

# B2 - Graphics intro

Jan-Philipp Kolb

16 Oktober 2018

# Eine Graphik sagt mehr als 1000 Worte.

## Aussagen zu Graphen in R

- Die grafische Datenanalyse ist großartig.
- Gute Graphiken können zu einem besseren Verständnis beitragen.
- Die Erzeugung eines Plot ist einfach.
- Einen guten Plot zu erstellen, kann sehr lange dauern.
- Das Erstellen von Plots mit R macht Spaß.
- Mit R erstellte Diagramme haben eine hohe Qualität.
- Fast jedes Graphikformat wird von R unterstützt.
- Eine große Anzahl von Exportformaten ist in R verfügbar.

# Nicht alle Diagramme sind gleich.

- Das Basispaket enthält bereits eine Vielzahl von Plotfunktionen.
- Andere Pakete wie `lattice`, `ggplot2`, etc. erweitern diese Funktionalität.

## Handbücher, die weit über diese Einführung hinausgehen:

- Murrell, P (2006): R Graphics.
- R Development Core Group **Graphiken mit R**
- Wiki zu **R Programmierung/Graphiken**
- Martin Meermeyer **Creating Reproducible Publication Quality Graphics with R: A Tutorial**
- Institute for Quantitative Social Science at Harvard - **R Graphik Tutorial**

# Task View für Graphiken

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

**Maintainer:** Nicholas Lewin-Koh

**Contact:** nikko at haimail.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.

R's base graphics are implemented in the same way as in the S3 system developed by Becker, Chambers, and Wilks. There is a static device, which is treated as a static canvas and objects are drawn on the device through R plotting commands. The device has a set of global parameters such as margins and layouts which can be manipulated by the user using `par()` commands. The R graphics engine does not maintain a user visible graphics list, and there is no system of double buffering, so objects cannot be easily edited without redrawing a whole plot. This situation may change in R 2.7.x, where developers are working on double buffering for R devices. Even so, the base R graphics can produce many plots with extremely fine graphics in many specialized instances.

One can quickly run into trouble with R's base graphic system if one wants to design complex layouts where scaling is maintained properly on resizing, nested graphs are desired or more interactivity is needed. grid was designed by Paul Murrell to overcome some of these limitations and as a result packages like [lattice](#), [ggplot2](#), [vcd](#) or [hexbin](#) use grid for the underlying primitives. When using plots designed with grid one needs to keep in mind that grid is based on a system of viewports and graphic objects. To add objects one needs to use grid commands, e.g., `grid.polygon()` rather than `polygon()`. Also grid maintains a stack of viewports from the device and one needs to make sure the desired viewport is at the top of the stack. There is a great deal of explanatory documentation included with grid as vignettes.

The graphics packages in R can be organized roughly into the following topics, which range from the more user oriented at the top to the more developer oriented at the bottom. The categories are not mutually exclusive but are for the convenience of presentation:

<https://cran.r-project.org/web/views/Graphics.html>

# GESIS Panel Daten importieren

- Zum importieren nutzen wir die Funktion `read.dta13` aus dem Paket `readstata13`

```
dat <- readstata13::read.dta13(  
  "../data/ZA5666_v1-0-0_Stata14.dta")
```

## Geschätzte Dauer (bfzq020a)

Wie lange haben Sie gebraucht, um den Fragebogen auszufüllen?

```
dat <- readstata13::read.dta13("ZA5666_v1-0-0_Stata14.dta")
summary(dat$duration)
```

```
dat$duration <- as.numeric(dat$bfzq020a)
```

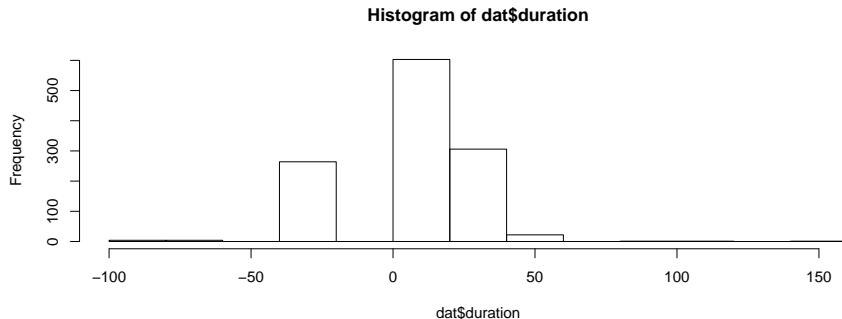
| ## | Min.   | 1st Qu. | Median | Mean  | 3rd Qu. | Max.   | NA's |
|----|--------|---------|--------|-------|---------|--------|------|
| ## | -99.00 | 10.00   | 16.00  | 10.02 | 25.00   | 156.00 | 16   |

# Histogramm - Die Funktion hist()

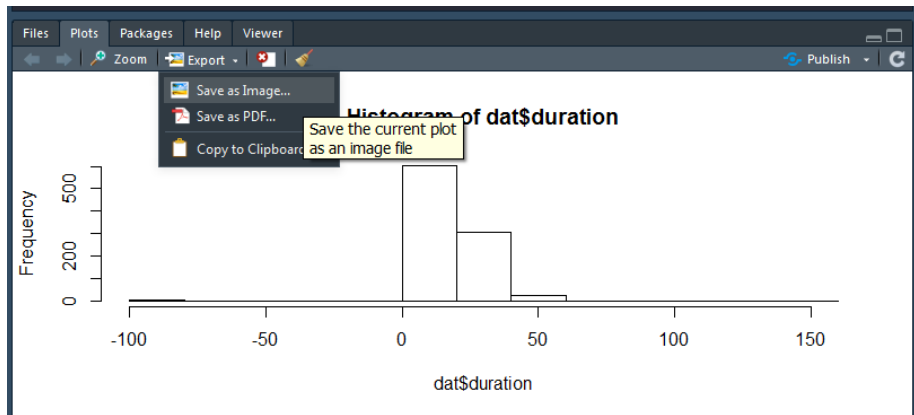
Wir erstellen ein Histogramm der Variablen Dauer:

```
?hist
```

```
hist(dat$duration)
```



# Export mit Rstudio





# Befehl zum Speichern der Grafik

- Alternativ auch mit den Befehlen `png`, `pdf` oder `jpeg` zum Beispiel.

```
png("Histogramm.png")  
  hist(dat$duration)  
dev.off()
```

```
pdf("Histogramm.pdf")  
  hist(dat$duration)  
dev.off()
```

```
jpeg("Histogramm.jpeg")  
  hist(dat$duration)  
dev.off()
```

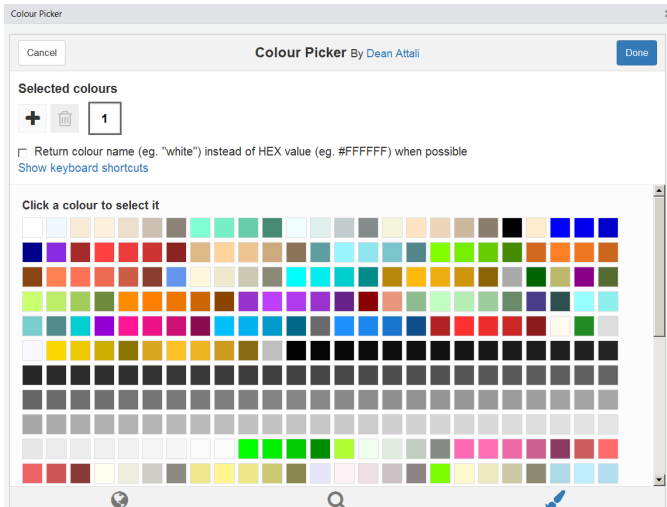
# Histogramm

- Der Befehl `hist()` zeichnet ein Histogramm.
- Mindestens ein Beobachtungsvektor muss an die Funktion übergeben werden.
- `hist()` hat viele weitere Argumente, die alle (sinnvolle) Standardwerte haben.

```
hist(dat$duration,col="blue",  
     main="Duration of interview",ylab="Frequency",  
     xlab="Duration")
```

# Rstudio Addin colourpicker

```
install.packages("colourpicker")
```



# Weitere Argumente:

?plot

# or

?par

## Graphical Parameters

adj

The value of `adj` determines the way in which text strings are justified in `text`, `mtext` and `title`. A value of 0 produces left-justified text, 0.5 (the default) centered text and 1 right-justified text. (Any value in  $[0, 1]$  is allowed, and on most devices values outside that interval will also work.)

Note that the `adj` argument of `text` also allows `adj = c(x, y)` for different adjustment in x- and y- directions. Note that whereas for `text` it refers to positioning of text about a point, for `mtext` and `title` it controls placement within the plot or device region.

ann

If set to `FALSE`, high-level plotting functions calling `plot.default` do not annotate the plots they produce with axis titles and overall titles. The default is to do annotation.

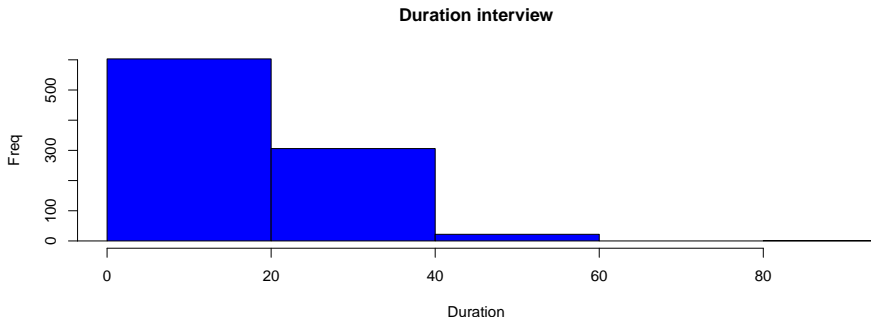
ask

logical. If `TRUE` (and the R session is interactive) the user is asked for input, before a new figure is drawn. As this applies to the device, it also affects output by packages `grid` and `lattice`. It can be set even on non-screen devices but may have no effect there.

This is not really a graphics parameter, and its use is deprecated in favour of `devAskNewPage`.

# Das xlim Argument

```
hist(dat$duration,col="blue",  
     main="Duration interview",ylab="Freq", xlab="Duration",  
     xlim=c(0,90))
```



# Das breaks Argument

- Während die vorherigen Argumente für viele Grafikfunktionen gelten, gilt das Folgende hauptsächlich für Histogramme:

```
hist(dat$duration,col="red",  
     main="Duration of interview", xlab="Duration",  
     xlim=c(0,90),breaks=60)
```

- Mit breaks kann man die Zahl der Balken kontrollieren:

# Tabellieren und barplot

```
sex <- as.character(dat$a11d054a)
sex[dat$a11d054a=="Männlich"] <- "m"
sex[dat$a11d054a=="Weiblich"] <- "f"
```

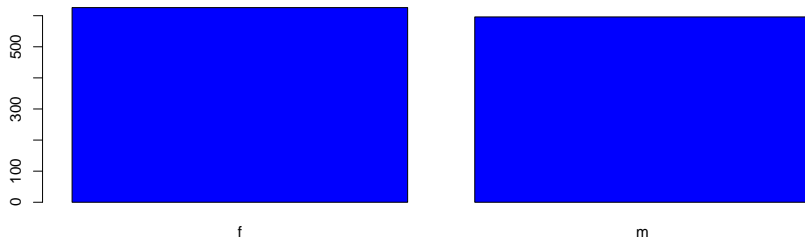
- Der Befehl `barplot()` erzeugt einen Barplot aus einer Frequenztabelle.
- Wir erhalten die Tabelle mit dem folgenden Befehl:

```
tab_sex <- table(sex)
```

```
barplot(tab_sex)
```

# Mehr Farbe:

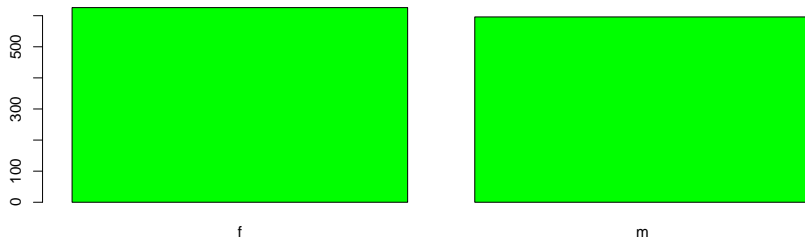
```
barplot(tab_sex,col=rgb(0,0,1))
```





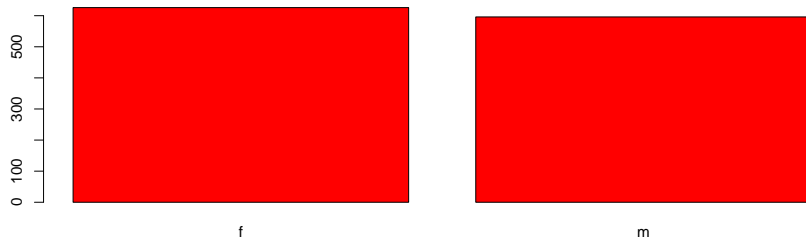
# Grüne Farbe

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



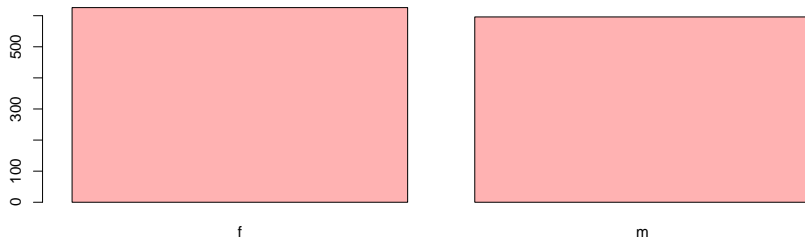
# Rote Farbe

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



# Transparent

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



# Eine zweidimensionale Tabelle

Internet-Suche nach Infos: Freunde (bbzc024a) und Geschlecht (a11d054a)

- Wenn das übergebene Tabellenobjekt zweidimensional ist, wird ein bedingter Barplot erstellt.

```
table(dat$bbzc024a,sex)
```

| ## | sex               |         |
|----|-------------------|---------|
| ## | f                 | m       |
| ## | Item nonresponse  | 25 27   |
| ## | Missing by filter | 66 50   |
| ## | Not reached       | 1 1     |
| ## | Unit nonresponse  | 79 91   |
| ## | Not in panel      | 4 6     |
| ## | Nein              | 220 213 |
| ## | Ja                | 231 208 |

# Fehlende Werte rekodieren

```
transform_miss <- function(x){  
  x[x%in%c(-11,-22,-33,-44,-55,-66,-77,-88,-99,-111)] <- NA  
  x[x%in%c("Item nonresponse","Missing by filter",  
           "Not reached","Unit nonresponse",  
           "Not in panel")] <- NA  
  return(x)  
}
```

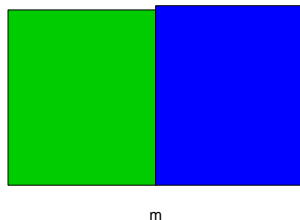
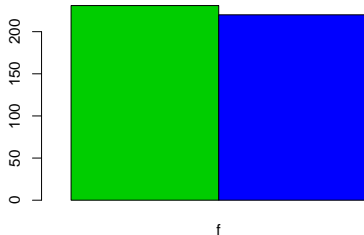
```
Inetfriends <- as.character(transform_miss(dat$bbzc024a))  
(tab2dim <- table(Inetfriends,sex))
```

```
##           sex  
## Inetfriends  f    m  
##           Ja   231 208  
##           Nein  220 213
```

# Bedingter barplot

```
barplot(tab2dim,col=1:2)
```

```
barplot(tab2dim,col=3:4,beside=T)
```

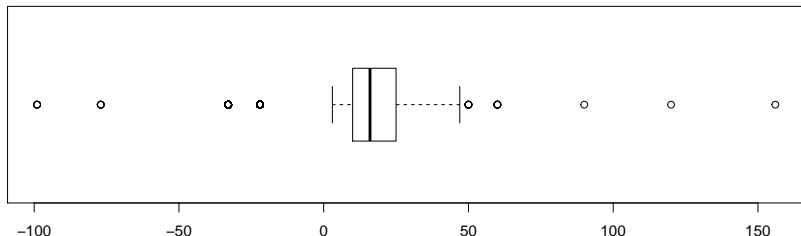


# Horizontaler Boxplot

- Ein einfacher **boxplot** kann mit `boxplot()` erstellt werden.
- Für den Befehl `boxplot()` muss mindestens ein Beobachtungsvektor übergeben werden.

```
?boxplot
```

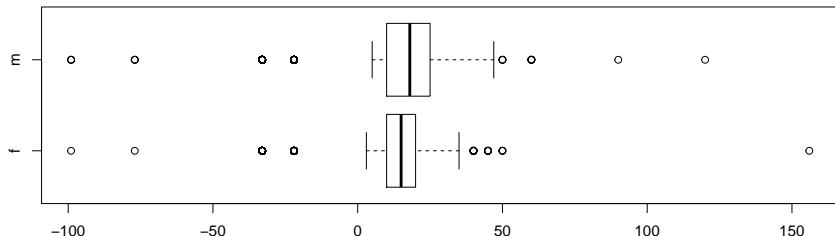
```
boxplot(dat$duration, horizontal=TRUE)
```



# Gruppierte Boxplots

- Ein sehr einfacher Weg, sich einen ersten Eindruck von bedingten Verteilungen zu verschaffen, ist über sogenannte gruppierte Boxplots.
- Dazu muss ein sogenanntes Formelobjekt an die Funktion `boxplot()` übergeben werden.
- Die bedingte Variable befindet sich auf der rechten Seite einer Tilde.

```
boxplot(dat$duration~sex, horizontal=TRUE)
```

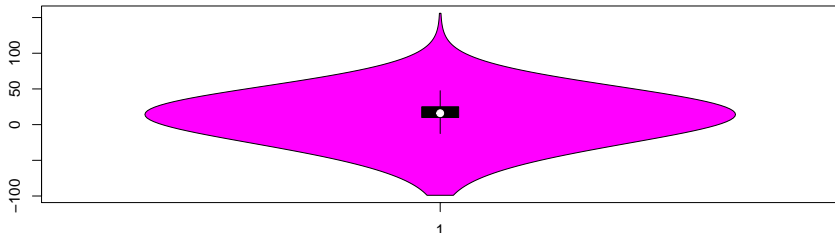




# Boxplot Alternativen - vioplot

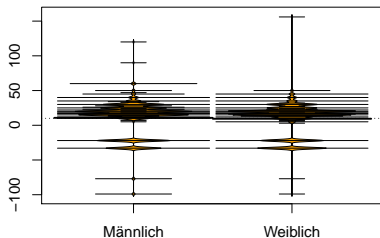
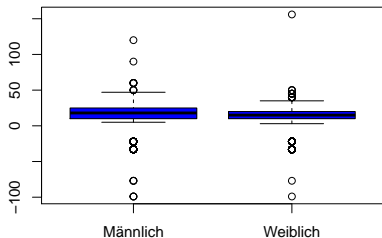
- Baut auf dem boxplot auf - Zusatzinformationen zur Dichte
- Die Dichte wird mit der Kernel-Methode berechnet.
- Je weiter die Ausdehnung, desto höher ist die Dichte an dieser Stelle.
- Weißer Punkt - Medianwert

```
library(vioplot)  
vioplot(na.omit(dat$duration))
```



# Alternativen zum boxplot()

```
library(beanplot)
par(mfrow = c(1,2))
boxplot(dat$duration~dat$a11d054a,data=dat,col="blue")
beanplot(dat$duration~dat$a11d054a,data=dat,col="orange")
```

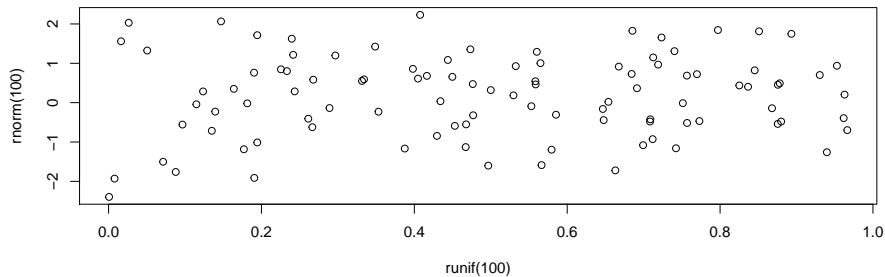


# Bedingte, bi- und multivariate Graphiken - Scatterplots

- Ein einfaches Streudiagramm kann mit der Funktion `plot()` erstellt werden.
- Um ein Scatterplot zu erstellen, müssen `x` und `y` als Beobachtungsvektoren übergeben werden.
- Argument `col` - Farbe als Zeichen oder numerisch
- Argument `pch` - Plotsymbol als Zeichen oder numerisch
- Achsenbeschriftung wird mit `xlab` und `ylab` definiert.

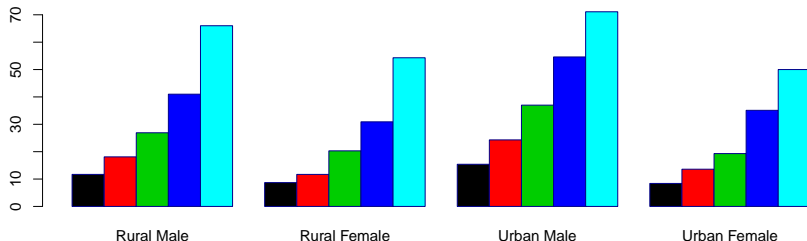
# Scatterplot

```
plot(runif(100), rnorm(100))
```



## B2A Übung - einfache Grafiken

- Laden Sie den Datensatz `VADeaths` und erstellen Sie die folgende Darstellung:

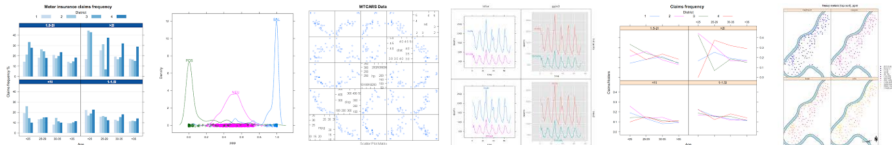


# Das lattice-Paket

## Definition einer lattice Graphik

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

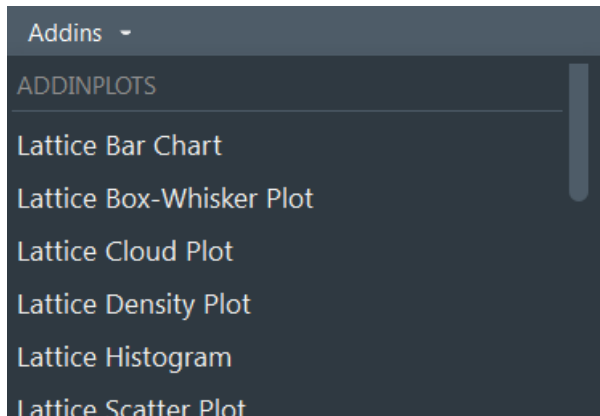
## Beispiele für lattice Graphiken



# Ein weiteres Addin für RStudio

- das `addinplots`-Paket installieren - den Datensatz markieren, der visualisiert werden soll, und einen Plottyp wählen:

```
devtools::install_github("homerhanumat/addinplots")
```



# Benutzer Interface für addinplots

Cancel

Histogram Code-Helper

Data

Choose the numerical variable.

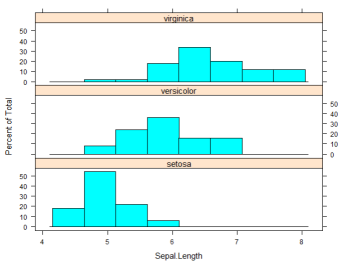
x

Group

Facet

Other

The Plot



Facet by:

Also facet by:

Rows in Layout

Columns in Layout

☐ Show Facet-Variable Names

The Code

```
lattice::histogram(~ Sepal.Length | Species,  
  data = iris,  
  layout = c(1,3),  
  type = "percent")
```

iris # Beispieldatensatz



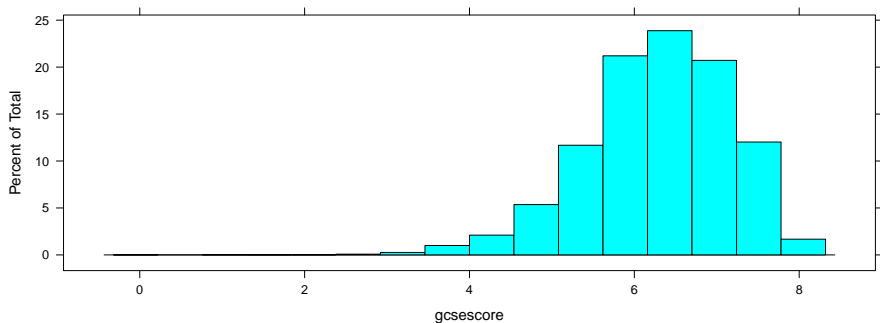
# Ein Beispieldatensatz - Testergebnisse bei A-level Chemie Test aus dem Jahr 1997

```
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  |
| gcse_score | Average GCSE score of individual                   |
| gcse_cnt   | Average GCSE score of individual, centered at mean |

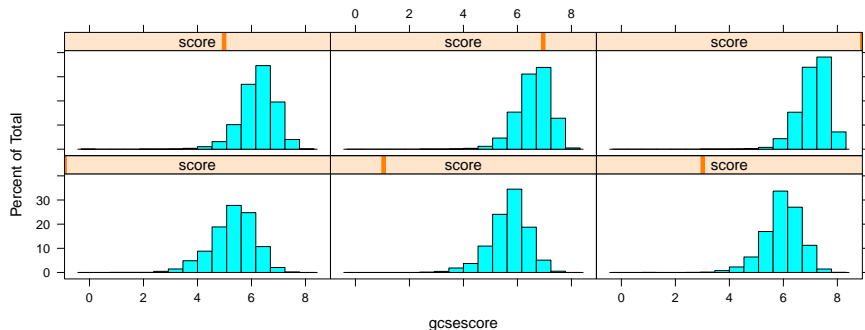
# Histogramm mit lattice

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



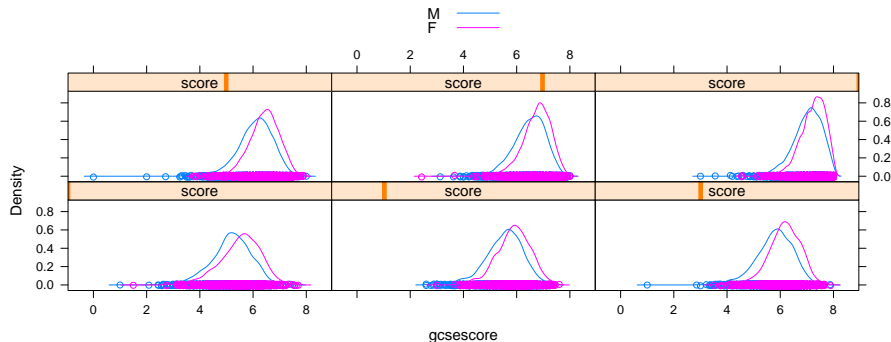
# Mehr Histogramme mit lattice

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



# Die Dichte plotten mit einer Legende

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

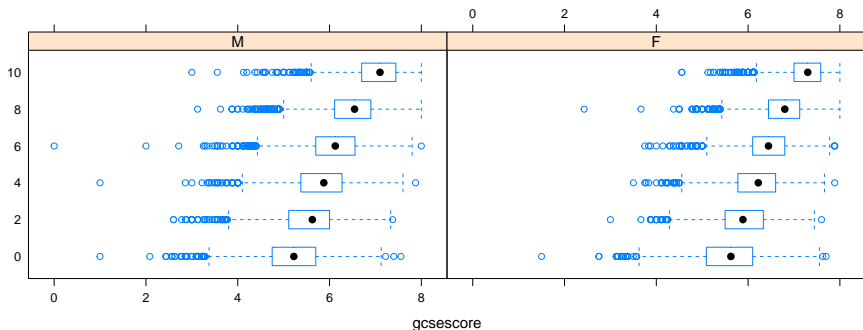


## Einführung in das lattice Paket

# Einen Boxplot mit lattice erzeugen

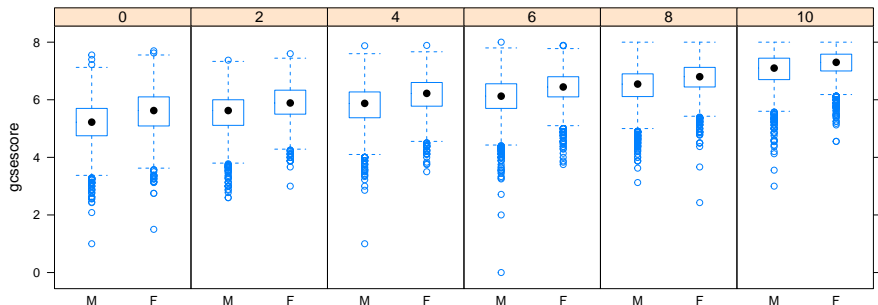
```
Chem97$score <- as.factor(Chem97$score)
```

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



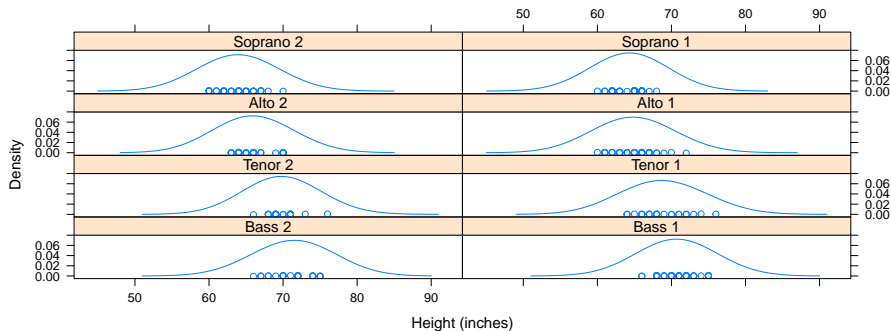
# Bedingte Boxplots mit lattice erzeugen

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



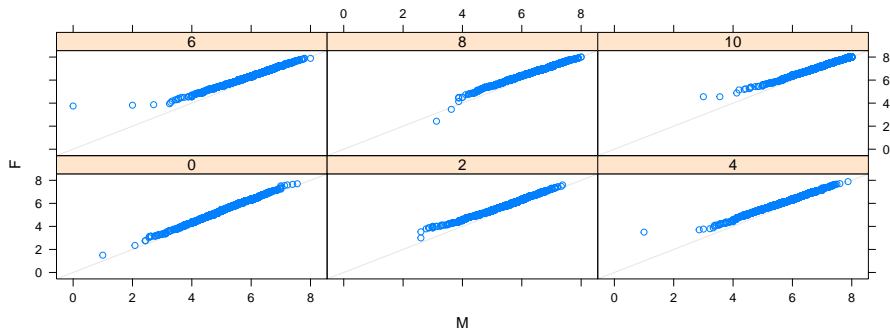
# Ein densityplot

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



# Bivariate Plots - Quantile-Quantile Plot

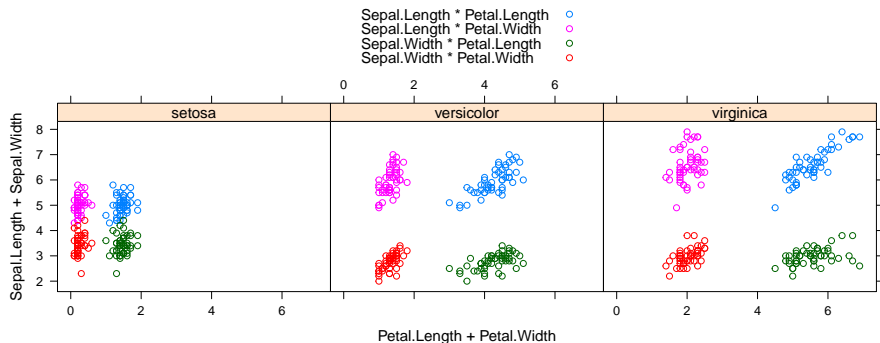
```
qq(gender ~ gcsescore | score, Chem97)
```





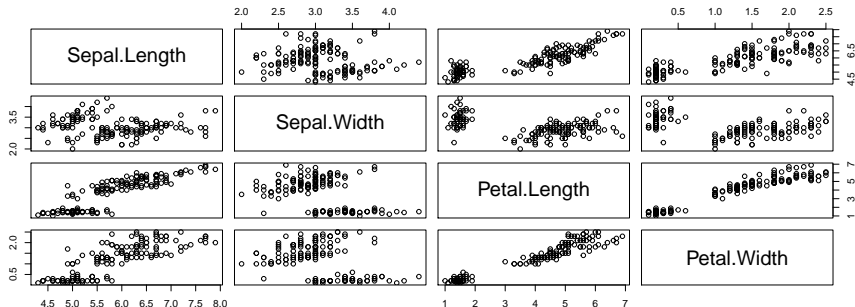
# Scatterplot mit lattice - xyplot

```
xyplot(Sepal.Length+Sepal.Width~Petal.Length+Petal.Width  
| Species,data = iris, auto.key = T)
```



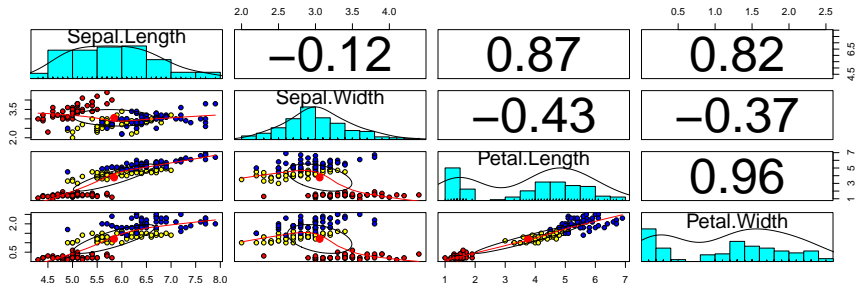
# Zusammenhang zwischen Variablen - pairs Plot

```
pairs(iris[,1:4])
```



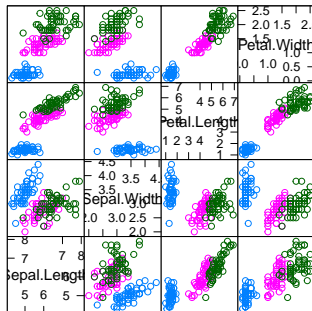
# Den pairsPlot erweitert

```
library("psych")  
pairs.panels(iris[,1:4],  
             bg=c("red","yellow","blue")[iris$Species],  
             pch=21,main="")
```



# Multivariate Plots - splom

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



Scatter Plot Matrix

# Mehr Argumente im splom Befehl

```
super.sym <- trellis.par.get("superpose.symbol")
splom(~iris[1:4], groups = Species, data = iris,
      panel = panel.superpose,
      key = list(title = "Three Varieties of Iris",
                  columns = 3,
                  points = list(pch = super.sym$pch[1:3],
                                col = super.sym$col[1:3]),
                  text = list(c("Setosa", "Versicolor",
                                "Virginica"))))
```

# Der Beispieldatensatz BankWages

```
install.packages("AER")
```

```
library("AER")  
data(BankWages)
```

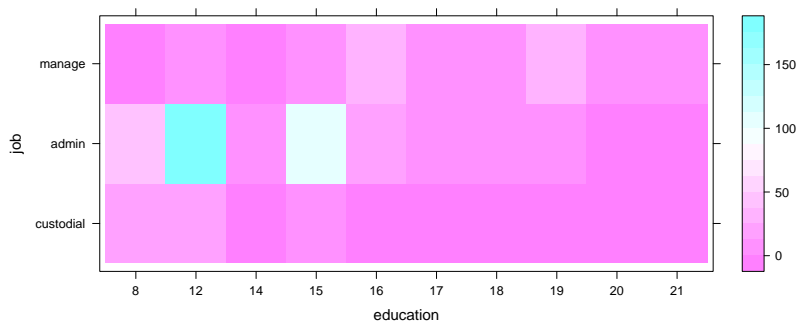
```
head(BankWages)
```

| ##   | job    | education | gender | minority |
|------|--------|-----------|--------|----------|
| ## 1 | manage | 15        | male   | no       |
| ## 2 | admin  | 16        | male   | no       |
| ## 3 | admin  | 12        | female | no       |
| ## 4 | admin  | 8         | female | no       |
| ## 5 | admin  | 15        | male   | no       |
| ## 6 | admin  | 15        | male   | no       |

# levelplot

- education in Jahren

```
library("lattice")  
levelplot(table(BankWages$education, BankWages$job),  
          xlab="education", ylab="job")
```



# Nutzung sozialer Netzwerke: Facebook (bbzc041a)

- 1 - Nein, bin kein Mitglied; 2 - Ja, nutze es aber nie; 3 - Ja, nutze es manchmal; 4 - Ja, nutze es oft

```
facebook <- transform_miss(datf$bbzc041a)
table(facebook)
```

```
## facebook
##      1      2      3      4
## 512    57   178   188
```



# Nutzung sozialer Netzwerke: Twitter (bbzc042a)

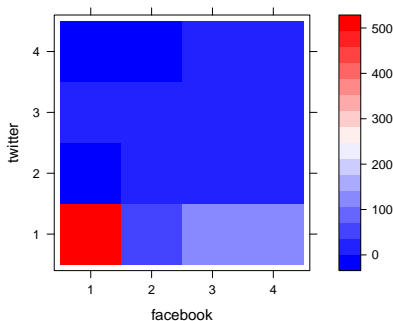
- 1 - Nein, bin kein Mitglied; 2 - Ja, nutze es aber nie; 3 - Ja, nutze es manchmal; 4 - Ja, nutze es oft

```
twitter <- as.character(transform_miss(datf$bbzc042a))  
table(twitter)
```

```
## twitter  
##      1      2      3      4  
## 791   38   20      6
```

# levelplot mit GESIS Panel Daten

```
levelplot(table facebook, twitter),  
col.regions=colorRampPalette(c("blue", "white", "red")))
```



# Internet Nutzung (GESIS Panel)

- a11c035a: Häufigkeit private Internetnutzung: Tischcomputer

```
internet <- transform_miss(datf$a11c035a)
```

- a11c037a: Häufigkeit private Internetnutzung: Smartphone

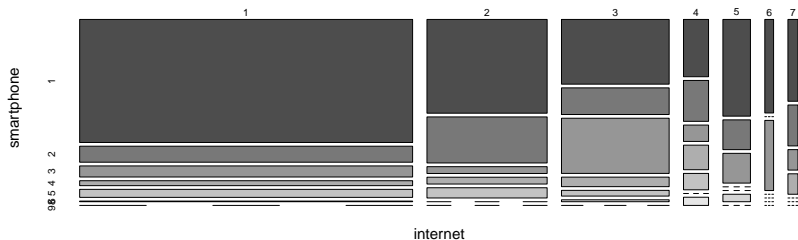
```
smartphone <- transform_miss(datf$a11c037a)
```

1 - Mehrmals täglich; 2 - Etwa einmal täglich; 3 - Mehrmals die Woche; 4 - Etwa einmal die Woche; 5 - Seltener; 6 - Nie; 98 - Weiß nicht

```
tab2 <- table(internet,smartphone)
```

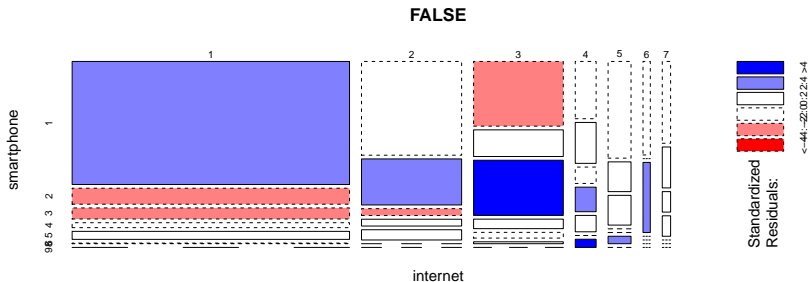
# Zusammenhang - kategoriale Variablen

```
mosaicplot(tab2, color = TRUE, main="")
```



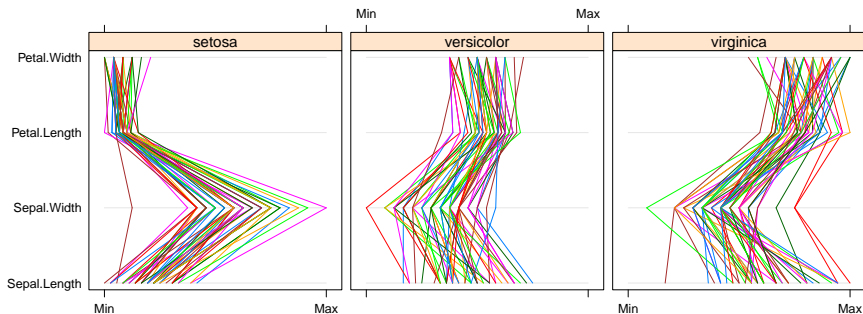
# Die Flächen werden anhand der Residuen eingefärbt:

```
mosaicplot(tab2, main=F, shade = TRUE)
```



# parallelplot()

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

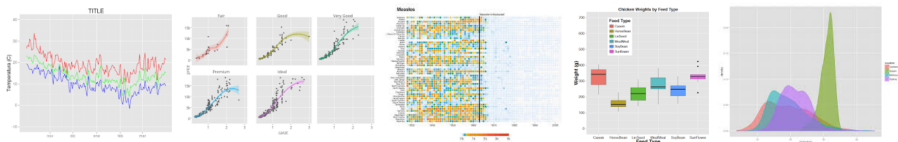


# Das ggplot2 Paket

## Einführung ggplot2

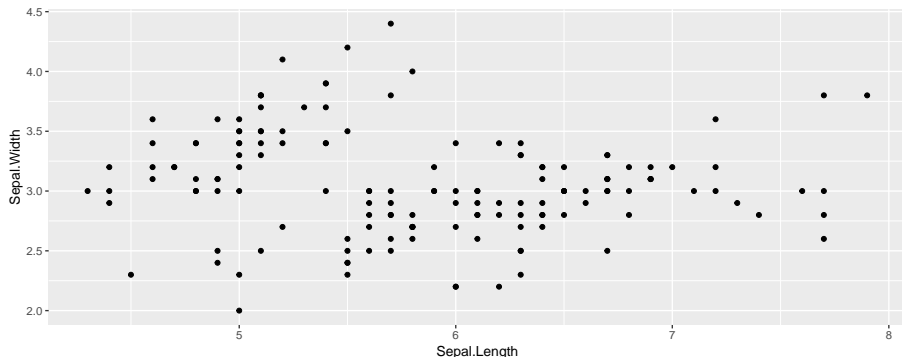
*The ggplot2 package, created by Hadley Wickham, offers a powerful graphics language for creating elegant and complex plots. Its popularity in the R community has exploded in recent years. Originally based on Leland Wilkinson's The Grammar of Graphics, ggplot2 allows you to create graphs that represent both univariate and multivariate numerical and categorical data in a straightforward manner.*

## Beispiele ggplot2 Graphiken



# Ein erstes Beispiel ggplot2

```
library(ggplot2)
ggplot(iris, aes(x=Sepal.Length, y=Sepal.Width)) +
  geom_point()
```





# Einige schöne Rstudio Addins

- Eine ggplot Grafik muss im Quellcode markiert werden, um die folgenden Addins zu verwenden

```
install.packages("ggThemeAssist")
```

## Plot Background

Fill

None ▼

Type

blank ▼

## Panel Background

Fill

gray92 ▼

Type

blank ▼

## Grid Major

Type

solid ▼

## Grid Minor

Type

solid ▼

```
install.packages('ggedit')
```

Cancel

Edit ggplots themes and layer aesthetics

Done

Update Plot Layer

Update Plot  
Theme

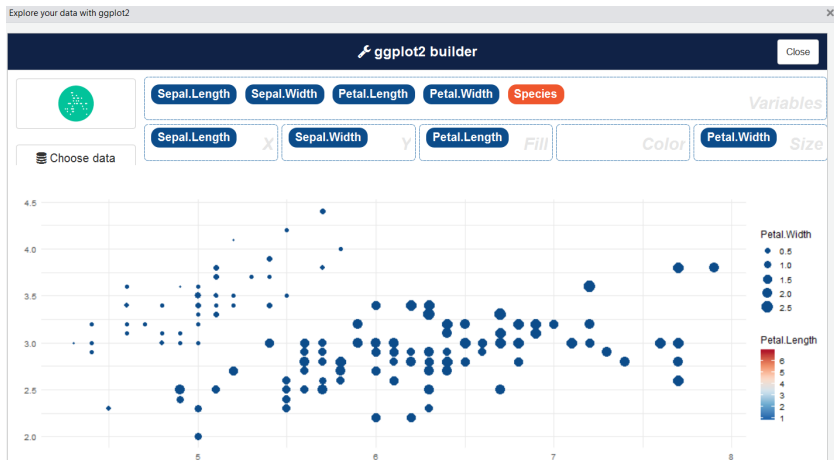
Update Grid  
Theme

Update Global Theme

View Layer  
Code

# RStudio Addin zum Erzeugen von ggplot2 Graphiken

```
devtools::install_github("dreamRs/esquisse")
```



# Shiny App - R Graphik Katalog

<http://shinyapps.stat.ubc.ca/r-graph-catalog/>

## R Graph Catalog

### About

This catalog is a complement to "Creating More Effective Graphs" by Naomi Robbins. All graphs were produced using the R language and the add-on package `ggplot2`, written by Hadley Wickham. The gallery is maintained by Joanna Zhao and Jennifer Bryan.

[More...](#)

#### Filter by type

- ☐ Good
- ☐ Not Recommended

#### Filter by tags

- ☐ Dot Plot
- ☐ Line Graph
- ☐ Pie Chart
- ☐ Bar Chart
- ☐ Scatterplot
- ☐ Trellis Display
- ☐ Histogram



[Go to GitHub to download figure and code](#)

# Ein Beispieldatensatz zu Diamanten

*A dataset containing the prices and other attributes of almost 54,000 diamonds.*

- price - price in US dollars (\$326–\$18,823)
- carat - weight of the diamond (0.2–5.01)
- cut - quality of the cut (Fair, Good, Very Good, Premium, Ideal)
- color - diamond colour, from J (worst) to D (best)
- clarity - a measurement of how clear the diamond is (I1 (worst), SI2, SI1, VS2, VS1, VVS2, VVS1, IF (best))

```
data(diamonds)
```

- Der Datensatz ist zu groß für unsere Anwendungszwecke:

```
d <- diamonds[sample(nrow(diamonds), 1000), ]
```

# Das Paket plotly

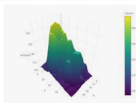
Create Interactive Web Graphics via 'plotly.js'

library(plotly)

heatmap shiny python linked scatter plot 3d scatter rstudio ggplot2 charts excel symbols graph interactive bubble gif pie chart



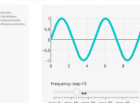
plotly for R  
[plotly-book.cpsievert.me](http://plotly-book.cpsievert.me)



3D Surface Plots in R  
[plot.ly](http://plot.ly)



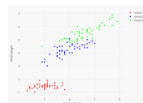
Box Plots in R | Examples | Plotly  
[plot.ly](http://plot.ly)



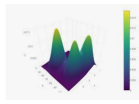
R Graphing Library | Plotly  
[plot.ly](http://plot.ly)



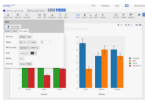
Getting Started with Plotly for R  
[plot.ly](http://plot.ly)



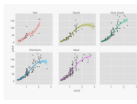
Scatter and Line Plots in R | Examples | PL...  
[plot.ly](http://plot.ly)



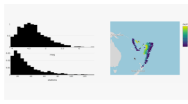
3D Surface Plots in R  
[plot.ly](http://plot.ly)



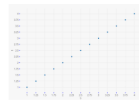
Using Plotly with R for Project Workflows (R...  
[blog.revolutionanalytics.com](http://blog.revolutionanalytics.com)



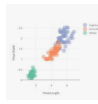
Five Interactive R Visualizations With D3...  
[moderndata.plot.ly](http://moderndata.plot.ly)



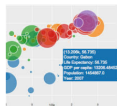
plotly 4.7.1 now on CRAN | Modern Data  
[moderndata.plot.ly](http://moderndata.plot.ly)



Modifying Axes in R | Examples | Plotly  
[plot.ly](http://plot.ly)



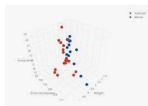
121 Manage colors with plotly...  
[r-graph-gallery.com](http://r-graph-gallery.com)



R Graphing Library | Plotly  
[plot.ly](http://plot.ly)



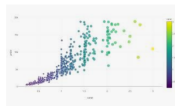
Modifying Axes in R | Examples | Plotly  
[plot.ly](http://plot.ly)



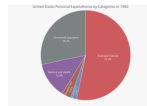
3D Scatter Plots in R | Examples | Plotly  
[plot.ly](http://plot.ly)



110 Basic interactive boxplot | plot...  
[r-graph-gallery.com](http://r-graph-gallery.com)



Exporting PNG files from Plotly in R - Stack Overflow  
[stackoverflow.com](http://stackoverflow.com)



Pie Charts in R | Examples | Plotly  
[plot.ly](http://plot.ly)

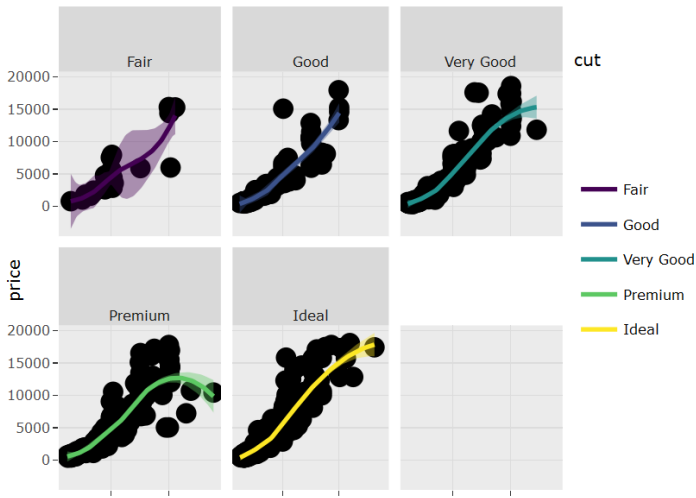
# Interaktivität hinzufügen

```
p <- ggplot(data = d, aes(x = carat, y = price)) +  
  geom_point(aes(text = clarity, size = 4)) +  
  geom_smooth(aes(colour = cut, fill = cut)) +  
  facet_wrap(~ cut)
```

Es wird eine ggplot Graphik erzeugt

# Das Ergebnis - eine interaktive Graphik

`ggplotly(p)`



- J H Maindonald - **Lattice and Other Graphics in R**
- Deepayan Sarkar - **An introduction to R - lattice lab**
- Flowingdata - **Comparing ggplot2 and R Base Graphics**
- **Quick R - ggplot2**
- **Top 50 ggplot2 Visualizations**
- **Bioconductor R manual** with an extensive part on graphics
- Shiny app to visualize **ggplot2 internals**
- **Shiny app** for **interactive plot editing**