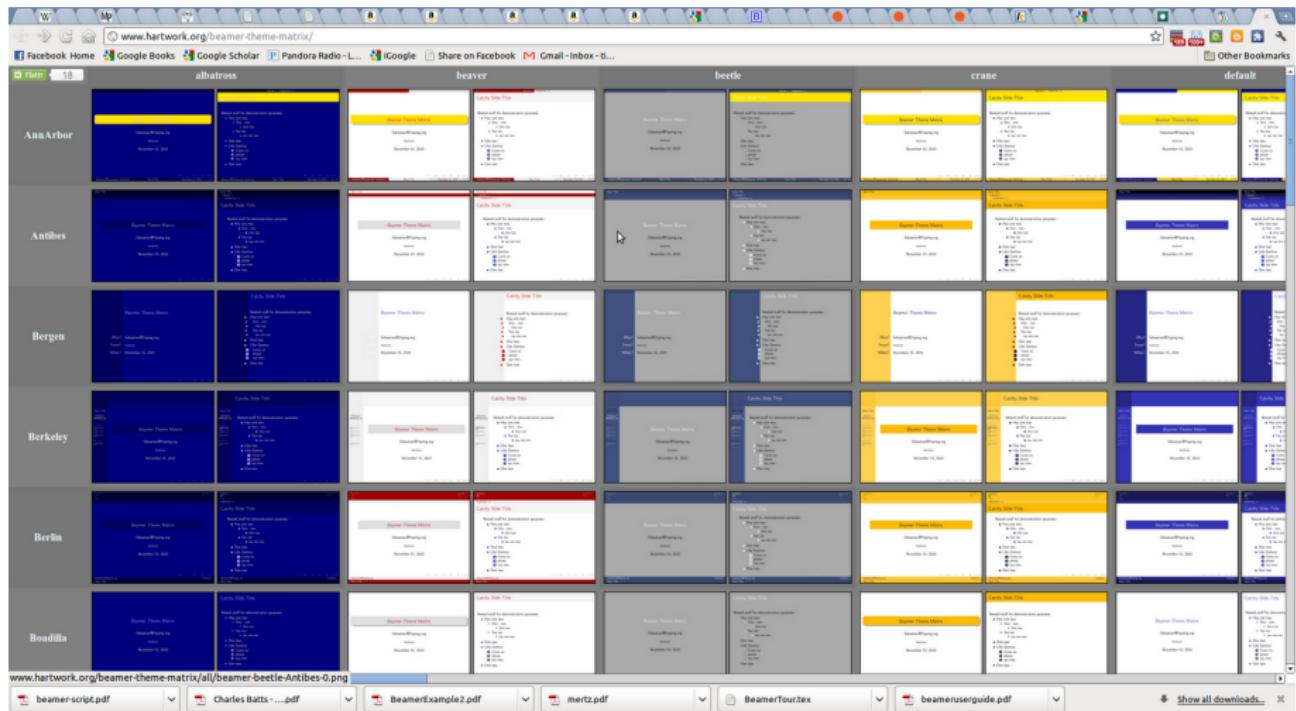
## Import csv

```
url <- "https://raw.githubusercontent.com/Japhilko/  
GeoData/master/2015/data/whcSites.csv"  
  
whcSites <- read.csv(url)
```

# Beamer Optionen



# Beamer Themen



[www.hartwork.org/beamer-theme-matrix/all/beamer-beetle-Antibes-0.png](http://www.hartwork.org/beamer-theme-matrix/all/beamer-beetle-Antibes-0.png)

beamerscript.pdf

Charles Batts -...pdf

BeamerExample2.pdf

mertz.pdf

BeamerTourtex

beamernuser/guide.pdf

Show all downloads...

# Chunks einfügen

- Auch hier lassen sich natürlich Chunks einfügen
- Wenn `cache=T` angegeben ist, wird das Ergebnis des Chunks abgespeichert
- Es ist sinnvoll die Chunks zu benennen, dann findet man auch das Ergebnis einfacher

```
```{r Zufallszahlen, cache=TRUE}
ab <- runif(1000)
````
```

# Ergebnis - Cache

| C:\Users\kolbjp\Documents\GitHub\RInterfaces\misc\RwordEx_cache\docx   |  |                  |             |       |
|--|--|------------------|-------------|-------|
| Computer \ System_Win7 (C:) \ Users \ kolbjp \ Eigene Dokumente \ GitHub \ RInterfaces \ misc \ RwordEx_cache \ docx |  |                  |             |       |
| Datei Bearbeiten Ansicht Extras ?  |  |                  |             |       |
| Organisieren In Bibliothek aufnehmen Freigeben für Brennen Neuer Ordner  |  |                  |             |       |
|  | Name   | Änderungsdatum   | Typ         | Größe |
| geosmdata2   |  |                  |             |       |
| geosmdata2.Rcheck  |  |                  |             |       |
| presentations  |  |                  |             |       |
| pythonscripts  |  |                  |             |       |
| rcode  |  |                  |             |       |
| slides   |  |                  |             |       |
| workshops  |  |                  |             |       |
|  | __packages   | 21.04.2017 15:55 | Datei       | 1 KB  |
|  | Zufallszahlen_ac0dfc8e07aa7d0c4bb3cfb18c491af8.RData | 21.04.2017 15:55 | RDATA-Datei | 3 KB  |
|  | Zufallszahlen_ac0dfc8e07aa7d0c4bb3cfb18c491af8.rdb   | 21.04.2017 15:55 | RDB-Datei   | 6 KB  |
|  | Zufallszahlen_ac0dfc8e07aa7d0c4bb3cfb18c491af8.rdx   | 21.04.2017 15:55 | RDX-Datei   | 1 KB  |

# Wie man das im Header des Dokuments angibt

---

```
title: "Intro - Erste Schritte"  
author: "Jan-Philipp Kolb"  
date: "10 April 2017"  
output:  
  beamer_presentation:  
    colortheme: beaver  
    theme: CambridgeUS
```

---

# Inhaltsverzeichnis I

---

```
output:  
  beamer_presentation:  
    toc: true
```

---

```
output:  
  beamer_presentation:  
    toc: yes
```

# Optionen für die Graphikeinbindung

- *fig\_caption: false*, wenn man keine Bildunterschriften haben möchte

---

```
title: "Habits"
```

```
output:
```

```
  beamer_presentation:
```

```
    fig_width: 7
```

```
    fig_height: 6
```

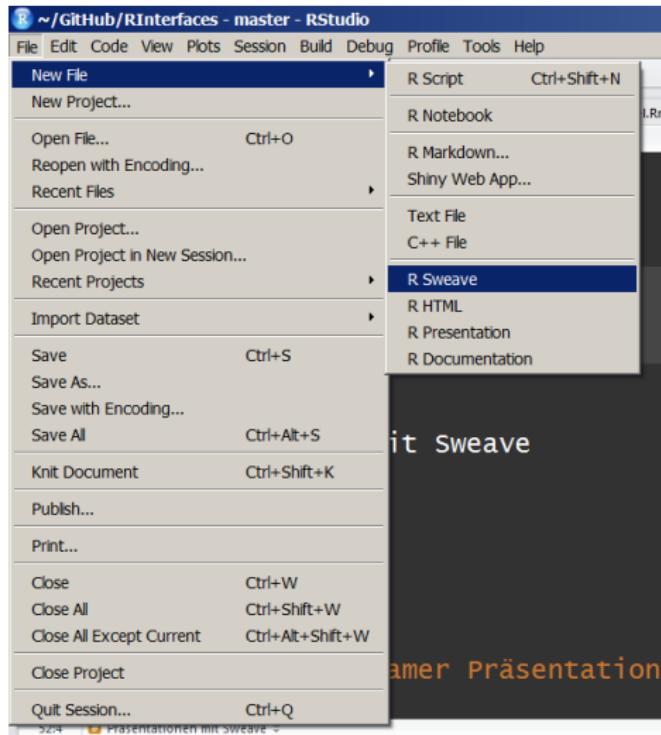
```
    fig_caption: true
```

---

- Man sollte keine Bilder im Format .svg einbinden

# Präsentationen mit Sweave

## • Das Dokument erstellen



# Sweave Präsentation

- Ganz normaler LaTeX Code wird verwendet

```
1 \documentclass{beamer}
2
3 \begin{document}
4
5 \begin{frame}
6 \frametitle{Erste Folie}
7 \end{frame}
8
9
10 \end{document}
```

## Chunks bei Sweave

- Auch hier kann R-code verwendet werden

```
9 \begin{frame}
10 \frametitle{Zweite Folie}
11
12 <<>>=
13 ab <- sample(1:10,8)
14 @
15
16 \end{frame}
```

# Chunk Optionen

- Auch bei Sweave Chunks können Optionen mitgegeben werden

|              |  |
|--------------|--|
| echo=false   | R-Code ist nicht sichtbar                    |
| results=tex  | die Ergebnisse werden wie TeX-Code behandelt |
| results=hide | die Ergebnisse sind nicht sichtbar           |
| fig=true     | Grafiken werden eingebunden                  |
| width=       | Breite der Grafik in Inch (z. B. 4)          |
| height=      | Höhe der Grafik in Inch                      |

# Inline Code

- Manchmal braucht man das Ergebnis direkt auf der Folie

```
\Sexpr{}
```

```
5 \begin{frame}
6 <<>>=
7 CRANmirror <- "http://cran.revolutionanalytics.com"
8
9 cran <- contrib.url(repos = CRANmirror, type = "source")
10
11 info <- available.packages(contriburl = cran, type = x)
12 @
13
14 Es gibt aktuell \Sexpr{nrow(info)} Pakete auf CRAN.
15 \end{frame}
```

## Inline Code - das Ergebnis

```
CRANmirror <- "http://cran.revolutionanalytics.com"
cran <- contrib.url(repos = CRANmirror, type = "source")

info <- available.packages(contriburl = cran, type = x)
```

Es gibt aktuell 10423 Pakete auf CRAN.

```
CRANmirror <- "http://cran.revolutionanalytics.com"
cran <- contrib.url(repos = CRANmirror, type = "source")
info <- available.packages(contriburl = cran, type = x)
nrow(info)

## [1] 11007
```

# PDF Paper mit R

- Mit R ist es möglich Berichte oder Paper zu erzeugen
- Dies eignet sich besonders gut, wenn man viel Code hat oder einen Bericht sehr oft erzeugen muss
- Literatur lässt sich am Besten mit einem bibtex file einbauen

# Jabref

## • Literaturverwaltungssystem

The screenshot shows the JabRef application window. The title bar reads "JabRef - C:\Users\kolbjp\Documents\GitHub\RInterfaces\paper\Rschnittstellen.bib\* (BIBTeX mode)". The menu bar includes File, Edit, Search, Groups, View, BibTeX, Quality, Tools, Options, Help. Below the menu is a toolbar with various icons for file operations like Open, Save, Print, and search. A search bar with placeholder "Search..." is followed by a "Filter" button. The main area displays a table of references:

| # | entrytype | author/editor           | title   | year | journal/booktitle | bibtexkey   | ranking |
|---|-----------|-------------------------|---|------|-------------------|-------------|---------|
| 1 | Manual    | Ooms et al.             | RMySQL: Database Interface and MySQL Driver for R                                 | 2017 |                   | ma.ooms     |         |
| 2 | Manual    | R Core Team             | foreign: Read Data Stored by Minib, S, SAS, SPSS, Stata, Systat, Weka, dBase, ... | 2016 |                   | ma.coreteam |         |
| 3 | Manual    | de Jonge and Tennekes   | tabplot3: Tabplot3, interactive inspection of large data                          | 2013 |                   | ma.jonge    |         |
| 4 | Book      | Wickham                 | ggplot2: Elegant Graphics for Data Analysis                                       | 2009 |                   | ma.wickham  |         |
| 5 | Manual    | Chang and Wickham       | grid: Interactive Grammar of Graphics   | 2016 |                   | ma.chang    |         |
| 6 | Manual    | Wickham and François    | dplyr: A Grammar of Data Manipulation   | 2016 |                   | ma.François |         |
| 7 | Article   | Gesmann and de Castillo | googleVis: Interface between R and the Google Visualisation API                   | 2011 | The R Journal     | at.gesman   |         |
| 8 | Manual    | Wickham and Chang       | devtools: Tools to Make Developing R Packages Easier                              | 2016 |                   | ma.wickham  |         |

Below the table, there is a form with fields for "Title" (d3Network: Tools for creating D3 JavaScript network, tree, dendrogram, and Sankey graphs from R) and "Bibtexkey" (ma.gandrud). A status message at the bottom says "Status: BibTeX key is unique."

# Referenz mit R bekommen

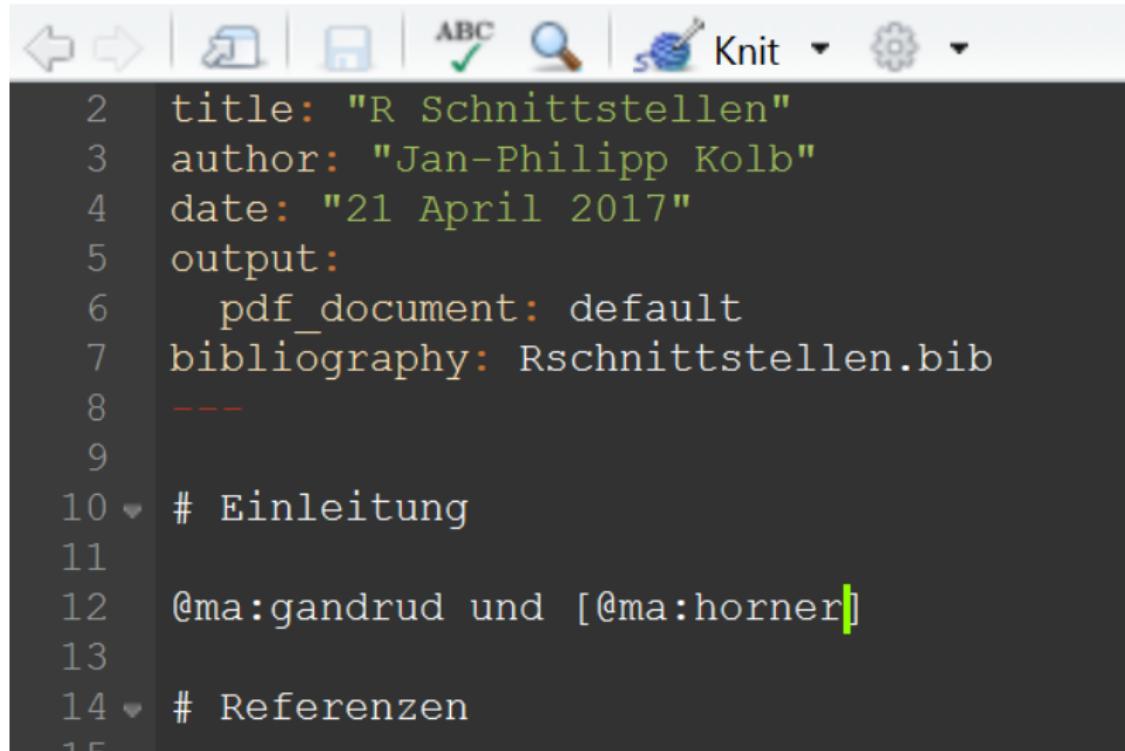
- Mit dem Befehl `citation()` bekommt man sehr schnell die Referenz

```
install.packages("RMySQL")
```

```
citation("RMySQL")
```

```
##  
## To cite package 'RMySQL' in publications use:  
##  
##   Jeroen Ooms, David James, Saikat DebRoy, Hadley Wickham a  
##   Jeffrey Horner (2017). RMySQL: Database Interface and 'My  
##   Driver for R. R package version 0.10.11.  
##   https://CRAN.R-project.org/package=RMySQL  
##  
## A BibTeX entry for LaTeX users is  
##  
##   @Manual{,
```

# Das bibtex file einbinden I



The screenshot shows a portion of the RStudio interface. At the top is a toolbar with icons for back, forward, file, ABC, search, and knit. Below the toolbar is a code editor window displaying the following R code:

```
2 title: "R Schnittstellen"
3 author: "Jan-Philipp Kolb"
4 date: "21 April 2017"
5 output:
6   pdf_document: default
7 bibliography: Rschnittstellen.bib
8 ---
9
10 # Einleitung
11
12 @ma:gandrud und [ @ma:horner]
13
14 # Referenzen
15
```

The code includes several comments and a bibliography entry. The line starting with '@ma:gandrud' is highlighted with a green background and a yellow cursor is positioned at the end of the line.

# Das bibtex file einbinden II

---

```
title: "R Schnittstellen"  
author: "Jan-Philipp Kolb"  
date: "21 April 2017"  
output:  
  pdf_document: default  
bibliography: Rschnittstellen.bib
```

---

# Das Ergebnis

## R Schnittstellen

*Jan-Philipp Kolb*

*21 April 2017*

### Einleitung

Gandrud (2015) und (Horner 2014)

### Referenzen

Gandrud, Christopher. 2015. *D3Network: Tools for Creating D3 Javascript Network, Tree, Dendrogram, and Sankey Graphs from R.* <https://CRAN.R-project.org/package=d3Network>.

Horner, Jeffrey. 2014. *Rook: Rook - a Web Server Interface for R.* <https://CRAN.R-project.org/package=Rook>.

# Links

- Optionen für Beamer Präsentationen
- Wie R und LaTeX zusammen funktionieren

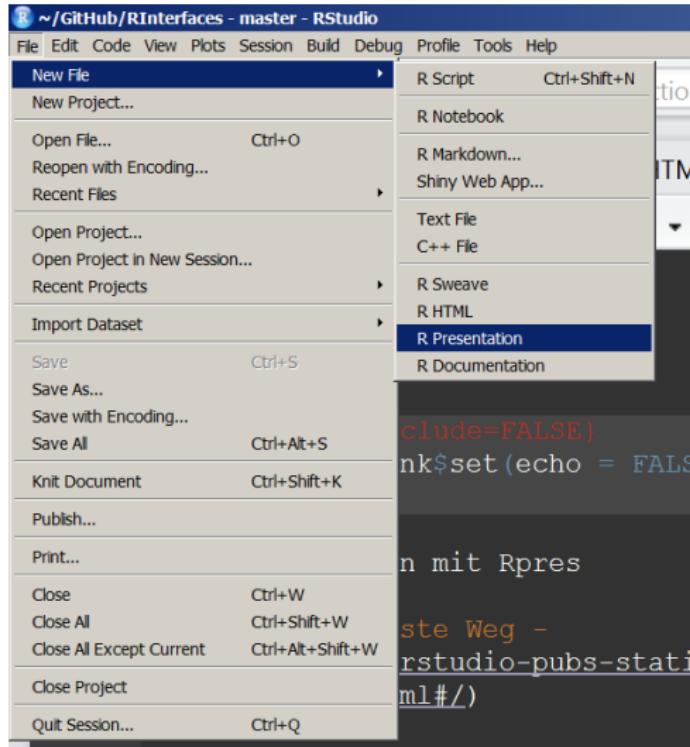
# HTML Dokumente, Präsentationen und Dashboards mit Rmarkdown

# Präsentationen - Rpres der einfachste Weg

The screenshot shows a web browser window with the following details:

- Address Bar:** File:///D:/Temp/RtmpW21Px3/view16e074596ca2.dir/presentation.html/
- Toolbar:** Standard browser icons for back, forward, search, and others.
- Search Bar:** Suchen (Search) with a magnifying glass icon.
- Content Area:** A large white area containing the presentation content.
- Content:**
  - # R presentations
  - Jan-Philipp Kolb  
11.04.2017

# Eine erste Präsentation



# Erste Daten eintragen

- Für Vergessliche:

```
date()
```

```
## [1] "Sun Jul 23 12:18:59 2017"
```

# Eine Folie mit Formel

- Die Formel kann wie in LaTeX eingegeben werden

\$\$

```
\begin{equation}\label{eq2}
```

```
t_{i}=\sum\limits_{k=1}^{M_i}{y_{ik}}=M_i\bar{Y}_i.
```

```
\end{equation}
```

\$\$

The screenshot shows the RStudio interface. On the left, the code editor displays the following LaTeX code:

```
29
30 Folie mit LaTeX Code
31
32
33 $$%
34 \begin{equation}\label{eq2}
35 t_{(i)} = \sum\limits_{k=1}^{M_{(i)}} (y_{(ik)}) = M_{(i)} \bar{Y}_{(i)} .
36 \end{equation}
37 $$%
38
```

The status bar at the bottom indicates "Folie mit LaTeX Code 44:1". On the right, the preview pane shows the rendered LaTeX output:

Folie mit LaTeX Code

$$t_i = \sum_{k=1}^{M_i} y_{ik} = M_i \bar{Y}_i$$

# Zwei Spalten

Folie mit zwei Spalten

---

Erste Spalte

\*\*\*

Zweite Spalte

# Folienübergänge

```
transition: rotate
```

```
1 Meine Erste Präsentation mit Markdown
2 ▾ =====
3 author: Jan-Philipp Kolb
4 date: Thu Apr 20 09:06:19 2017
5 autosize: true
6 transition: rotate|
```

# Weitere mögliche Folienübergänge

- none
- linear
- rotate
- fade
- zoom
- concave

# Folientypen

Ein neues Kapitel einfügen

```
=====
```

type: section

Anderer Folientyp

```
=====
```

type: prompt

Noch ein anderer Folientyp

```
=====
```

type: alert

# Die Schriftart wechseln

- Die CSS Schrifttypen können verwendet werden

Meine Präsentation

---

author: Jan-Philipp Kolb  
font-family: 'Impact'

# Schrifttypen können auch importiert werden

Meine Präsentation

---

author: Jan-Philipp Kolb

font-import: http://fonts.googleapis.com/css?family=Risque

font-family: 'Risque'

The screenshot shows a presentation slide with the following details:

- Title Bar:** Shows a house icon, the title "Meine Erste Präsentation mit Markdown...", and three icons for edit, search, and settings.
- Content Area:** Displays the main title "Meine Erste Präsentation mit Markdown" in a large, serif font.
- Author Information:** At the bottom left, it says "Jan-Philipp Kolb".
- Page Number:** At the bottom right, it says "381 / 463".

## Kleineren Text

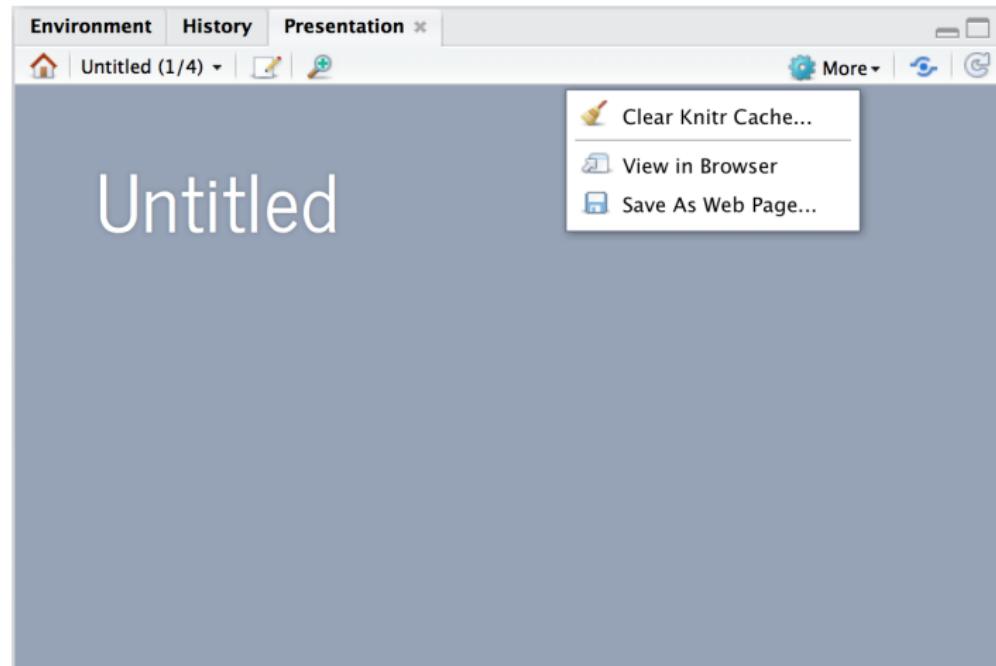
Normale Schriftgröße

<small>This sentence will appear smaller.</small>

# Die Präsentation anschauen

- Das Ergebnis ist hier zu sehen:

<http://rpubs.com/Japhilko82/FirstRpubs>



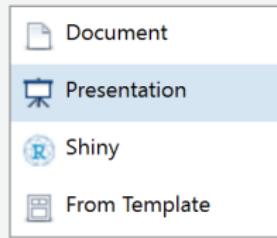
# Eine ioslides Präsentation

# Eine ioslides Präsentation

The screenshot shows a presentation slide in a web browser window. The address bar at the top displays the URL: file:///C:/Users/kolbjp/Documents/GitHub/RInterfaces/slides/R2pdf.html#1. The slide itself has a black background with a white central content area. In the center, there is large, bold, dark gray text that reads "Präsentationen mit R und Rstudio". Below this title, in a smaller font, is the author's name "Jan-Philipp Kolb" and the date "11 April 2017". The overall layout is clean and minimalist.

# ioslides - Der Start

New R Markdown



**Title:** ioslides Beispiel

**Author:** Jan-Philipp Kolb

**Default Output Format:** HTML (ioslides)

HTML presentation viewable with any browser (you can also print ioslides to PDF with Chrome).

 HTML (Slidy)

HTML presentation viewable with any browser (you can also print Slidy to PDF with Chrome).

 PDF (Beamer)

PDF output requires TeX (MiKTeX on Windows, MacTeX 2013+ on OS X, TeX Live 2013+ on Linux).

OK

Cancel

# Weitere Dinge tun

- Ein Bild einbinden

! [picture of spaghetti] (images/spaghetti.jpg)

# Ein Logo hinzufügen

---

```
title: "ioslides Beispiel"  
author: "Jan-Philipp Kolb"  
date: "20 April 2017"  
output:  
  ioslides_presentation:  
    logo: figure/Rlogo.png
```

---

```
1 ---  
2 title: "ioslides Beispiel"  
3 author: "Jan-Philipp Kolb"  
4 date: "20 April 2017"  
5 output:  
6   ioslides presentation:
```

# Tabellen

- Quelle: R Studio, and Presentations, and Git! Oh my!

```
library(knitr)
a <- data.frame(a=1:10,b=10:1)
kable(table(a))
```

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|----|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1  |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0  |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0  |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0  |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0  |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0  |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0  |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0  |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0  |

# knitr Engines

- knitr Language Engines
- slidify

# Eine slidy Präsentation

# slidy Präsentationen



## Präsentationen mit R und Rstudio

Jan-Philipp Kolb

11 April 2017

# [Was sind Cascading Style Files (CSS)]



- Stylesheet-Sprache für elektronische Dokumente
- eine der Kernsprachen des World Wide Webs.
- CSS wurde entworfen, um Darstellungsvorgaben weitgehend von den Inhalten zu trennen

## CSS und R

- Custom CSS
- CSS pro tipps

# Beispiel CSS

```
1 body, td {  
2   font-family: Lucida Console;  
3   background-color: transparent;  
4   font-size: 20px;  
5 }  
6
```

# Das CSS ändern

Um den Präsentationstyp zu ändern kann man das CSS verändern

- Cascading Style Sheets (CSS)
- Bspw. lässt sich die Farbe (HTML) ändern.
- Man kann eine andere Schriftart wählen
- Es gibt zahlreiche Möglichkeiten der Schriftformatierung
- Daneben gibt es viele weitere Dinge, die sich mit dem CSS steuern lassen

# HTML Dokumente

# Ein HTML Dokument erzeugen

New R Markdown

Document

Presentation

Shiny

From Template

**Title:** Neues HTML Dokument

**Author:** Jan-Philipp Kolb

**Default Output Format:**

HTML  
Recommended format for authoring (you can switch to PDF or Word output anytime).

PDF  
PDF output requires TeX (MiKTeX on Windows, MacTeX 2013+ on OS X, TeX Live 2013+ on Linux).

Word  
Previewing Word documents requires an installation of MS Word (or Libre/Open Office on Linux).

# Ein Template verwenden

New R Markdown

**Template:** [Using R Markdown Templates](#)

|  |                 |
|--|-----------------|
| R Journal Submission                         | {rticles}       |
| Flex Dashboard                               | {flexdashboard} |
| Github Document (Markdown)                   | {rmarkdown}     |
| Package Vignette (HTML)                      | {rmarkdown}     |
| HTML clean template                          | {rmdformats}    |
| HTML clean template (ProjectTemplate report) | {rmdformats}    |
| HTML docco template                          | {rmdformats}    |

This template contains multiple files. Create a new directory for these files:

**Name:**

**Location:**

# Weitere Vorlagen nutzen

- Es gibt viele Formate - manche müssen erst aktiviert werden:

```
install.packages("rticles")
```

## Short Paper

Alice Anonymous  
Some Institute of Technology  
[alice@example.com](mailto:alice@example.com)

Bob Security  
Another University  
[bob@example.com](mailto:bob@example.com)

### ABSTRACT

This is the abstract.

It consists of two paragraphs.

### 1. INTRODUCTION

ut diam. Nulla ut dapibus quam.

Sed est odio, ornare in rutrum et, dapibus in urna. Suspendisse varius massa in ipsum placerat, quis tristique magna consequat. Suspendisse non convallis augue. Quisque fermentum justo et lorem volutpat euismod. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cu-

# Vorlagen für Markdown

Das Paket `rmdformats` - HTML Output Formats and Templates for 'rmarkdown'

```
install.packages("rmdformats")
```

- `ProjectTemplate` - Automates the Creation of New Statistical Analysis

```
install.packages("ProjectTemplate")
```

- `tufte` - Tufte's Styles for R Markdown Documents

```
install.packages("tufte")
```

# Beispiele für Templates

**readthedown template example**

Code and tables

Figures

MathJax

## Code and tables

### Syntax highlighting

Here is a sample code chunk, just to show that syntax highlighting works as expected.

```
library(ggplot2)
library(dplyr)

not_null <- function (v) {
  if (!is.null(v)) return(casteCv, "not null"))
}

data(iris)
tab <- iris %>%
  group_by(Species) %>%
  summarise(Sepal.Length = mean(Sepal.Length),
            Sepal.Width = mean(Sepal.Width),
            Petal.Length = mean(Petal.Length),
            Petal.Width = mean(Petal.Width))
```

### Verbatim

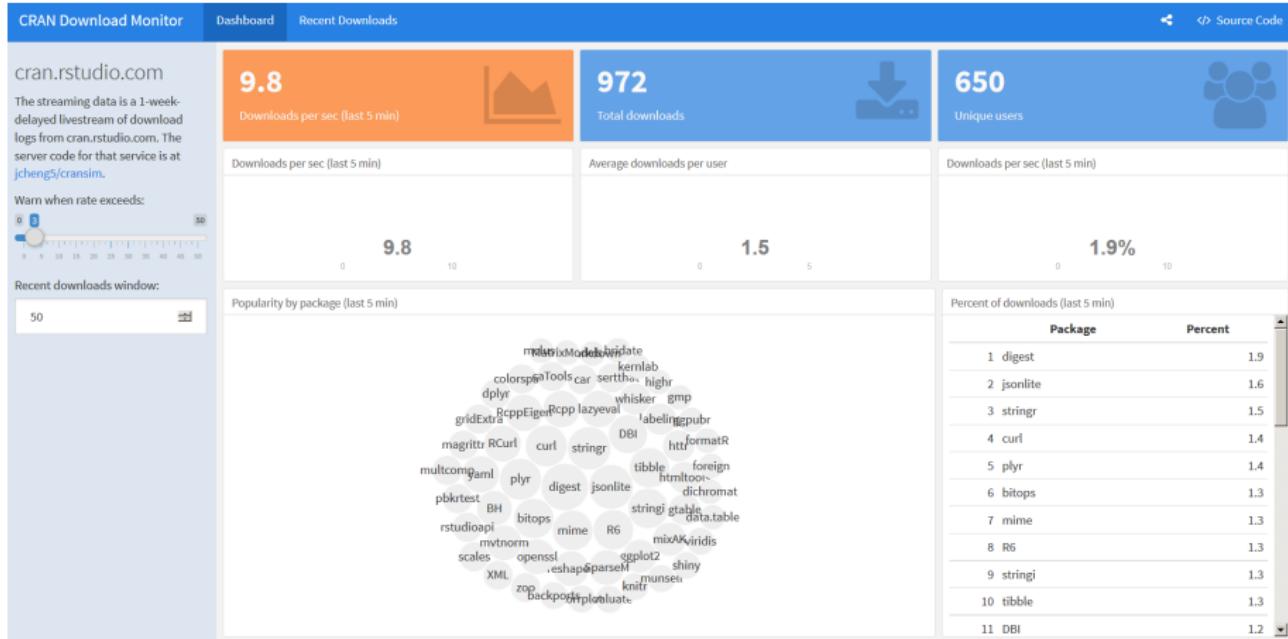
Here is the structure of the `iris` dataset.

```
str(iris)
```

```
'data.frame':  150 obs. of  5 variables:
 $ Sepal.Length: num  5.1 4.9 4.7 4.6 5.5 4.6 5.4 4.4 4.9 ...
 $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
 $ Petal.Length: num  1.7 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
 $ Petal.Width : num  0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
 $ Species     : Factor w/ 3 levels "setosa","versicolor",... 1 2 3 1 1 1 1 1 1 1 1 ...
```

# Dashboards

# Beispiel R-Pakete

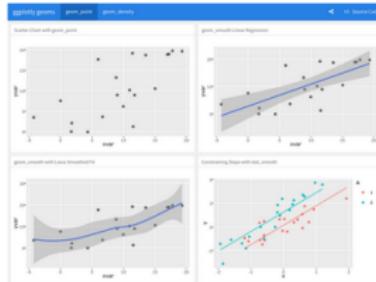


# Paket installieren

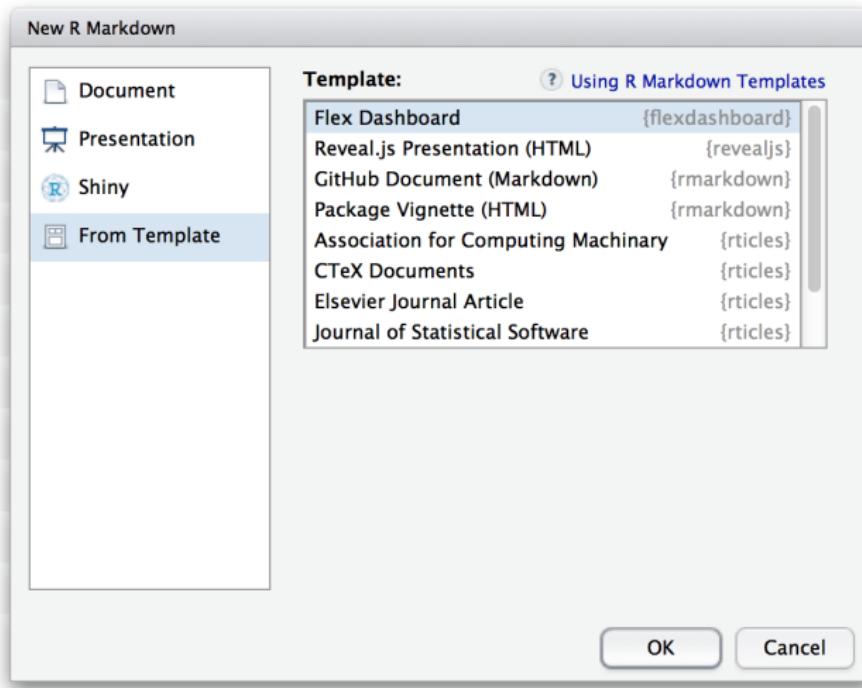
```
install.packages("flexdashboard", type = "source")
```

## flexdashboard: Easy interactive dashboards for R

- Use [R Markdown](#) to publish a group of related data visualizations as a dashboard.
- Support for a wide variety of components including [htmlwidgets](#); base, lattice, and grid graphics; tabular data; gauges and value boxes; and text annotations.
- Flexible and easy to specify row and column-based [layouts](#). Components are intelligently re-sized to fill the browser and adapted for display on mobile devices.
- [Storyboard](#) layouts for presenting sequences of visualizations and related commentary.
- Optionally use [Shiny](#) to drive visualizations dynamically.



# Ein Dashboard erstellen mit Rstudio



# Mein erstes Dashboard

**R**Pubs brought to you by RStudio

## Mein erstes Dashboard

Eine interaktive Datentabelle mit [DT](#)

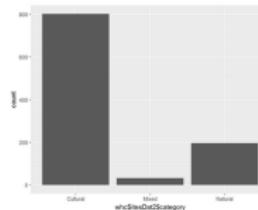
Show **10** entries

|   | name_en  | category | longitude   | latitude    | date_inscribed | area_hectares | danger_list |
|---|--|----------|-------------|-------------|----------------|---------------|-------------|
| 1 | Cultural Landscape and Archaeological Remains of the Bamyan Valley | Cultural | 67.82525    | 34.84694    | 2003           | 158.9265      | Y 2003      |
| 2 | Minaret and Archaeological Remains of Jam                          | Cultural | 64.51605556 | 34.39655556 | 2002           | 70            | Y 2002      |
| 3 | Historic Centres of Berat and Gjirokastra                          | Cultural | 20.13333333 | 40.06944444 | 2005           | 58.9          |             |
| 4 | Butrint   | Cultural | 20.02611111 | 39.75111111 | 1992           |               |             |
| 5 | Al Qal'a of Beni Hammad  | Cultural | 4.78684     | 35.81844    | 1980           | 150           |             |
| 6 | M'Zab Valley   | Cultural | 3.68333     | 32.48333    | 1982           | 665.03        |             |
| 7 | Dj  l  mila  | Cultural | 5.73667     | 36.32056    | 1982           | 30.6          |             |
| 8 | Timgad   | Cultural | 6.63333     | 35.45       | 1982           | 90.54         |             |
| 9 | Kasbah of Algiers  | Cultural | 3.06028     | 36.78333    | 1992           | 60            |             |

Showing 1 to 10 of 1,031 entries

Previous 1 2 3 4 5 ... 104 Next

Eine einfache [ggplot2](#) Graphik



Eine Karte mit [leaflet](#)



Leaflet | © OpenStreetMap contributors, CC-BY-SA

[Edit Details](#)

[Delete](#)

Mein erstes Dashboard

by Jan-Philipp Kolb

Last updated 1 minute ago

[Comments \(-\)](#)

[Share](#)

[Hide Toolbars](#)

# Gallerie

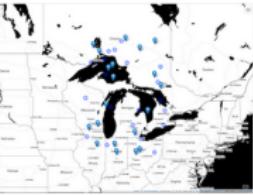
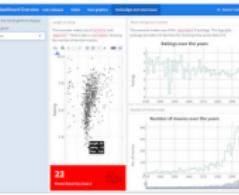
## Documents

With R Markdown, you write a single .Rmd file and then use it to render finished output in a variety of formats.

|   |   |   |  |
|---|---|---|--|
| <p><b>Great NYT Interactive -- Now Reusable with rCharts</b></p> <p><a href="#">Disclaimer and Attribution</a></p> <p>This document provides an introduction to R Markdown, argue its benefits, and present examples of how it can be used to produce documents. It also describes how to use R Markdown and provides a detailed working example of how the analysis can be conducted within a document.</p> <p><a href="#">Another Example from NYT</a></p> <p><a href="#">Great NY Times Interactive Article</a></p> <p><a href="#">Great NY Times Interactive Article</a></p> <p><b>HTML</b></p> <p>HTML documents for web publishing.</p> | <p><b>A Pandoc Markdown Article Starter and Template*</b></p> <p><a href="#">Issue 1 Note</a></p> <p>This document provides an introduction to R Markdown, argue its benefits, and present examples of how it can be used to produce documents. It also describes how to use R Markdown and provides a detailed working example of how the analysis can be conducted within a document.</p> <p><a href="#">Another Example from NYT</a></p> <p><a href="#">Great NY Times Interactive Article</a></p> <p><b>PDF</b></p> <p>PDF documents for printing. <a href="#">Example Code</a></p> | <p><b>A Microsoft Word document</b></p> <p><a href="#">About</a></p> <p><a href="#">Issue 1 Note</a></p> <p>This document provides an introduction to R Markdown, argue its benefits, and present examples of how it can be used to produce documents. It also describes how to use R Markdown and provides a detailed working example of how the analysis can be conducted within a document.</p> <p><a href="#">Another Example from NYT</a></p> <p><a href="#">Great NY Times Interactive Article</a></p> <p><b>Microsoft Word</b></p> <p>Microsoft Word documents for Office workflows.</p> | <p><b>Tufte Handout</b></p> <p><a href="#">An implementation in R Markdown</a></p> <p><a href="#">If All Else Fails</a></p> <p><a href="#">Handout</a></p> <p>This document provides an introduction to R Markdown, argue its benefits, and present examples of how it can be used to produce documents. It also describes how to use R Markdown and provides a detailed working example of how the analysis can be conducted within a document.</p> <p><a href="#">Another Example from NYT</a></p> <p><a href="#">Great NY Times Interactive Article</a></p> <p><b>Handouts</b></p> <p>Tufte styled documents for handouts. <a href="#">Example Code</a></p> |
|---|---|---|--|

## Interactive Documents

Combine R Markdown with htmlwidgets or the shiny package to make interactive documents.

|   |   |   |   |
|---|---|---|---|
| <p><b>HTML Widgets</b></p> <p>Add interactive graphics with htmlwidgets, such as the leaflet map widget.</p>  | <p><b>UNCC Data Report</b></p> <p><a href="#">1 Disclaimer</a></p> <p><a href="#">2 Data Overview</a></p> <p><b>HTML Widgets</b></p> <p>Embed htmlwidgets such as dygraphs and dataTables directly into your reports.</p> | <p><b>Shiny leaflet example</b></p> <p><a href="#">Version 0 - Use observe</a></p> <p><b>Shiny</b></p> <p>Add interactive analysis with shiny, which lets your user rerun the actual analysis within your report.</p> | <p><b>Shiny</b></p> <p>Shiny components and htmlwidgets will work in any HTML based output, such as a file, slide show or dashboard.</p>  |
|---|---|---|---|

# Links

- Verschiedene Markdown Dokumente zusammen fügen



55



I'm not sure this is exactly what you're looking for, but when I want to break a large report into separate Rmd, I usually create a parent Rmd and include the chapters as children. This approach is also easy for new users to understand. It doesn't create a nice title for each chapter, but as long as you include a toc, it is easy to navigate between chapters. One pitfall doing this is that all chunk names between all parent/children need to be unique.

report.Rmd

```
---
```

```
title: My Report
```

```
output:
```

```
  pdf_document:
```

```
    toc: yes
```

```
---
```

```
```{r child = 'chapter1.Rmd'}
```

```
```
```

```
```{r child = 'chapter2.Rmd'}
```

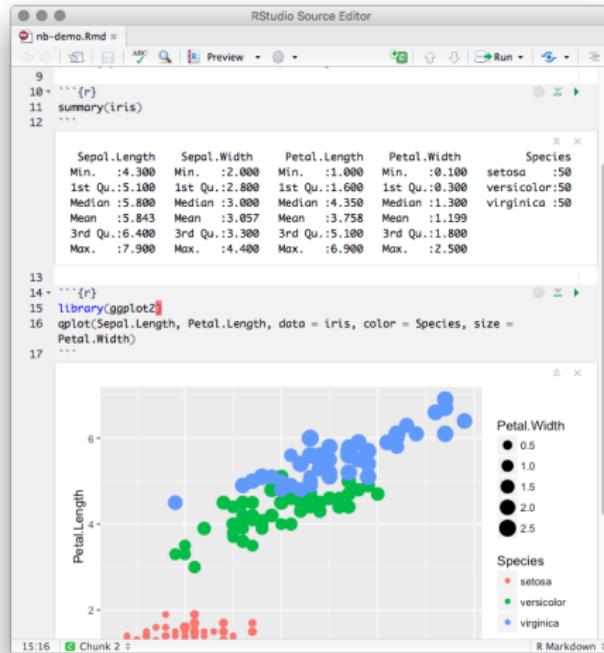
```
```
```

- Verschiedene CSS Fonts

# Notebooks zur Integration von anderen Programmiersprachen (Python,LaTeX,Julia)

# Notebooks

- Warum R Notebook nutzen





# Ein Rnotebook anlegen

The screenshot shows the RStudio interface. The top menu bar has "File" selected, and a dropdown menu is open showing options like "New File", "New Project...", "Open File...", "Save", and "Close". Below this, a sub-menu for "R Notebook" is open, listing "R Markdown...", "Shiny Web App...", "Text File", "C++ File", "R Sweave", "R HTML", "R Presentation", and "R Documentation". The main workspace shows some R code:

```

tation:
beaver
structurebold
tango
ridgeUS
: false
rclutter=FALSE)
ink$set(echo = TRUE)

```

The bottom panel is a "Console" window showing the command "R Markdown". To the right, the "Presentation" tab is active in the top navigation bar, and the slide content is displayed:

## R presentations

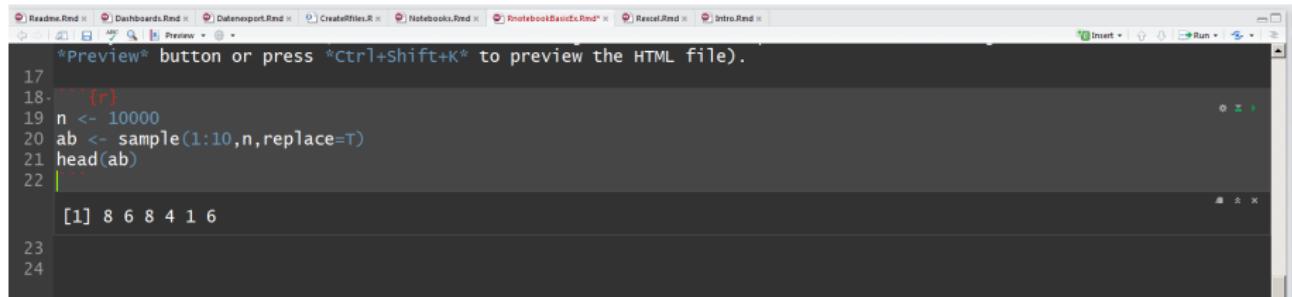
Jan-Philipp Kolb  
11.04.2017

The file browser on the right shows the contents of a folder:

| Name              | Size     |
|-------------------|----------|
| presentHTMLmd     | 1.1 KB   |
| presentHTMLRmd    | 857 B    |
| presentHTML_files |          |
| R2pdf.html        | 292.3 KB |
| R2pdf.pdf         | 775 KB   |
| R2pdf.Rmd         | 1.1 KB   |
| rcpp.html         | 692.7 KB |
| rcpp.md           | 372 B    |
| rcpp.Rmd          | 862 B    |
| Rexcel.html       | 692.5 KB |

# Rnotebook - erste Schritte

- Es lassen sich so genannte Chunks einfügen
- In diesen Chunks wird ganz normaler R-code geschrieben



The screenshot shows an R notebook window with multiple tabs at the top. The active tab is titled "RnotebookBasics.Rnd". The main area contains R code and its output:

```
*Preview* button or press *Ctrl+Shift+K* to preview the HTML file.
17
18  ````[r]
19 n <- 10000
20 ab <- sample(1:10,n,replace=T)
21 head(ab)
22
[1] 8 6 8 4 1 6
23
24
```

The code uses a chunk indicator ````[r]` to define a chunk where the R code is executed. The output of the `head(ab)` command is shown below the code.

# Python Code integrieren

- Ebenso lässt sich Python code implementieren



A screenshot of the Rnotebooks interface. On the left, there is a code editor window with the following Python code:

```
20 ````(python)
21 print("Hello World")
22 ````
```

The output window on the right shows the result of the code execution:

```
Hello World
```

The interface includes standard window controls (minimize, maximize, close) and a toolbar with icons for settings, file operations, and other functions.

```
import sys
print(sys.version)
```

```
## 2.7.10 (default, May 23 2015, 09:44:00) [MSC v.1500 64 bit]
```

# LaTeX Code integrieren

- LaTeX code wird mit zwei Dollarzeichen gekennzeichnet

```
$$\alpha = \frac{\beta}{\lambda}$$
```

$$\alpha = \frac{\beta}{\lambda}$$



# Notebook veröffentlichen I

The screenshot shows an R Notebook interface. At the top, there's a menu bar with tabs: Files, Packages, Help, and Viewer. Below the menu is a toolbar with icons for file operations like Open, Save, and Print, and for RStudio-specific functions like Reload and Check Syntax. The main content area has a large title "R Notebook" and a section titled "R code inline". Inside this section, the code "plot(cars)" is shown, along with its resulting plot. A "Code" dropdown menu is open above the code block, and a "Hide" button is located to the right of the plot. A vertical scroll bar is visible on the right side of the content area.

# R Notebook

## R code inline

```
plot(cars)
```

• Other language engines

# Notebook veröffentlichen II

Publish

## Publish To

 RPubs

### RPubs

RPubs is a free service from RStudio for sharing documents on the web.



### RStudio Connect

RStudio Connect is a server product from RStudio for secure sharing of applications, reports, and plots.



Cancel

# Andere Notebooks

# Jupyter Notebook

- Anaconda installieren
- folgenden Befehl in die Eingabeaufforderung eingeben
- Bei Windows findet man diese, wenn man cmd in Suche eingibt.

```
jupyter notebook
```

# Start Jupyter Notebook



Files    Running    Clusters

Select items to perform actions on them.

Upload    New ▾    ⌂

- Anaconda3
- AppData

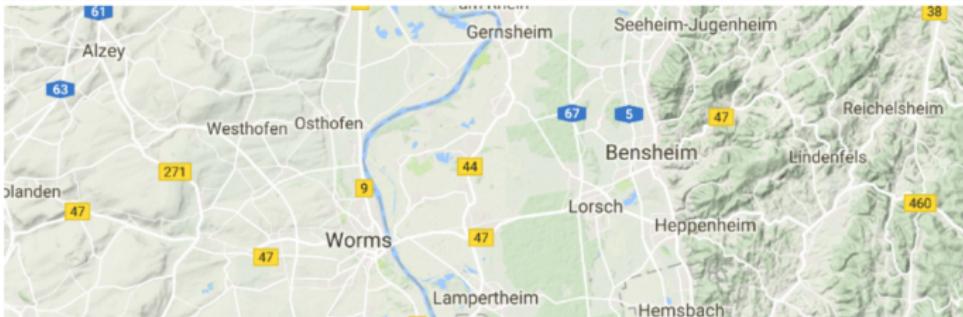
# Beispiel Eingabe Code

Jupyter An Jupyter R notebook Last Checkpoint: 04/27/2016 (unsaved changes)

File Edit View Insert Cell Kernel Help | R

In [2]: `library(ggmap)`  
Loading required package: ggplot2

In [3]: `qmap("Mannheim")`  
Map from URL : <http://maps.googleapis.com/maps/api/staticmap?center=Mannheim&zoom=10&size=640x640&scale=2&maptype=terrain&language=en-EN&sensor=false>  
Information from URL : <http://maps.googleapis.com/maps/api/geocode/json?address=Mannheim&sensor=false>



# Beaker Notebook

# Beaker Notebook

- Auch bei Beaker kann man R-code einbauen

The banner features a blue-toned background with a radial pattern of light rays emanating from the center. In the lower-left quadrant, there is a white icon of a flask with a dropper above it, followed by the word "BEAKER" in large, bold, white capital letters with a trademark symbol. Below this, the text "THE DATA SCIENTIST'S LABORATORY" is written in a smaller white font. To the right, there is a green rectangular button with the text "Get Beaker" in white. Further down, another green button contains the text "Mac | Win | Linux" in white. In the top right corner, the text "Follow Beaker" is displayed above several social media icons: Twitter, GitHub, Facebook, and Google+.

Follow Beaker

Get Beaker

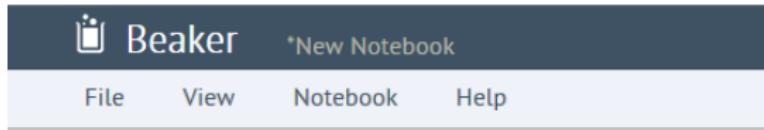
Mac | Win | Linux

THE DATA SCIENTIST'S LABORATORY

Beaker is a notebook-style development environment for working interactively with large and complex datasets. Its plugin-based architecture allows you to switch between languages or add new ones with ease, ensuring that you always have the right tool for any of your analysis and visualization needs.

# Beaker starten

- Beaker installieren ...
- ... und mit beaker.command.bat starten



The screenshot shows an R notebook cell. At the top left are two buttons: a blue one labeled "R" and a green one labeled "R ▾". Below them is the R code: 

```
1 | sample(1:10, 5, replace=T)
```

. To the right of the code is a "Run" button. The output of the code is shown below, preceded by a horizontal line: 

```
[1] 10 6 3 5 9
```

.

Insert R Cell code ▾ text section ▾

# Links

- knitr Language Engines
- More engines
- Andere Programmiersprachen können eingebunden werden
- Video - Einführung in Rnotebook
- R Notebooks
- IPython vs knitr, or Python vs R
- Datacamp Tutorial - Jupyter Notebook
- Better interactive data science with Beaker and Rodeo
- Knit directly to jupyter notebooks from RStudio
- Python-Markdown
- Podcast - die Welt von Python kennenlernen
- Deploying JupyterHub for Education
- JupyterHub - github
- Jupyter autograder

# Interaktive Tabellen mit DataTables

# The R-package DT

- DT: An R interface to the DataTables library

```
install.packages('DT')
```

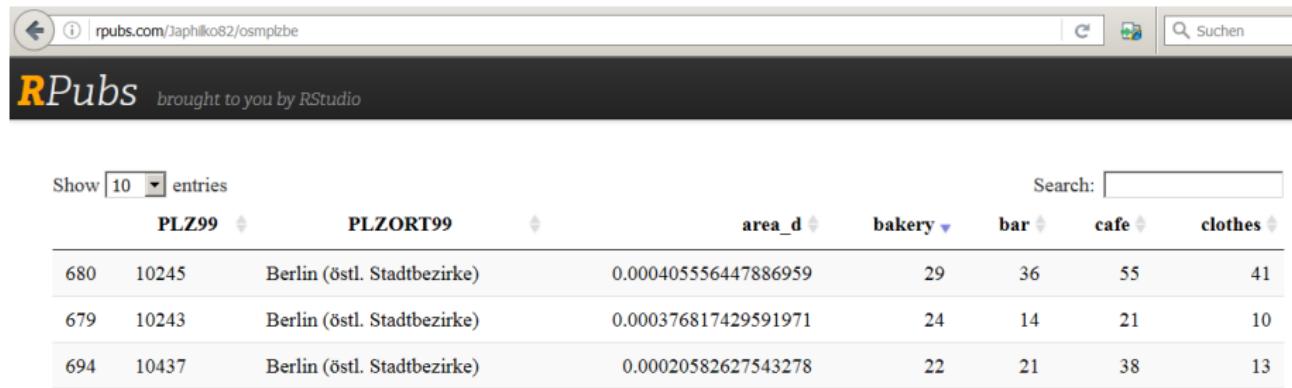
```
library('DT')
```

```
exdat <- read.csv("data/exdat.csv")
```

```
datatable(exdat)
```

# Beispiel für interaktive Tabelle

Hier ist das Ergebnis - Beispiel für eine interaktive Tabelle



The screenshot shows a web browser displaying an Rpubs page. The URL in the address bar is rpubs.com/Japhilko82/osmplzbe. The page title is "RPub" followed by "brought to you by RStudio". A search bar is visible on the right. The main content is an interactive table with the following data:

| PLZ99 |       |                             | PLZORT99             | area_d | bakery | bar | cafe | clothes |
|-------|-------|-----------------------------|----------------------|--------|--------|-----|------|---------|
| 680   | 10245 | Berlin (östl. Stadtbezirke) | 0.000405556447886959 | 29     | 36     | 55  | 41   |         |
| 679   | 10243 | Berlin (östl. Stadtbezirke) | 0.000376817429591971 | 24     | 14     | 21  | 10   |         |
| 694   | 10437 | Berlin (östl. Stadtbezirke) | 0.00020582627543278  | 22     | 21     | 38  | 13   |         |

# Default Optionen verändern

```
datatable(head(exdat, 20), options = list(  
  columnDefs = list(list(className = 'dt-center', targets = 5),  
  pageLength = 5,  
  lengthMenu = c(5, 10, 15, 20)  
))
```

# Suchoptionen kennzeichnen

```
datatable(exdat, options = list(searchHighlight = TRUE),
         filter = 'top')
```

Show  entries

Search:

|     | mpg  | cyl | disp | hp  | drat | model             |
|-----|------|-----|------|-----|------|-------------------|
| All | All  | All | All  | All | All  | All               |
| 1   | 21   | 6   | 160  | 110 | 3.9  | Mazda RX4         |
| 2   | 21   | 6   | 160  | 110 | 3.9  | Mazda RX4 Wag     |
| 3   | 22.8 | 4   | 108  | 93  | 3.85 | Datsun 710        |
| 4   | 21.4 | 6   | 258  | 110 | 3.08 | Hornet 4 Drive    |
| 5   | 18.7 | 8   | 360  | 175 | 3.15 | Hornet Sportabout |

Showing 1 to 5 of 20 entries

Previous

2 3 4 Next

# Interaktive Karten mit dem Javascript Paket leaflet

# Die Daten - Weltkulturerbe

- die Daten einlesen:

```
url <- "https://raw.githubusercontent.com/Japhilko/  
GeoData/master/2015/data/whcSites.csv"
```

```
whcSites <- read.csv(url)
```

- die Daten werden eingeschränkt:

```
whcSitesDat <- with(whcSites, data.frame(name_en,  
category))
```

# Eine Tabelle erzeugen mit knitr

```
library(knitr)  
kable(head(whcSitesDat))
```

---

name\_en

|   |      |
|---|------|
| Cultural Landscape and Archaeological Remains of the Bamiyan Valley | Cult |
| Minaret and Archaeological Remains of Jam                           | Cult |
| Historic Centres of Berat and Gjirokastra                           | Cult |
| Butrint   | Cult |
| Al Qal'a of Beni Hammad   | Cult |
| M'Zab Valley  | Cult |

# Eine erste interaktive Tabelle - Das Paket DT

```
install.packages("DT")
```

## DT: An R interface to the DataTables library

The R package **DT** provides an R interface to the JavaScript library **DataTables**. R data objects (matrices or data frames) can be displayed as tables on HTML pages, and **DataTables** provides filtering, pagination, sorting, and many other features in the tables.

You may install the stable version from CRAN, or the development version using `devtools::install_github('rstudio/DT')` if necessary (this website reflects the development version of **DT**):

# Weitere Variablen WHC Datensatz

```
whcSitesDat2 <- with(whcSites,data.frame(name_en,category,  
                                             longitude,latitude,da
```

- mit dem Befehl datatable kann man eine erste interaktive Tabelle erstellen:

```
library('DT')  
datatable(whcSitesDat2)
```

# Das Ergebnis bei Rpubs

<http://rpubs.com/Japhilko82/WHCdata>

**R**Pubs brought to you by RStudio

| Show 10 entries |                              |          |             |             |                |               |             | Search:   | <input type="text"/> |
|-----------------|------------------------------|----------|-------------|-------------|----------------|---------------|-------------|-----------|----------------------|
|                 | name_en                      | category | longitude   | latitude    | date_inscribed | area_hectares | danger_list |           |                      |
| 101             | Angkor                       | Cultural | 103.8333333 | 13.43333333 | 1992           | 40100         | P           | 1992-2004 |                      |
| 96              | Srebarna Nature Reserve      | Natural  | 27.07806    | 44.11444    | 1983           | 638           | P           | 1992-2003 |                      |
| 184             | Plitvice Lakes National Park | Natural  | 15.61444    | 44.87778    | 1979           | 29482         | P           | 1992-1997 |                      |
| 182             | Old City of Dubrovnik        | Cultural | 18.09139    | 42.65056    | 1979           | 96.7          | P           | 1991-1998 |                      |

# Das Paket magrittr

- magrittr - für den Pipe Operator in R:

```
install.packages("magrittr")
```

```
library("magrittr")
```

## Simpler R coding with pipes > the present and future of the magrittr package



Tal Galili

August 5, 2014

Guest Post, R, R  
programming

0  
SHARES

f Share

t Tweet

e Subscribe

This is a guest post by Stefan Milton, the author of the [magrittr](#) package which introduces the `%>%` operator to R programming.

# Die Pipes nutzen

```
library(magrittr)

str1 <- "Hallo Welt"
str1 %>% substr(1,5)

## [1] "Hallo"

str1 %>% substr(1,5) %>% toupper()

## [1] "HALLO"
```

# Das Paket leaflet

- leaflet - um interaktive Karten mit der JavaScript Bibliothek leaflet zu erzeugen

```
install.packages("leaflet")
```

```
library("leaflet")
```

- Bei leaflet wird mit so genannten Tiles gearbeitet.
- Robin Lovelace - The leaflet package for online mapping in R

# Was sind Tiles?

- Die Übersetzung aus dem englischen ist Fliese und dieses Bild erklärt es eigentlich ganz gut.
- Es geht um Kachelgrafiken.
- Es ist eine Grafik bezeichnet, die mosaikartig zusammengesetzt ein vielfach größeres Gesamtbild ergibt.

# Eine interaktive Karte erstellen

```
m <- leaflet() %>%
  addTiles() %>% # Add default OpenStreetMap map tiles
  addMarkers(lng=whcSites$lon,
             lat=whcSites$lat,
             popup=whcSites$name_en)

m
```

# Die Karte zeigen



# Farbe hinzufügen

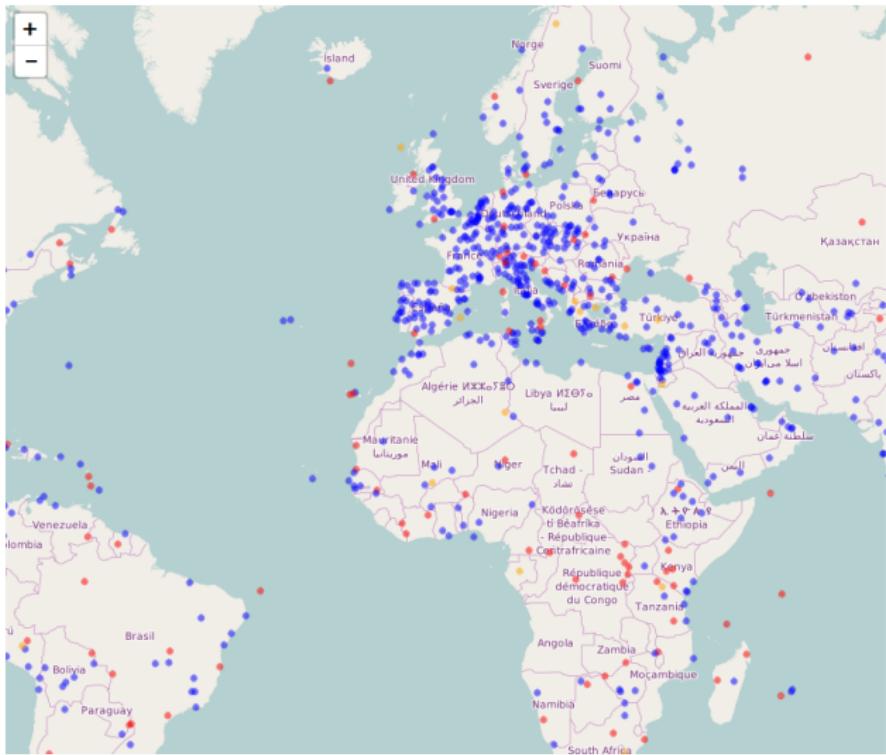
- die unterschiedlichen Kategorien farblich einfärben

```
whcSites$color <- "red"  
whcSites$color[whcSites$category=="Cultural"] <- "blue"  
whcSites$color[whcSites$category=="Mixed"] <- "orange"
```

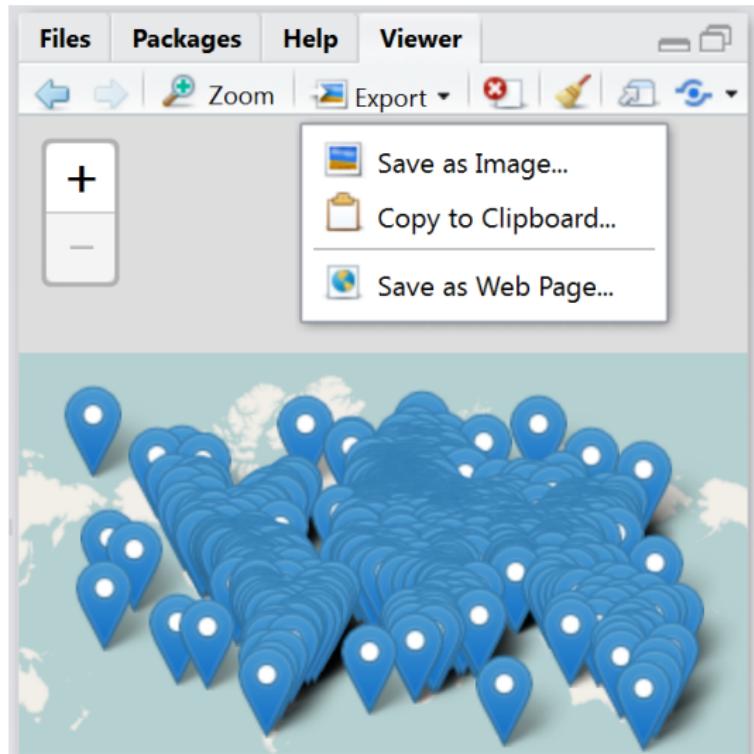
## Eine Karte mit Farbe erzeugen

```
m1 <- leaflet() %>%  
  addTiles() %>%  
  addCircles(lng=whcSites$lon,  
             lat=whcSites$lat,  
             popup=whcSites$name_en,  
             color=whcSites$color)
```

# Die Karte mit mehr Farbe



# Die Karte abspeichern



# Layers ein- und ausblenden

```
m2 <- leaflet() %>%
  addTiles(group = "OSM (default)") %>%
  addProviderTiles("Stamen.Toner", group = "Toner") %>%
  addProviderTiles("Stamen.TonerLite", group = "Toner Lite") %

  addCircles(lng=whcSites$lon,
             lat=whcSites$lat,
             popup=whcSites$name_en) %>%

addLayersControl(
  baseGroups = c("OSM (default)", "Toner", "Toner Lite"),
  options = layersControlOptions(collapsed = FALSE)
)
m2
```

Files

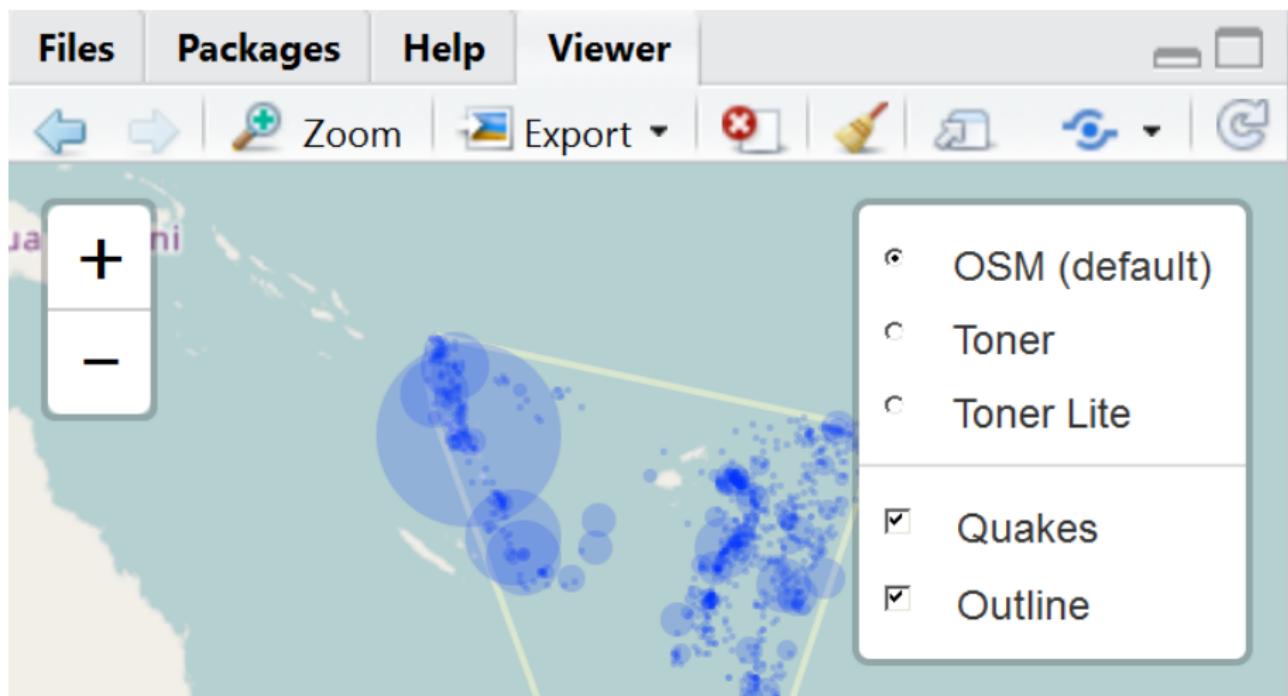
Packages

Help

Viewer



# Ein weiteres Beispiel mit Erdbebendaten



```
outline <- quakes[chull(quakes$long, quakes$lat),]
```

```
map <- Leaflet(quakes) %>%
```

# Karte mit Polygonen erzeugen

```
library(sp)
Sr1 = Polygon(cbind(c(2, 4, 4, 1, 2), c(2, 3, 5, 4, 2)))
Sr2 = Polygon(cbind(c(5, 4, 2, 5), c(2, 3, 2, 2)))
Sr3 = Polygon(cbind(c(4, 4, 5, 10, 4), c(5, 3, 2, 5, 5)))
Sr4 = Polygon(cbind(c(5, 6, 6, 5, 5), c(4, 4, 3, 3, 4)), hole=TRUE)
Srs1 = Polygons(list(Sr1), "s1")
Srs2 = Polygons(list(Sr2), "s2")
Srs3 = Polygons(list(Sr4, Sr3), "s3/4")
SpP = SpatialPolygons(list(Srs1, Srs2, Srs3), 1:3)
```

- so wird die Karte erzeugt:

```
leaflet(height = "300px") %>% addPolygons(data = SpP)
```

# Beispiel US Staaten

```
library(maps)
mapStates = map("state", fill = TRUE, plot = FALSE)
leaflet(data = mapStates) %>% addTiles() %>%
  addPolygons(fillColor = topo.colors(10, alpha = NULL), strok
```

# Der Befehl setView

- mit setView kann man bestimmen welchen Ausschnitt man für die Hintergrundkarte haben möchte
- dazu muss man die latitude und Longitude Koordinaten und ein zoom Level angegeben
- dabei kann man nur ganze Zahlen angeben
- je kleiner die Zahl, desto größer ist der Kartenausschnitt:
- level 3 - Kontinent
- level 10 - Stadt
- level 21 - Gebäude

# Die Basiskarte ändern

- Neben der Default Basiskarte kann man auch andere Hintergründe aktivieren

```
m <- leaflet() %>% setView(lng = -71.0589, lat = 42.3601, zoom = 13)
m %>% addTiles()
m %>% addProviderTiles("Stamen.Toner")
```

# Basiskarte - CartoDB

```
m %>% addProviderTiles("CartoDB.Positron")
```

# Esri.NatGeoWorldMap

```
m %>% addProviderTiles("Esri.NatGeoWorldMap")
```

# OpenTopoMap

```
m %>% addProviderTiles("OpenTopoMap")
```

# Thunderforest.OpenCycleMap

```
m %>% addProviderTiles("Thunderforest.OpenCycleMap")
```

# WMS Tiles hinzufügen

```
leaflet() %>% addTiles() %>% setView(-93.65, 42.0285, zoom = 4)
addWMSTiles(
  "http://mesonet.agron.iastate.edu/cgi-bin/wms/nexrad/n0r.cgi",
  layers = "nexrad-n0r-900913",
  options = WMSTileOptions(format = "image/png", transparent = TRUE),
  attribution = "Weather data © 2012 IEM Nexrad"
)
```

# Mehrere Layer miteinander kombinieren

```
m %>% addProviderTiles("MtbMap") %>%  
  addProviderTiles("Stamen.TonerLines",  
    options = providerTileOptions(opacity = 0.35)) %>%  
  addProviderTiles("Stamen.TonerLabels")
```

# Andere Marker benutzen

```
greenLeafIcon <- makeIcon(  
  iconUrl = "http://leafletjs.com/examples/custom-icons/leaf-g  
  iconWidth = 38, iconHeight = 95,  
  iconAnchorX = 22, iconAnchorY = 94,  
  shadowUrl = "http://leafletjs.com/examples/custom-icons/leaf  
  shadowWidth = 50, shadowHeight = 64,  
  shadowAnchorX = 4, shadowAnchorY = 62  
)  
  
leaflet(data = quakes[1:4,]) %>% addTiles() %>%  
  addMarkers(~long, ~lat, icon = greenLeafIcon)
```

# Andere Icons einfügen

- es lassen sich alle möglichen Icons einfügen

```
menIcon <- makeIcon("https://img.clipartfest.com/707b339dc88f5  
    iconWidth = 38, iconHeight = 95,  
    iconAnchorX = 22, iconAnchorY = 94)  
  
leaflet(data = quakes[1:4,]) %>% addTiles() %>%  
addMarkers(~long, ~lat, icon = menIcon)
```

# Cluster Optionen für Marker

```
leaflet(quakes) %>% addTiles() %>% addMarkers(  
  clusterOptions = markerClusterOptions()  
)
```

# Ein Rechteck hinzufügen

```
leaflet() %>% addTiles() %>%  
addRectangles(  
  lng1=-118.456554, lat1=34.078039,  
  lng2=-118.436383, lat2=34.062717,  
  fillColor = "transparent"  
)
```

# Links und Quellen

- 4 Tricks zum Arbeiten mit Leaflet
- [http://www.r-bloggers.com/  
the-leaflet-package-for-online-mapping-in-r/](http://www.r-bloggers.com/the-leaflet-package-for-online-mapping-in-r/)
- <https://rstudio.github.io/leaflet/>

# Aufgabe leaflet

- Verwenden Sie die Adresse, die Sie zuvor geokodiert haben, um eine interaktive Karte um diesen Punkt herum zu erstellen.