B2 - Graphics intro

Kolb / Murray-Watters

07 August 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.

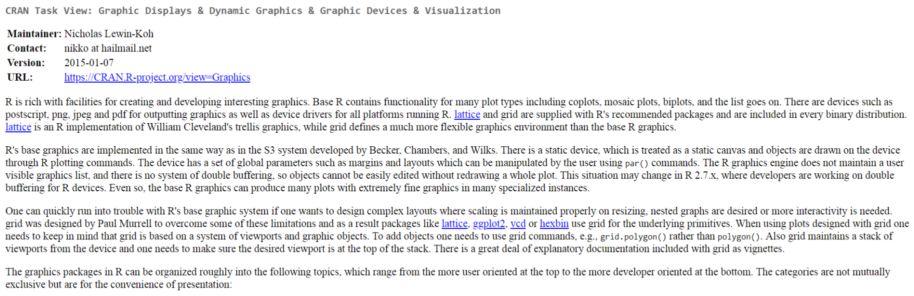
Nicht alle Diagramme sind gleich.

- 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 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.

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)
```

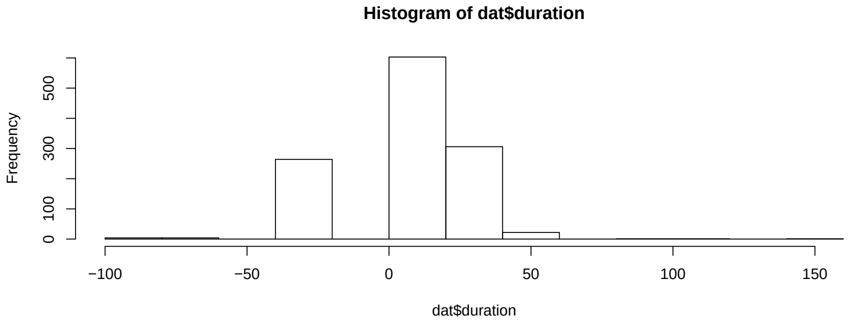
```
##     Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's 
##   -99.00   10.00   16.00   10.02   25.00  156.00      16
```

Histogramm - Die Funktion `hist()`

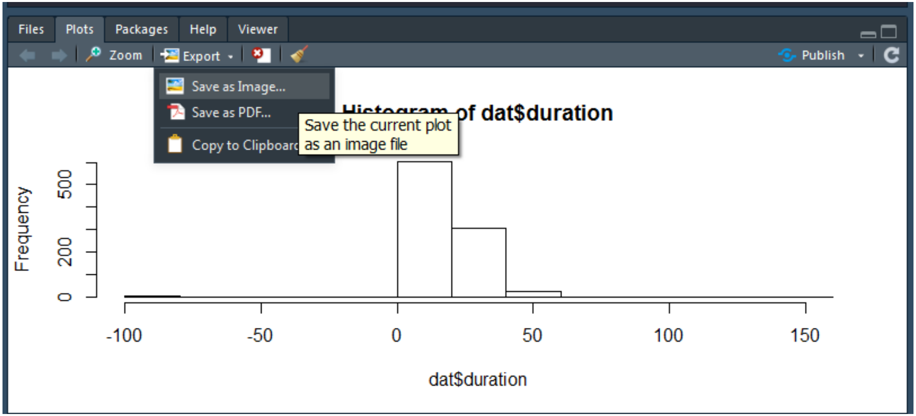
Wir erstellen ein Histogramm der Variablen `Dauer`:

```
?hist
```

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



Export with Rstudio



Command to save graphic

- Alternatively also with the commands png, pdf or jpeg for example

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

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

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

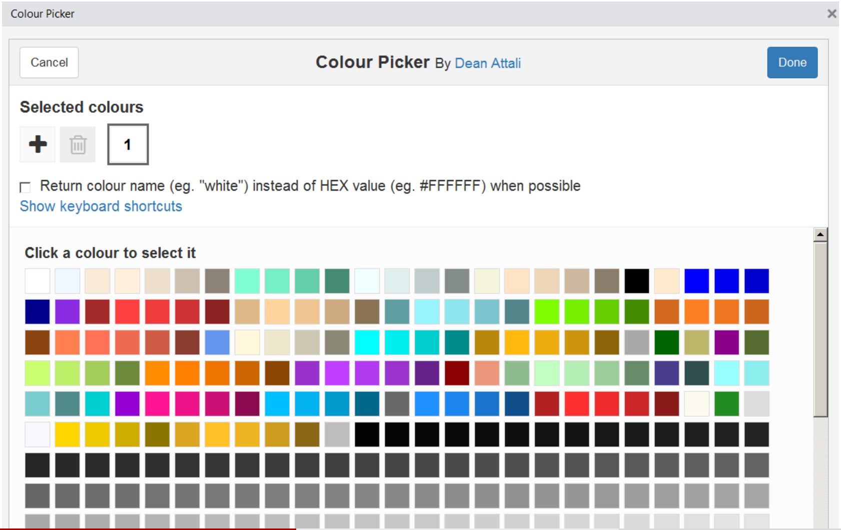
Histogram

- Command `hist()` plots a histogram
- At least one observation vector must be passed to the function
- `hist()` has many more arguments, which all have (meaningful) default values

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

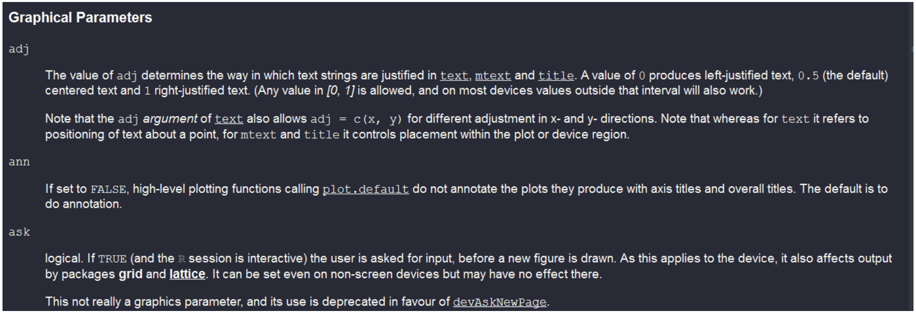
Rstudio addin colourpicker

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


Colour Picker
Cancel
Colour Picker By Dean Attali
Done
Selected colours
1
Return colour name (eg. "white") instead of HEX value (eg. #FFFFFF) when possible
Show keyboard shortcuts
Click a colour to select it

Further arguments:

```r
?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.

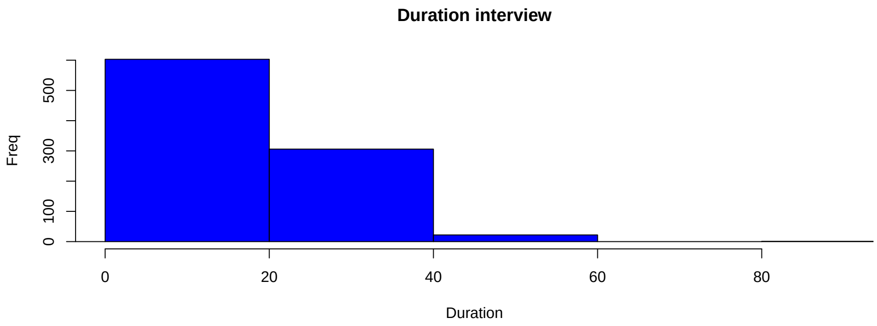
`ask`

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

The `xlim` argument

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



The `breaks` argument

- While the previous arguments are valid for many graphics functions, the following apply mainly to histogrames:

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

- with `breaks` you can control the number of bars...

Tabulate and `barplot`

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

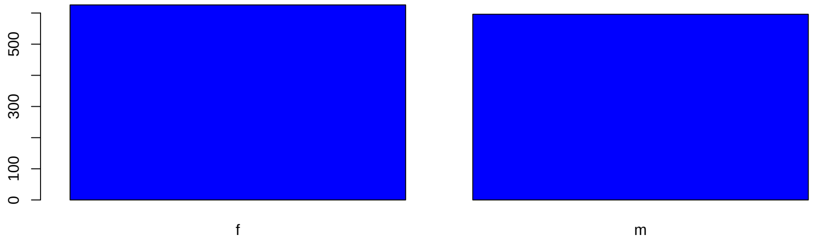
- The command `barplot()` generates a barplot from a frequency table
- We get the frequency table with the following command:

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

```
barplot(tab_sex)
```

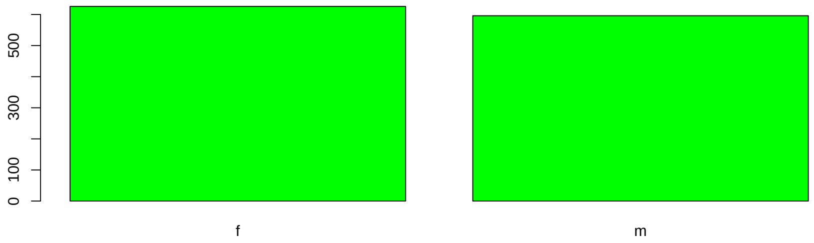
More colour:

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



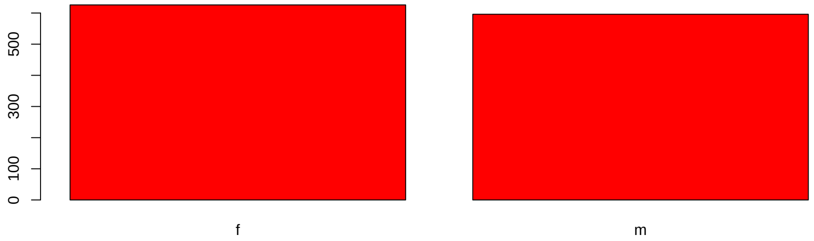
Green colour

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



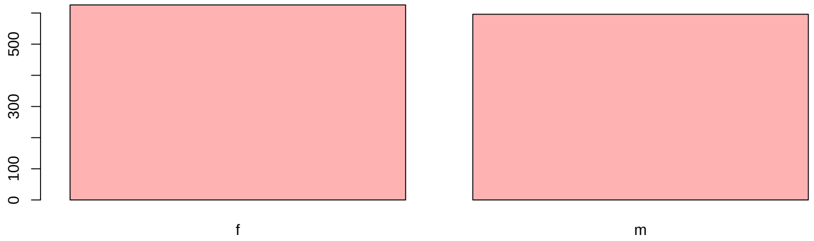
Red colour

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



Transparent

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



A two dimensional table

Internet search for information: Friends (bbzc024a) and gender (a11d054a)

```
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
```

- If the passed table object is two-dimensional, a conditional barplot is created

Recode the missing values

```r
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)
}
```

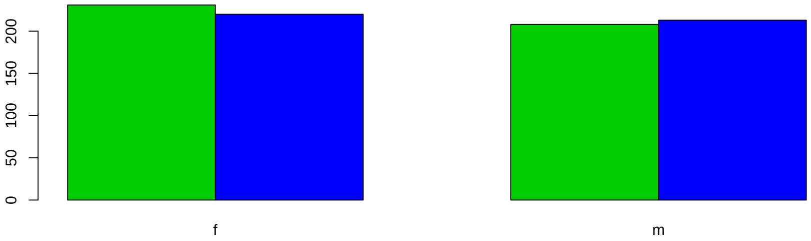
```r
Inetfriends <- as.character(transform_miss(dat$bbzc024a))
(tab2dim <- table(Inetfriends,sex))
```

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

Conditional `barplot`

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

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

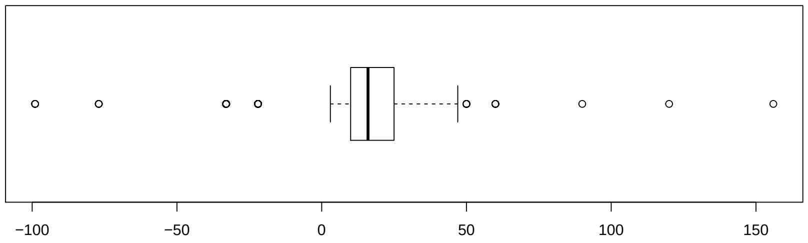


Horizontal `boxplot`

- A simple **boxplot** can be created with `boxplot()`
- For the command `boxplot()` at least one observation vector must be passed

```
?boxplot
```

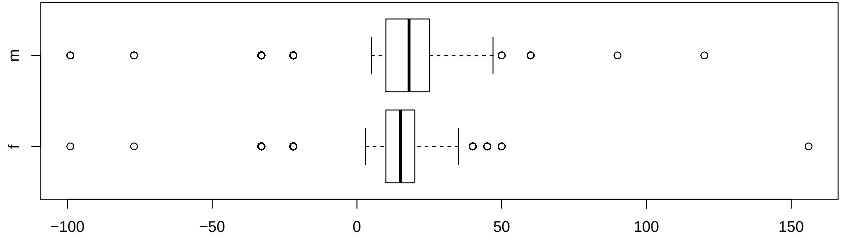
```
boxplot(dat$duration,horizontal=TRUE)
```



Grouped boxplots

- A very simple way to get a first impression of conditional distributions is via so-called grouped notched boxplots
- To do this, a so-called formula object must be passed to the `boxplot()` function.
- The conditional variable is located on the right side of a tilde

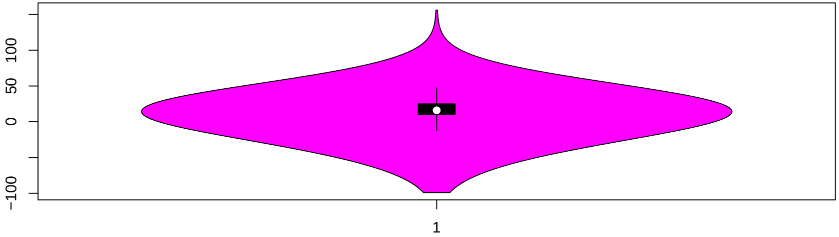
```
boxplot(dat$duration~sex,horizontal=TRUE)
```



Boxplot alternatives - `vioplot`

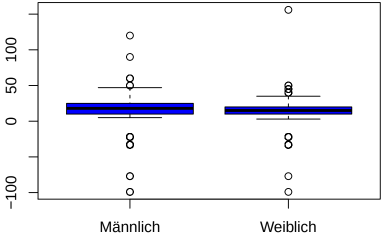
- Builds on Boxplot - additional information about data density
- Density is calculated using the kernel method.
- The further the expansion, the higher the density at this point.
- White dot - median

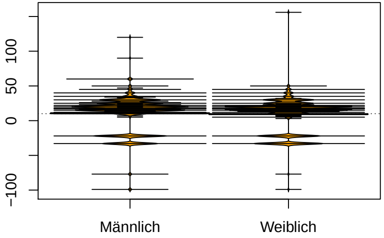
```
library(vioplot)
vioplot(na.omit(dat$duration))
```



Alternatives boxplot()

```r
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")
```

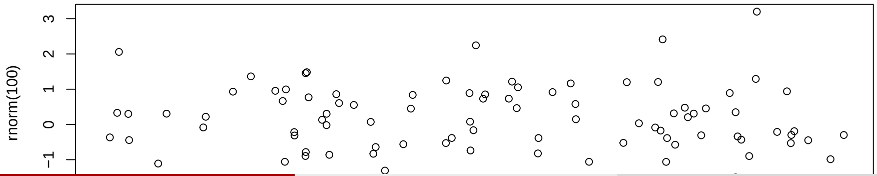




Conditional, bi- and multivariate distribution graphics - scatterplots

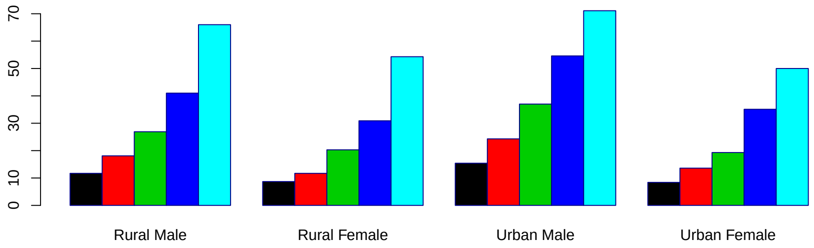
- A simple two-way scatterplot can be created with the `plot()` function
- To create a scatterplot x and y observation vector must be passed
- Argument `col` to specify the color (color as character or numeric)
- Argument `pch` to specify plot symbols (plotting character) (character or numeric)
- The labels are defined with `xlab` and `ylab`.

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



B2A Exercise - simple graphics

- Load the dataset `VADeaths` and create the following plot:

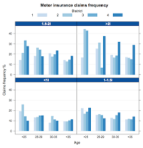


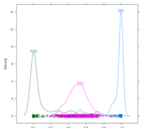
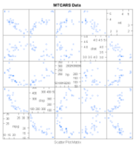
The lattice-Package

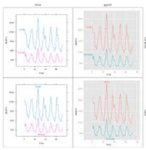
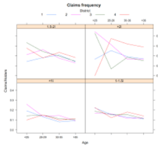
Definition of a lattice graphic

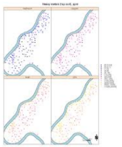
It is designed to meet most typical graphics needs with minimal tuning, but can also be easily extended to handle most nonstandard requirements.

Examples for lattice graphics



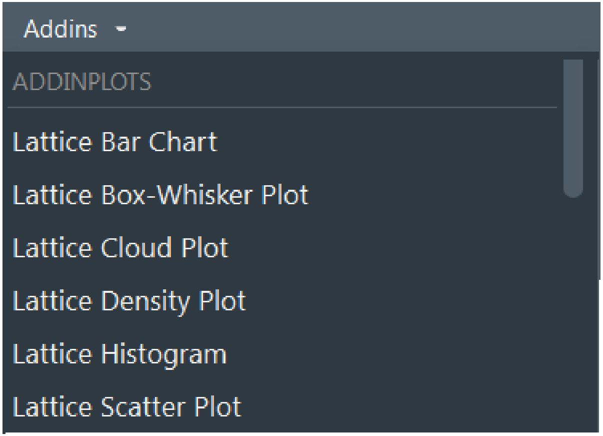





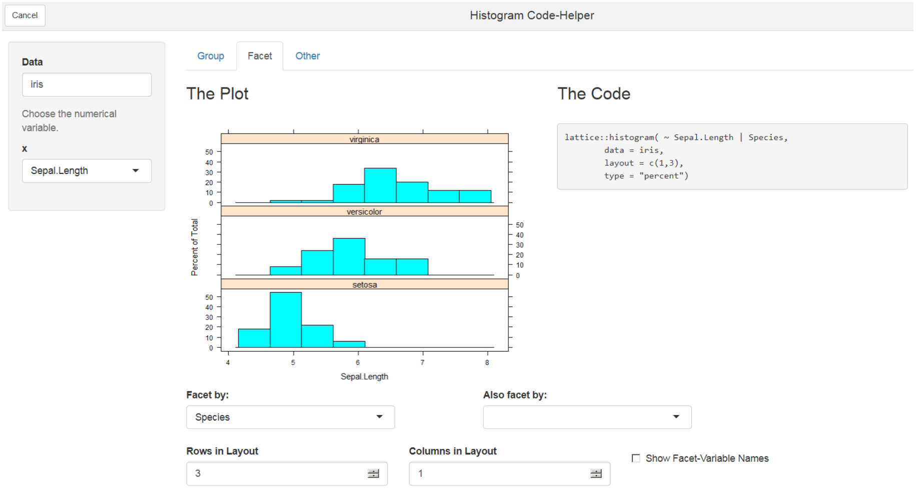
A addin for RStudio

- install the addinplots package - mark the dataset you want to visualize and choose a plot type:

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



User interface of `addinplots`



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

Facet by: Species

Also facet by:

Rows in Layout: 3

Columns in Layout: 1

☐ Show Facet-Variable Names

```
iris # example dataset used
```

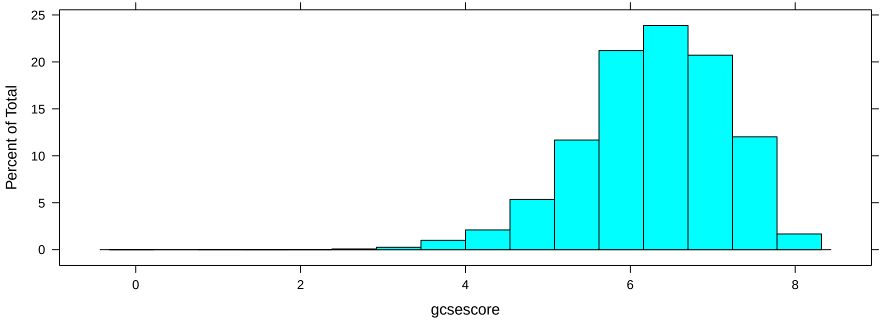
The dataset - Scores on A-level Chemistry in 1997

```r
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

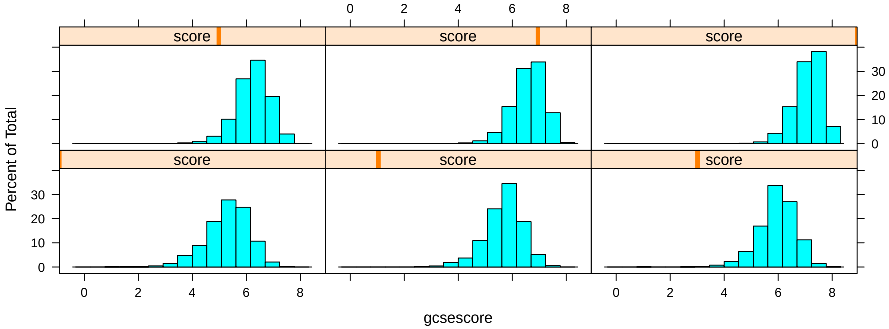
Histogram with `lattice`

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



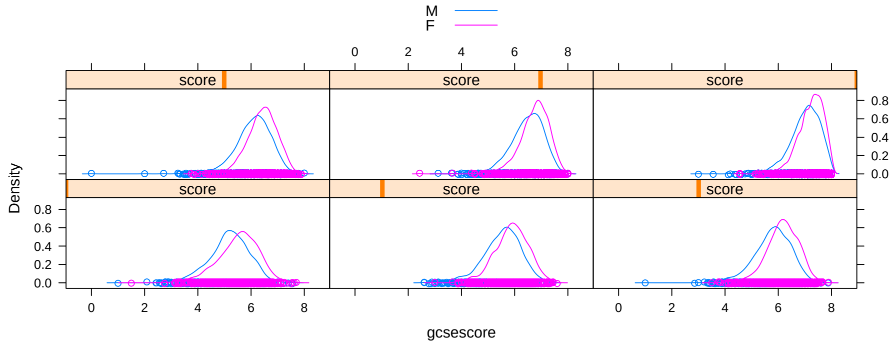
More histograms with `lattice`

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



Plotting the density with a legend

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

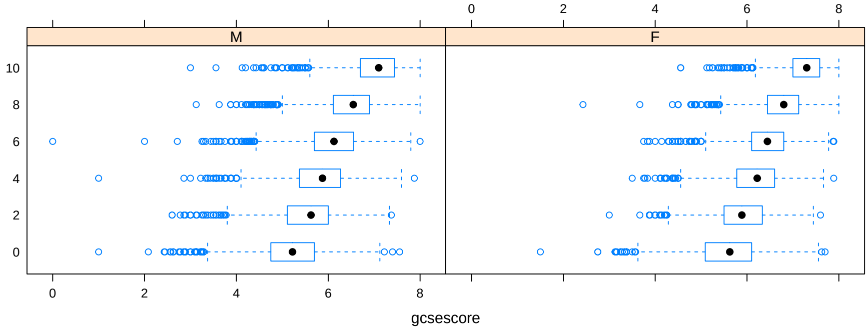


Introduction to the `lattice` package

Creating a boxplot with `lattice`

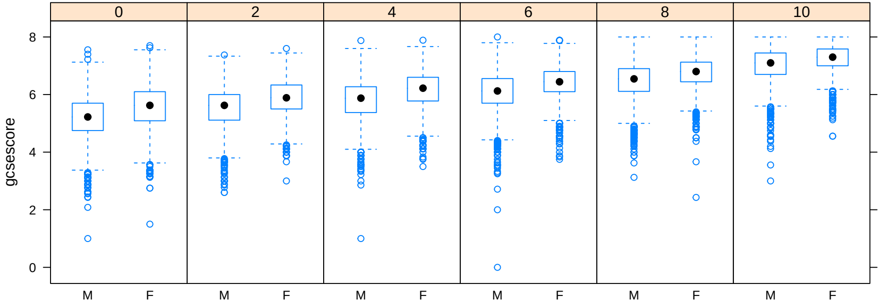
```
Chem97$score <- as.factor(Chem97$score)
```

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



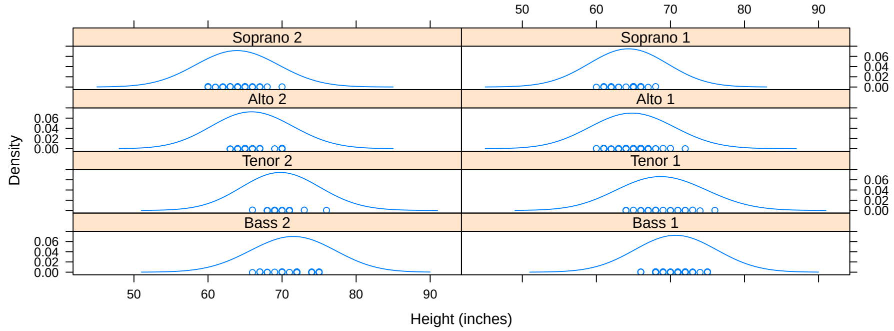
Plotting a boxplot with lattice

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



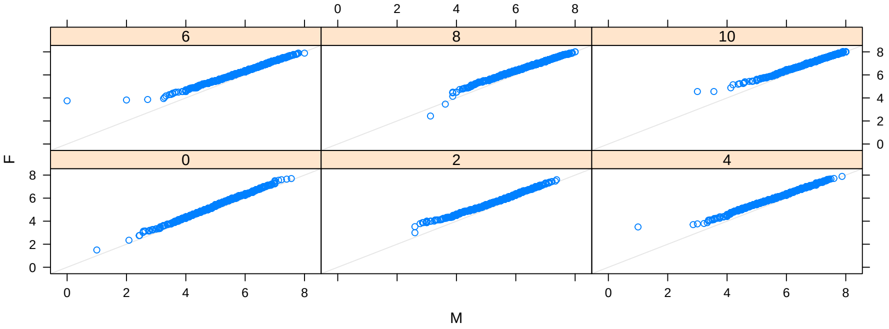
A densityplot

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



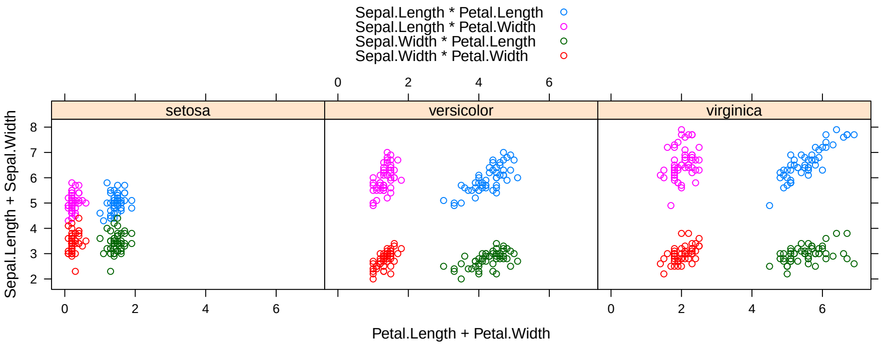
Bivariate Plots - Quantile-Quantile Plot

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



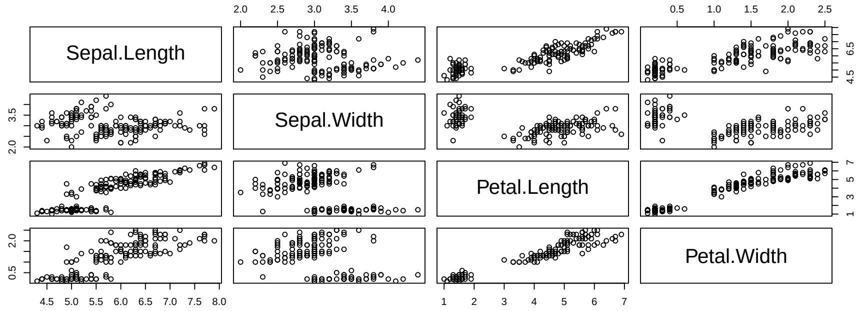
Scatterplot with `lattice` - `xyplot`

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



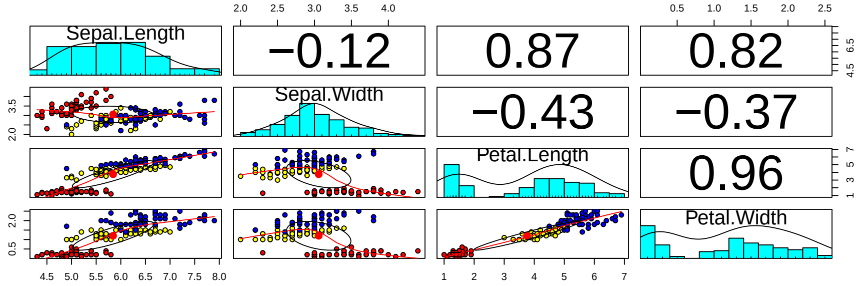
Relationship between variables - `pairs` plot

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



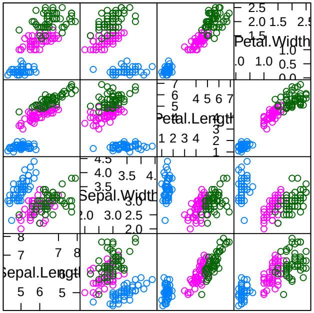
Enhanced multivariate plots

```r
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

```
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",
```

The dataset BankWages

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

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

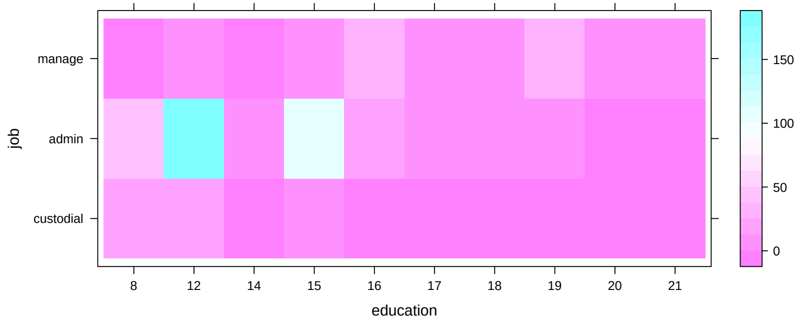
```r
head(BankWages)
```

```
##       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 years

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



Social network usage: Facebook (bbzc041a)

- 1 - No, I am no member; 2 - Yes, but never using it; 3 - Yes, use it sometimes; 4 - Yes, use it a lot

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

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

Social network usage: Twitter (bbzc042a)

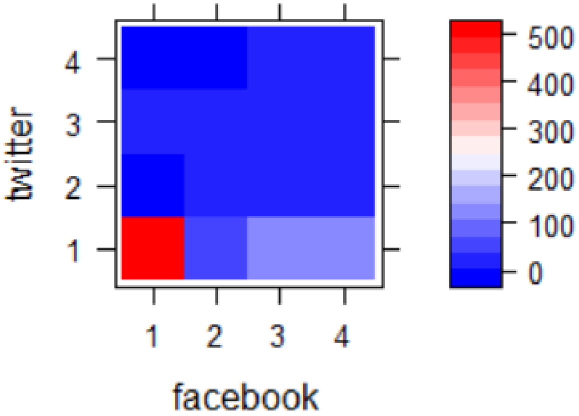
- 1 - No, I am no member; 2 - Yes, but never using it; 3 - Yes, use it sometimes; 4 - Yes, use it a lot

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

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

levelplot with GESIS Panel data

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



Internet use (GESIS Panel)

- a11c035a: Frequency private Internet usage: PC

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

- a11c037a: Frequency private Internet usage: smart phone

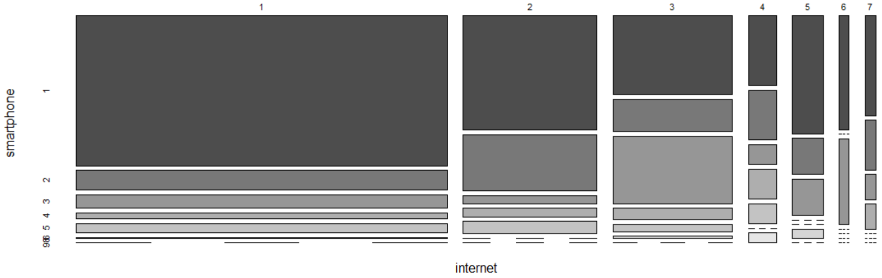
```
smartphone <- transform_miss(datf$a11c037a)
```

1 - Several times a day; 2 - About twice a day; 3 - More than once a week; 4 - About once a week; 5 - Rarer; 6 - Never; 98 - Don't know

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

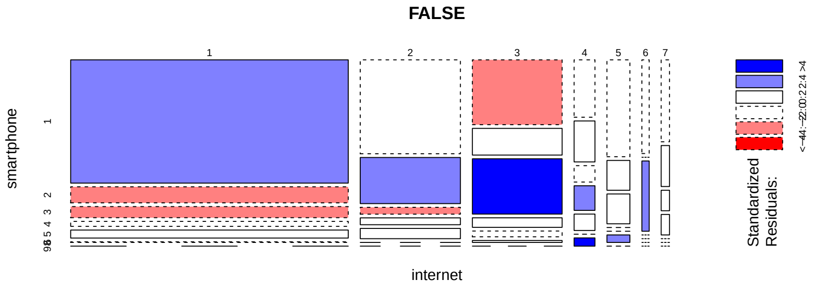
Relationship - categorial variables

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



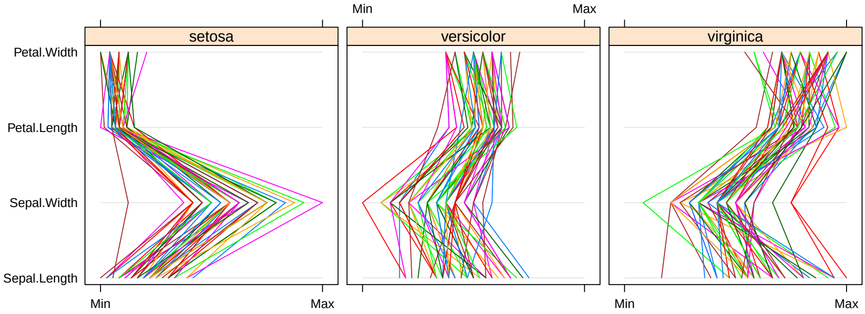
Surfaces are shaded according to the residuals:

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



parallelplot()

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

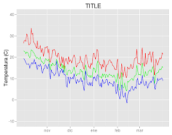


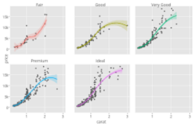
The ggplot2 package

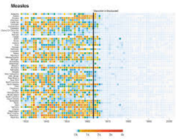
Introduction ggplot2

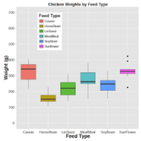
The ggplot2 package, created by Hadley Wickham, offers a powerful graphics language for creating elegant and complex plots. Its popularity in the R community has exploded in recent years. Origianlly based on Leland Wilkinson's The Grammar of Graphics, ggplot2 allows you to create graphs that represent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

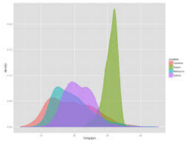
Examples ggplot2 graphics





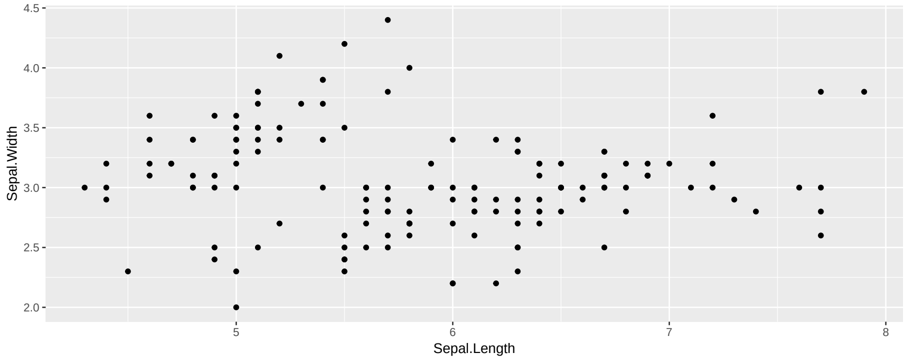






A first example ggplot2

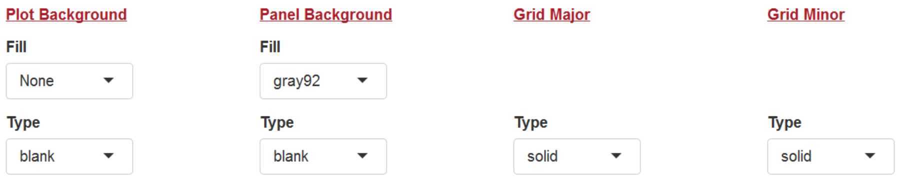
```r
library(ggplot2)
ggplot(iris, aes(x=Sepal.Length, y=Sepal.Width)) +
  geom_point()
```



Some nice Rstudio Addins

- A `ggplot` graphic has to be marked in source code, to use the following addins

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

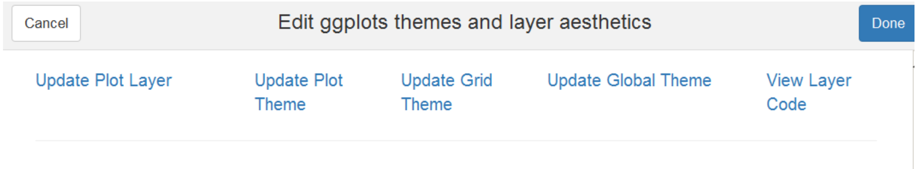

Plot Background
Fill
None
Type
blank

Panel Background
Fill
gray92
Type
blank

Grid Major
Type
solid

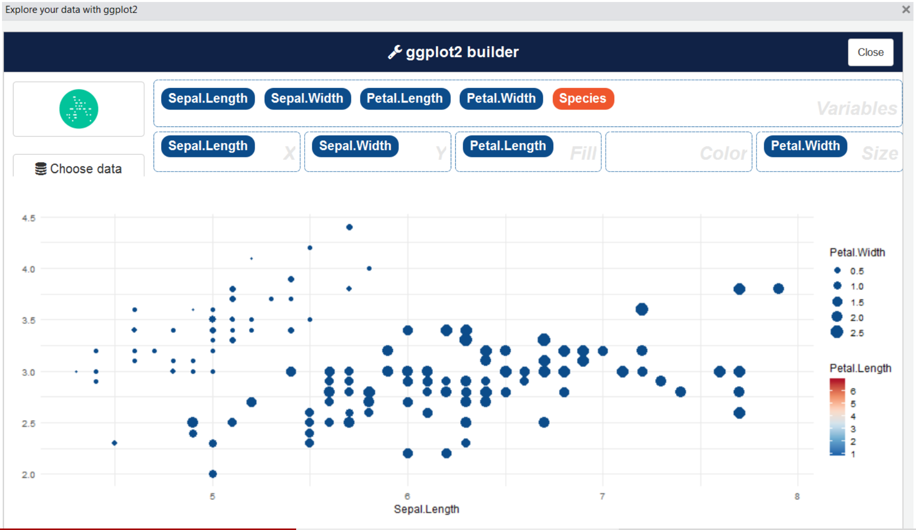
Grid Minor
Type
solid

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



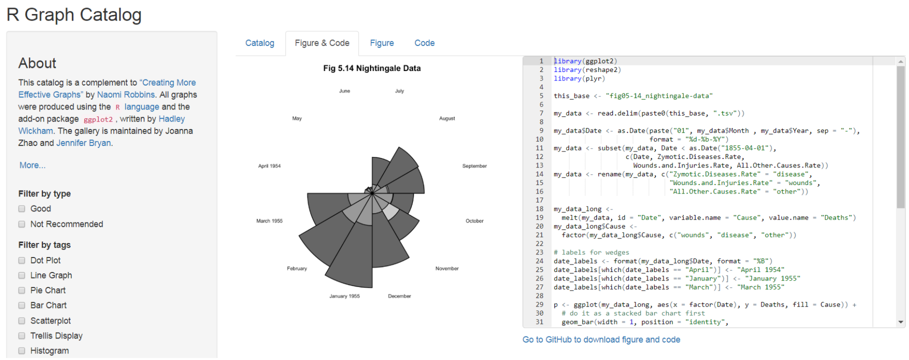
ggplot2 builder addin for RStudio

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



Shiny App - R graphs catalogue

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

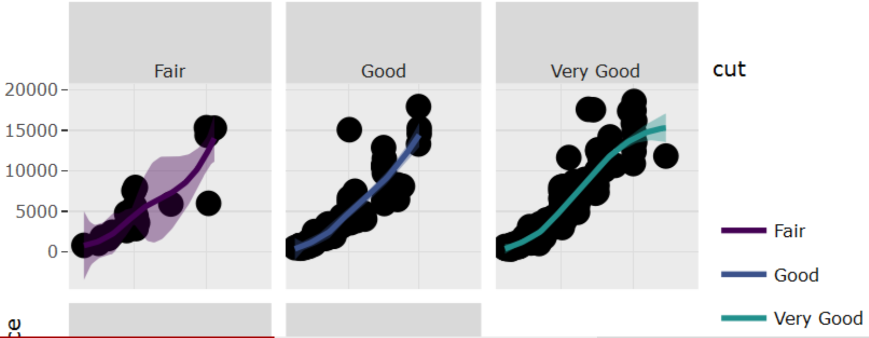
- ☐ Dot Plot
- ☐ Line Graph
- ☐ Pie Chart
- ☐ Bar Chart
- ☐ Scatterplot
- ☐ Trellis Display
- ☐ Histogram

```r
library(ggplot2)
library(reshape2)
library(plyr)

this_base <- "fig05-14_nightingale-data"

my_data <- read.delim(paste0(this_base, ".tsv"))

my_data$Date <- as.Date(paste("01", my_data$Month , my_data$Year, sep = "-"),
                        format = "%d-%b-%Y")
my_data <- subset(my_data, Date < as.Date("1855-04-01"),
                  c(Date, Zymotic.Diseases.Rate,
                    Wounds.and.Injuries.Rate, All.Other.Causes.Rate))
my_data <- rename(my_data, c("Zymotic.Diseases.Rate" = "disease",
                             "Wounds.and.Injuries.Rate" = "wounds",
                             "All.Other.Causes.Rate" = "other"))

my_data_long <-
  melt(my_data, id = "Date", variable.name = "Cause", value.name = "Deaths")
my_data_long$Cause <-
  factor(my_data_long$Cause, c("wounds", "disease", "other"))

# labels for wedges
date_labels <- format(my_data_long$Date, format = "%B")
date_labels[which(date_labels == "April")] <- "April 1954"
date_labels[which(date_labels == "January")] <- "January 1955"
date_labels[which(date_labels == "March")] <- "March 1955"

p <- ggplot(my_data_long, aes(x = factor(Date), y = Deaths, fill = Cause)) +
  # do it as a stacked bar chart first
  geom_bar(width = 1, position = "identity",
```

Go to GitHub to download figure and code

Add some interactivity

```
library(plotly)
d <- diamonds[sample(nrow(diamonds), 1000), ]
p <- ggplot(data = d, aes(x = carat, y = price)) +
  geom_point(aes(text = paste("Clarity:", clarity)), size = 4)
  geom_smooth(aes(colour = cut, fill = cut)) + facet_wrap(~ cu
(gg <- ggplotly(p))
```



Links

- 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**