A1 Getting started with R

Jan-Philipp Kolb

03 Februar, 2020

COURSE OBJECTIVES

- Perform your data analysis in a literate programming environment
- Import and manage structured and unstructured data
- Manipulate, transform, and summarize your data
- Join disparate data sources
- Methodically explore and visualize your data
- Perform iterative functions
- Write your own functions

. . . all with R!

Introduction round

Please tell us shortly...

- Where are you from? What are you studying/working?
- What is your experience level in R/other programming languages?
- What are your expectations of this course?
- Where do you think you can use R in the future?

Preliminaries

- Usually we have big differences in knowledge and abilities of the participants - please tell, if it is too fast or slow.
- We have lots of hands-on coding **exercises** - later you can only learn on your own
- We have many **examples** - try them
- If there are questions - always ask
- R is more fun together - ask your neighbor - strong proponent of collaborative work!

Sources of this course

Sources for figures, text, exercises etc:

- If the source is a website, the links are often in the header or in bold somewhere on the slide.
- At the end of a chapter, we often have additional links to read on.
- Please ask us, if something is unclear.

Reasons for using R...

- ... because it is an **open source language**
- ... outstanding graphs - **graphics**, **graphics**, **graphics**
- ... relates to other languages - **R can be used in combination with other programs** - e.g. **data linking**
- ...R can be used **for automation**
- ... Vast Community - **you can use the intelligence of other people ;-)**
- ...

Advantages of R

- R can be downloaded for **free**.

[Home]

Download

CRAN

The R Project for Statistical Computing

Getting Started

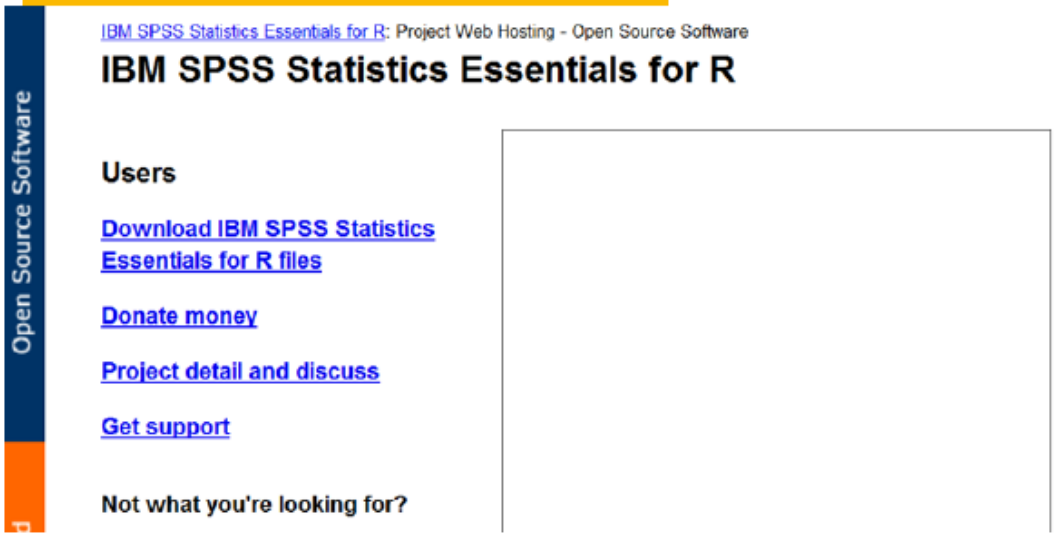
R is a free software environment for statistical computing and graphics. It compiles and runs on a wide variety of UNIX platforms, Windows and MacOS. To **download R**, please choose your preferred CRAN mirror.

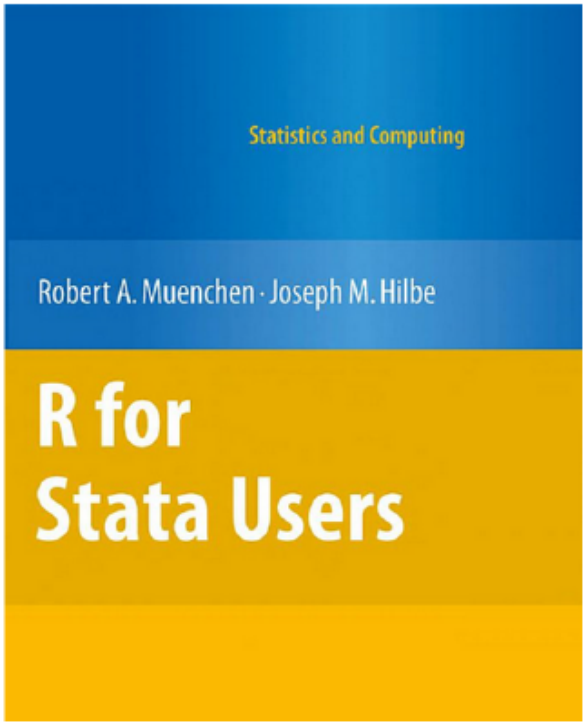
- R is a **scripting language**
- R is becoming more **popular**
- **Good** possibilities for **visualization**

R can be used in combination...

SASmixed

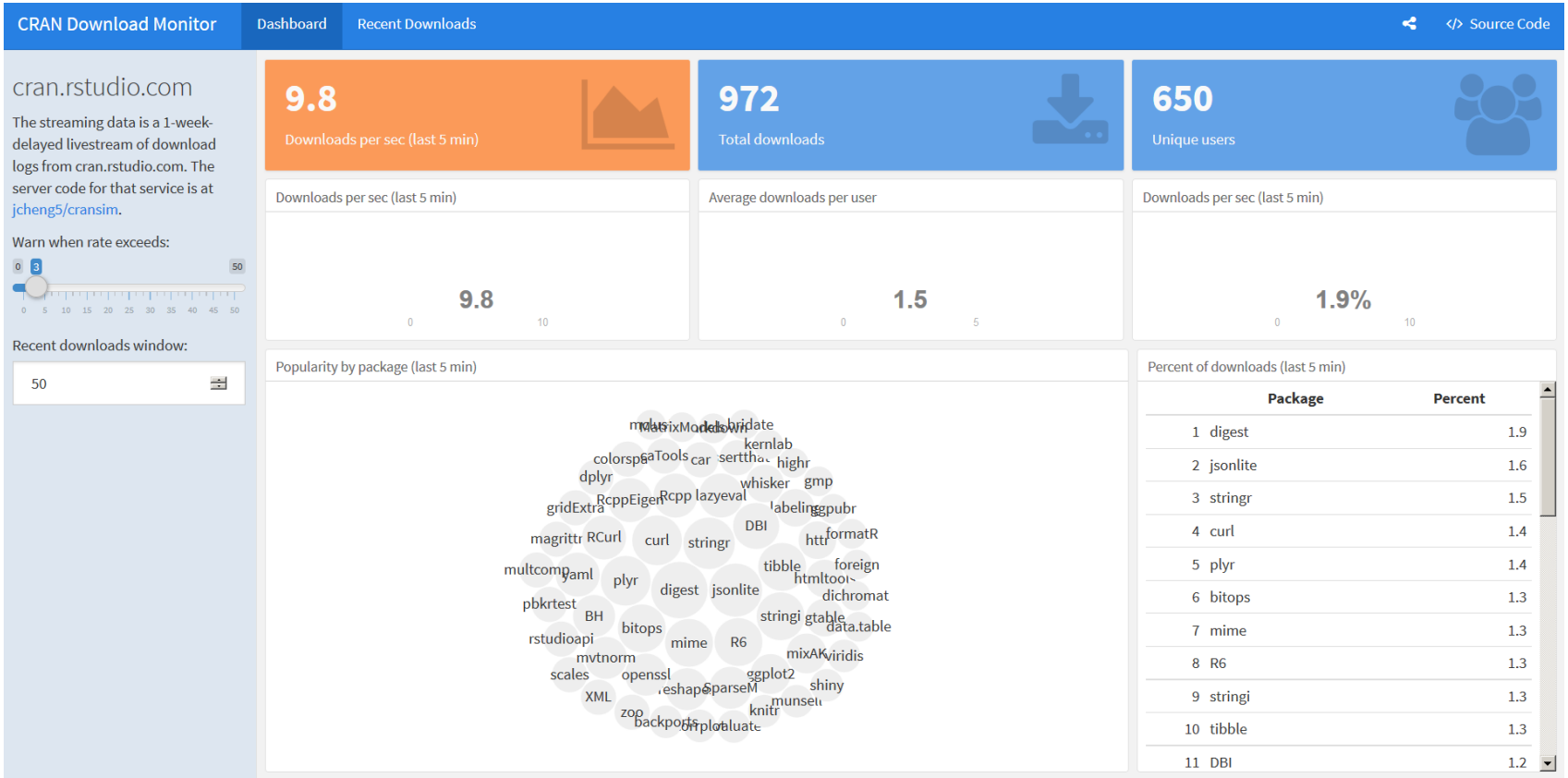
rPython R package





- Interface to: **Python**, **Excel**, **SPSS**, **SAS**, **Stata**

The popularity of R-packages



Download R:

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

CRAN
Mirrors
What's new?
Task Views
Search

About R
R Homepage
The R Journal

Software
R Sources
R Binaries
Packages
Other

The Comprehensive R Archive Network

Download and Install R

Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- Download R for Linux
- Download R for (Mac) OS X
- Download R for Windows

R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

Source Code for all Platforms

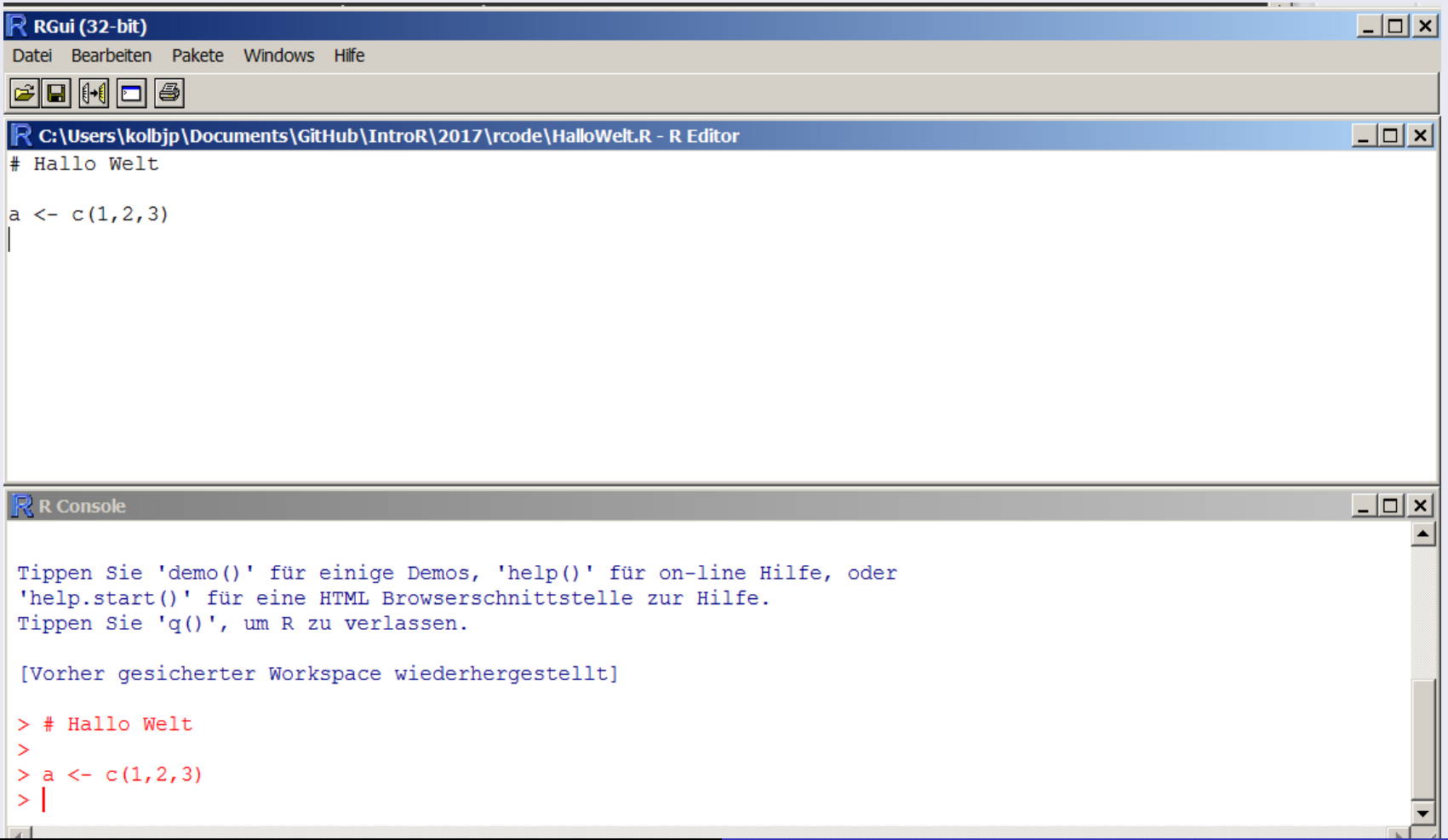
Windows and Mac users most likely want to download the precompiled binaries listed in the upper box, not the source code. The sources have to be compiled before you can use them. If you do not know what this means, you probably do not want to do it!

- The latest release (Friday 2017-04-21, You Stupid Darkness) R-3.4.0.tar.gz, read what's new in the latest version.

Open Source Programm R

- R is a free, non-commercial implementation of the S programming language (by AT&T Bell Laboratories)
- Free participation - modular structure

This is base R:



Graphical user interface

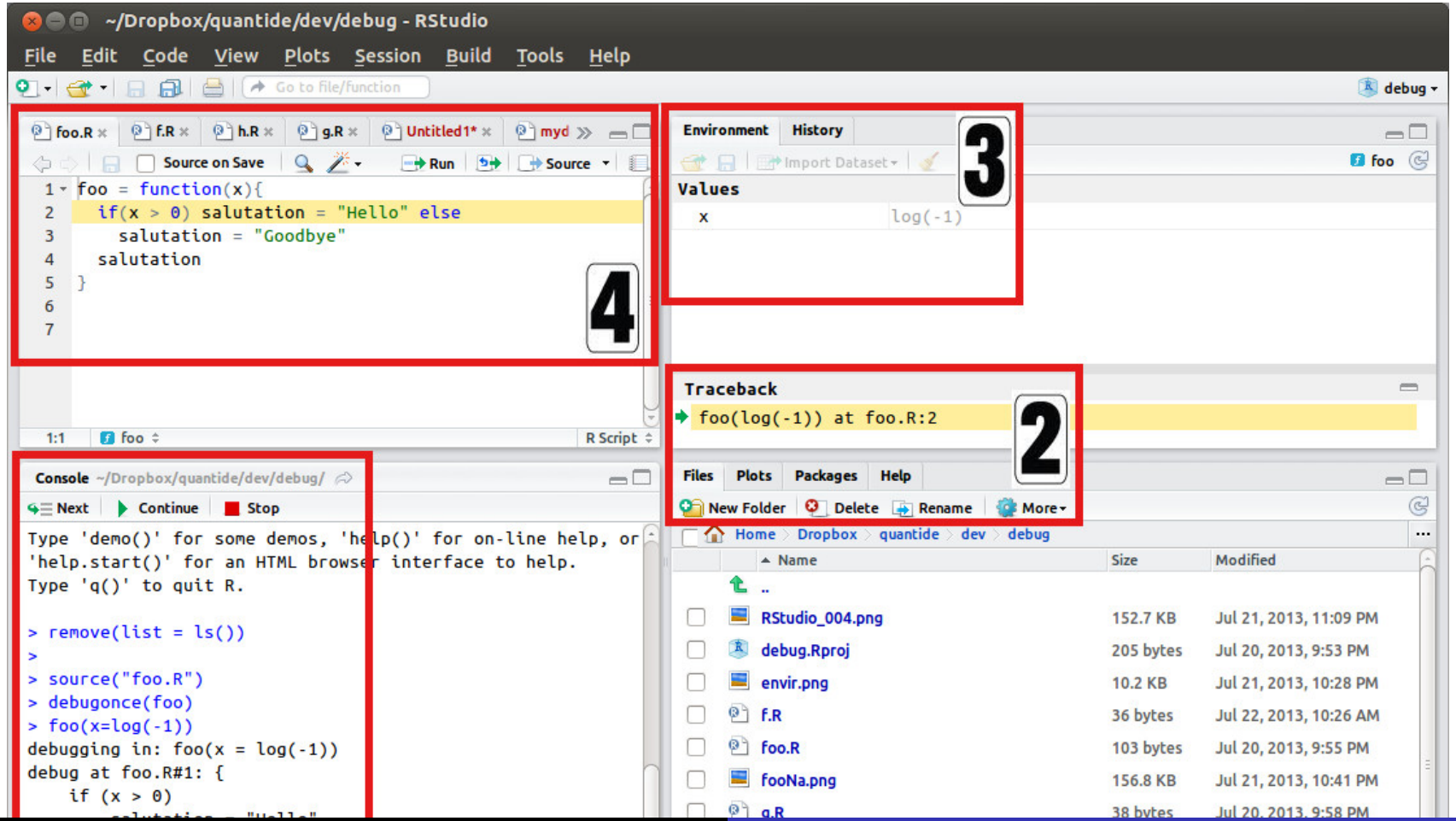
But many people use a graphical user interface (GUI) or a integrated development interface (IDE).

For the following reasons:

- Syntax highlighting
- Auto-completion
- Better overview on graphics, libraries, files, ...

Various text editors / IDEs

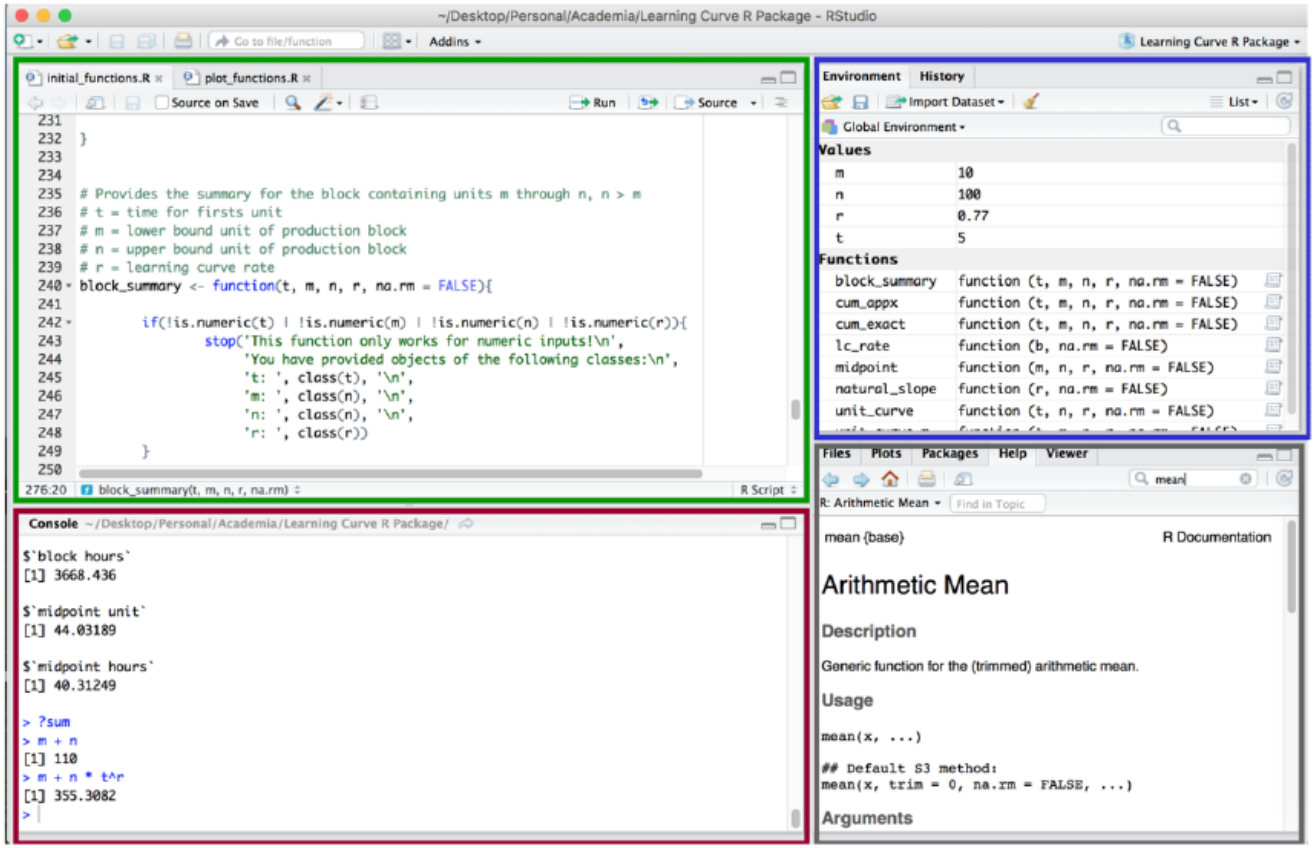
- **Gedit** with R-specific Add-ons for Linux
- **Emacs** and ESS (Emacs speaks statistics)- An extensible, customizable, free/libre text editor — and more.
- I use **Rstudio!**



RStudio

Script files
- Saves your script
- Allows code & comments
- Can have multiple files open at a time

Console/Command line
- Can use as calculator
- Does not save code
- This is where your output is displayed



Workspace environmen
- Holds your objects
- Can review history

Misc - Displays:
- files in working directory
- plots when produced
- help files/search

Important Rstudio Buttons

 create a new script

 open an existing script

run line where cursor is

R as a calculator

```
3 + 2 / 10^2 # Uses PEMDAS convention (order of operations)
```

```
## [1] 3.02
```

```
3 + (2 / 10^2)
```

```
## [1] 3.02
```

```
(3 + 2) / 10^2
```

```
## [1] 0.05
```

```
1 /19^4 # scientific notation is used for large numbers
```

```
## [1] 7.67336e-06
```

```
1/0 # Undefined calculations
```

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

```
Inf - Inf
```

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

Exercise: Preparation

- Check if R is installed on your computer.
- If not, download **R** and install it.
- Check if Rstudio is installed.
- If not - **install** Rstudio.
- Start RStudio. Go to the console (lower left window) and write

```
3+2
```

- If there is not already an editor open in the upper left window, then go to the file menu and open a new script. Check the date with `date()` and the R version with `sessionInfo()`.

```
date()
```

```
sessionInfo()
```

Exercise: See where things happen

- Create a new `.R` script named `my_first_script.R`
- Write and execute the following code in the `.R` script and identify where in Rstudio the outputs can be found.

```
mtcars
?sum
hist(mtcars$mpg)
random_numbers <- runif(40)
history()
```

R is a object-orientiented language

Vectors and assignments

- R is a object-orientiented language
- <- is the assignment operator

```
b <- c(1,2) # create an object with the numbers 1 and 2
```

- A function can be applied to this object:

```
mean(b) # computes the mean
```

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

We can learn something about the properties of the object:

```
length(b) # b has the length 2
```

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

```
sqrt(b) # the square root of b
```

```
## [1] 1.000000 1.414214
```

Functions in base-package

Function	Meaning	Example
str()	Object structure	str(b)
max()	Maximum	max(b)
min()	Minimum	min(b)
sd()	Standard deviation	sd(b)
var()	Variance	var(b)
mean()	Mean	mean(b)
median()	Median	median(b)

These functions only need one argument.

Functions with more arguments

Other functions need more arguments:

Argument	Meaning	Example
quantile()	90 % Quantile	quantile(b,.9)
sample()	Draw a sample	sample(b,1)

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

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

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

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

Examples - Functions with more than one argument

```
max(b); min(b)
```

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

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

```
sd(b); var(b)
```

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

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

Functions with one argument

```
mean(b)
```

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

```
median(b)
```

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

Exercise: Assignments and functions

Create a vector `b` with the numbers from 1 to 5 and calculate ...

1. the mean
2. the variance
3. the standard deviation
4. the square root from the mean

Overview commands

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

An Introduction to R

Table of Contents

Exercise: Economic Order Quantity Model

Economic Order Quantity Model

$$Q = \sqrt{\frac{2DK}{h}}$$

Calculate Q where:

- D = 1000
- K = 5
- h = 0.25

R Data Types

- R supports a few basic data types: integer, numeric, logical, character/string, factor, and complex

Logical

– binary, two possible values represented by `TRUE` and `FALSE`

```
x <- c(3,7, 1, 2)
x > 2
```

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

```
x == 2
```

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

```
!(x < 3)
```

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

```
which(x > 2)
```

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

Character vectors

```
y <- c("a","bc","def")
length(y)
```

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

```
nchar(y)
```

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

```
y == "a"
```

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

```
y == "b"
```

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

```
str(b) # b is a numeric vector
```

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

Variable type character

```
a <- letters
length(letters)
```

```
## [1] 26
```

```
a[1:4]
```

```
## [1] "a" "b" "c" "d"
```

```
str(a)
```

```
##  chr [1:26] "a" "b" "c" "d" "e" "f" "g" "h" "i" "j" "k" "l" "
```

Problems with character vector

```
mean(b)
```

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

```
(b1 <- c(b,"a"))
```

```
## [1] "1" "2" "a"
```

```
mean(b1)
```

```
## Warning in mean.default(b1): argument is not numeric or logic
```

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

Coercion

- All elements in a vector must be of the same type. R coerces the elements to a common type
- `c(1.2,3,TRUE)` – In this case all elements are coerced to numeric.

```
x <- c(TRUE,FALSE,TRUE)
c(1.2,x)
```

```
## [1] 1.2 1.0 0.0 1.0
```

```
y <- c("2","3",".2")
c(1.2,y, x)
```

```
## [1] "1.2"   "2"     "3"     ".2"    "TRUE"  "FALSE" "TRUE"
```

- Sometimes this coercion occurs to perform an arithmetic operation:

```
1 + x
```

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

Perform the coercion

- Other times we need to perform the coercion

```
c(1.2,y)
```

```
## [1] "1.2" "2"   "3"   ".2"
```

```
c(1.2,as.numeric(y))
```

```
## [1] 1.2 2.0 3.0 0.2
```

Information about Vectors

- Aggregator functions - `sum`, `mean`, `range`, `min`, `max`, `summary`, `table`, `cut`, `...`
- `class(x)` – returns the type of an object.
- `is.logical(x)` – tells us whether the object is a logical type. There is also `is.numeric`, `is.character`, `is.integer`
- `is.null` – determines whether an object is empty, i.e. has no content. 'NULL' is used mainly to represent the lists with zero length, and is often returned by expressions and functions whose value is undefined.

Coerce objects from one to another

- `as.numeric(x)` – we use the as-type functions to coerce objects from one type (e.g. logical) to another, in this case numeric.
- There are several of these functions, including `as.integer`, `as.character`, `as.logical`

```
x <- c("1",2,"one","1plus","2_and")
as.numeric(x)
```

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

Data Frames

A data frame is a collection of vectors - different columns can have different modes (numeric, character, factor, etc.).

Three example vectors

```
d <- c(1,2,3,4)
e <- c("red", "white", "red", NA)
f <- c(TRUE,TRUE,TRUE,FALSE)
```

Bind the example vectors together:

```
mydata <- data.frame(d,e,f)
```

Give the columns some names

```
names(mydata) <- c("ID","Color","Passed") # variable names
```

Identify the elements of a data frame

There are a variety of ways to identify the elements of a data frame .

```
myframe[3:5] # columns 3,4,5 of data frame
myframe[c("ID","Age")] # columns ID and Age from data frame
myframe$X1 # variable x1 in the data frame
```

Matrices

All columns in a matrix must have the same mode (numeric, character, etc.) and the same length. The general format is:

```
# generates 5 x 4 numeric matrix
y <- matrix(1:20, nrow=5,ncol=4)
```

- byrow=TRUE indicates that the matrix should be filled by rows.
- byrow=FALSE - matrix should be filled by columns (the default).

```
# an example
cells <- c(1,26,24,68)
mymatrix <- matrix(cells, nrow=2, ncol=2, byrow=TRUE)
```

Matrix - dimnames

- `dimnames` provides optional labels for the columns and rows.

```
# another example
rnames <- c("R1", "R2")
cnames <- c("C1", "C2")
mymatrix <- matrix(cells, nrow=2, ncol=2, byrow=TRUE,
  dimnames=list(rnames, cnames))
```

Matrices and subscripts

Identify rows, columns or elements using subscripts.

```
x[,4] # 4th column of matrix
x[3,] # 3rd row of matrix
x[2:4,1:3] # rows 2,3,4 of columns 1,2,3
```

Lists

An ordered collection of objects (components). A list allows you to gather a variety of (possibly unrelated) objects under one name.

```
# example of a list with 4 components -
# a string, a numeric vector, a matrix, and a scaler
w <- list(name="Fred", mynumbers=a, mymatrix=y, age=5.3)

# example of a list containing two lists
v <- c(list1,list2)
```

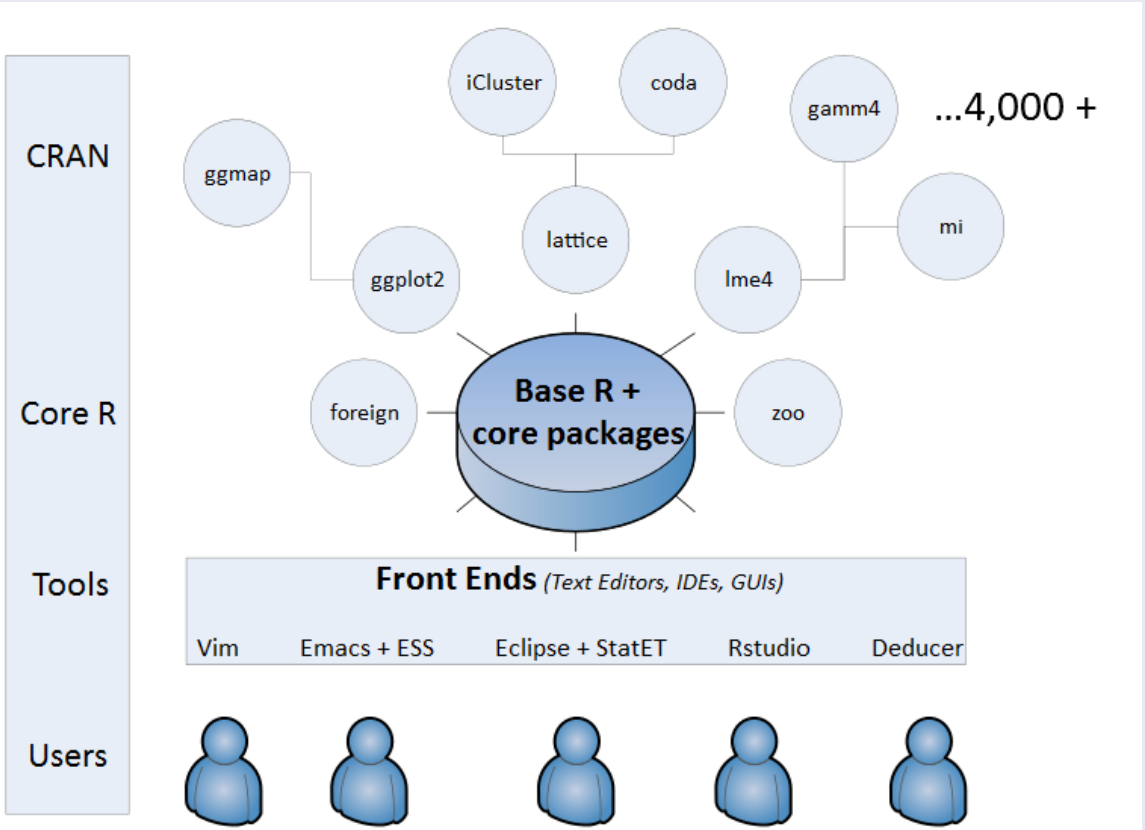
Identify elements of a list using the [[ ]] convention.

```
mylist[[2]] # 2nd component of the list
mylist[["mynumbers"]] # component named mynumbers in list
```

Where to find routines

- Many functions are included in basic R
- Many specific functions are integrated in additional libraries
- R can be modularly extended by so-called packages or libraries
- Most important packages hosted on CRAN (15320 at Mo Feb 03)
- Further packages can be found e.g. at **bioconductor**

Overview R packages



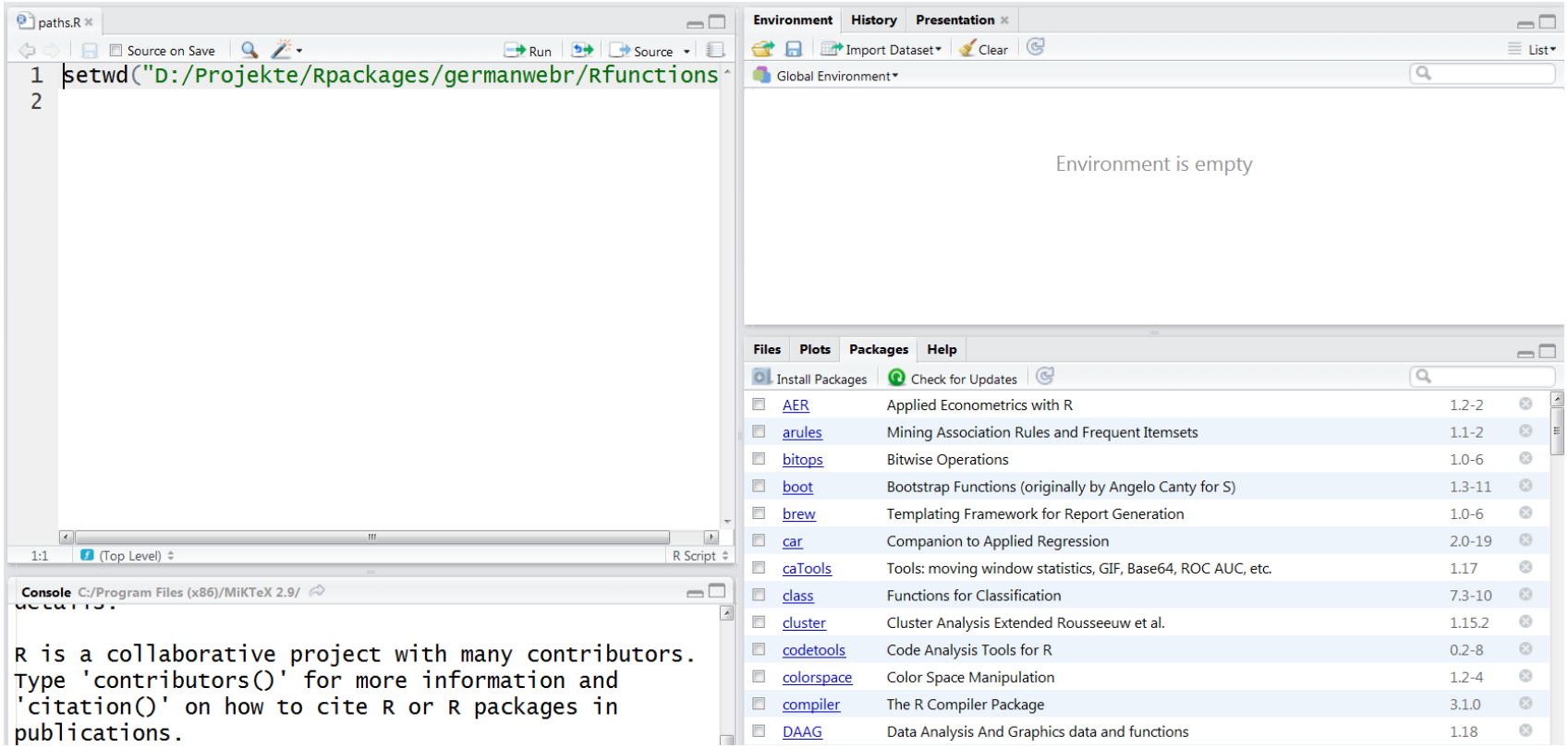
Installation of packages

- The quotes around the package name are necessary for the command `install.packages`.
- They are optional for the command `library`.
- You can also use `require` instead of `library`.

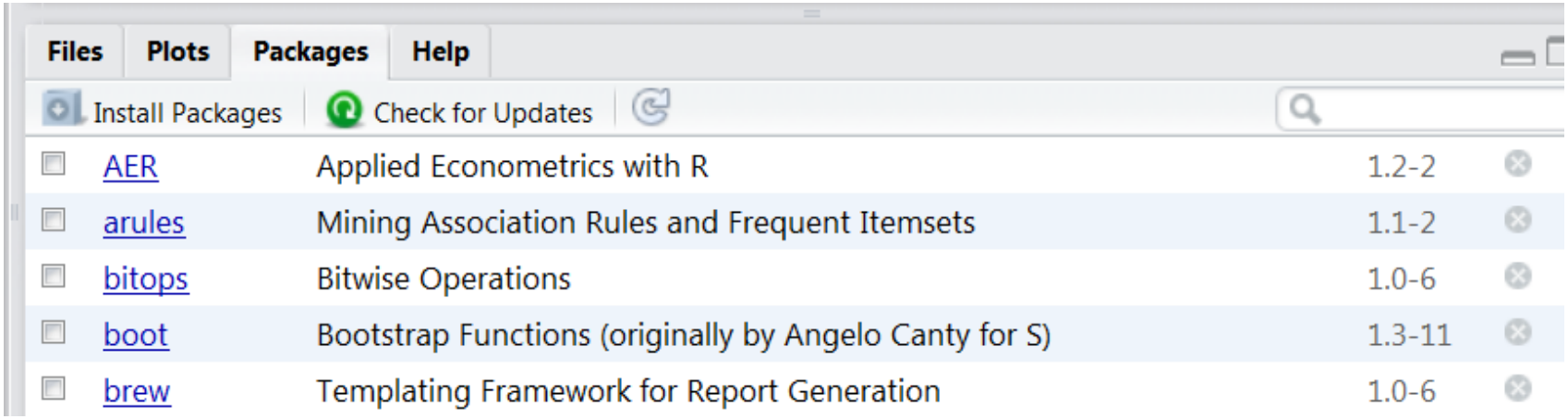
```
install.packages("lme4")

library(lme4)
```

Installation of packages with RStudio



Existing packages and installation



Exercise: Download packages

Download and install the following packages from CRAN:

- `tidyverse`
- `nycflights13`
- `cluster`
- `ggplot2`
- `tmap`

Have a look at the package documentation. What are these packages for?

Overview of many useful packages:

- Luhmann - **Table with many useful packages**

Other interesting packages:

- Package for Import/Export - `foreign`
- **`sampling`-package for survey Sampling**
- `xtable` Package for integrating LateX in R (**xtable Galerie**)
- **`dummies` package for creating dummies**
- **Package `mvtnorm` for getting a multivariate normal distribution**
- **Package `maptools` for creating maps**

Install packages from various sources

Install packages from CRAN Server

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

Install packages from Bioconductor Server

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

Install packages from Github

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

install_github("hadley/ggplot2")
```

Packages

```
# load the package to use in the current R session
library(tidyverse)

# use a particular function within a package
# without loading the package
stringr::str_replace()
```

Getting help on packages

```
# provides details regarding contents of a package
help(package = "tidyr")
# list vignettes available for a specific package
vignette(package="tidyr")
# view specific vignette
vignette("tidy-data")
```

How do I get an overview

- **Discover packages recently uploaded to CRAN**
- Look at the Shiny web app that shows the **packages recently downloaded from CRAN**
- Have a look at a **quick-list of useful packages**,. . .
- . . . , or at a list with the **best packages for data processing and analysis**,. . .
- . . . , or at **the 50 most used packages**

CRAN Task Views

- For some topics all possibilities are arranged in R. (**Overview of Task Views**)
- Currently there are 35 task views.
- All packages of a task view can be installed with the following **command:**

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

CRAN Task Views

Bayesian	Bayesian Inference
ChemPhys	Chemometrics and Computational Physics
ClinicalTrials	Clinical Trial Design, Monitoring, and Analysis
Cluster	Cluster Analysis & Finite Mixture Models
DifferentialEquations	Differential Equations
Distributions	Probability Distributions
Econometrics	Econometrics
Environmetrics	Analysis of Ecological and Environmental Data
ExperimentalDesign	Design of Experiments (DoE) & Analysis of Experimental Data
ExtremeValue	Extreme Value Analysis
Finance	Empirical Finance

Exercise: additional packages

Go for example to:

https://cran.r-project.org/
https://awesome-r.com/

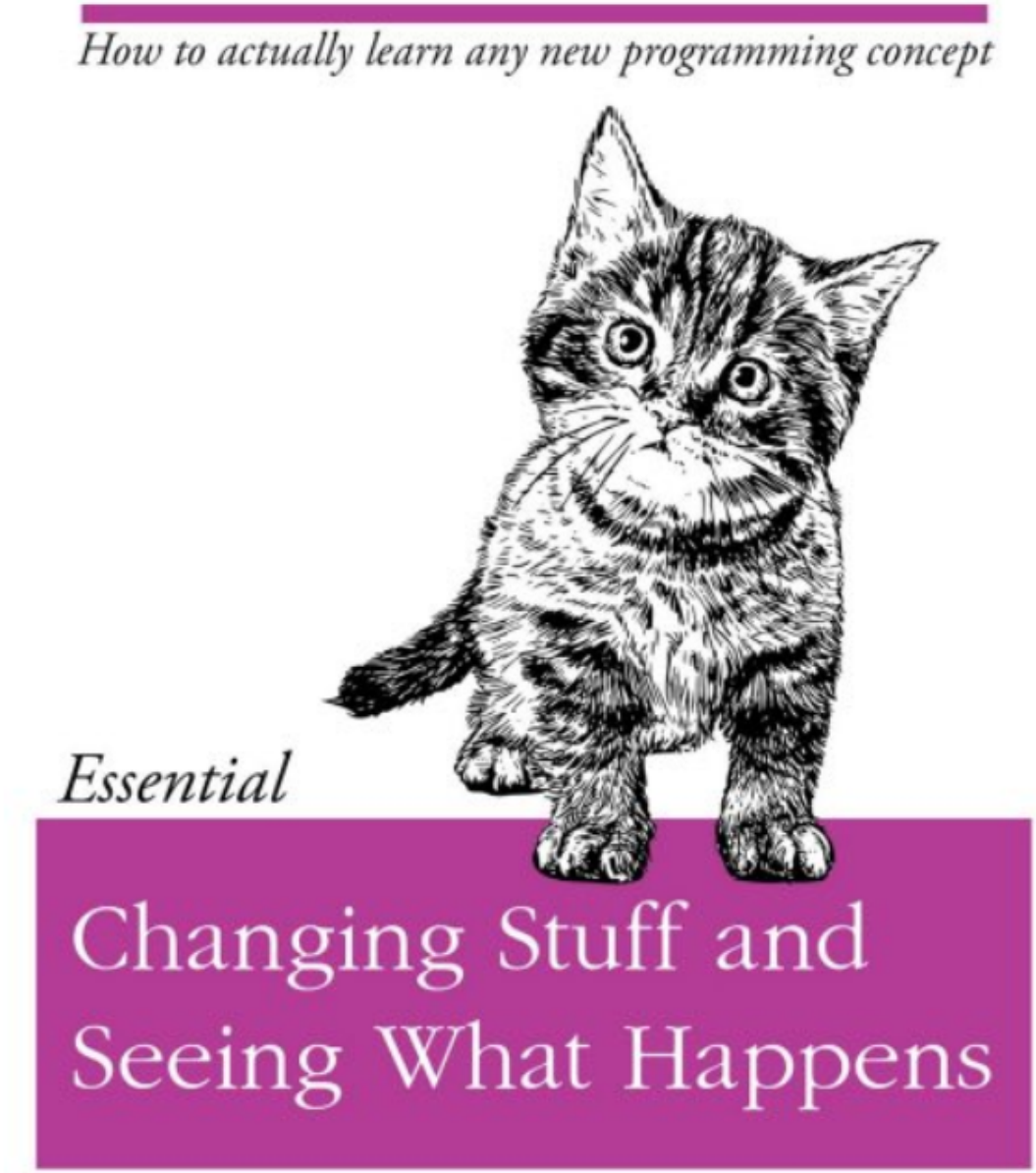
or search for

`most interesting r packages`

and search for packages ...

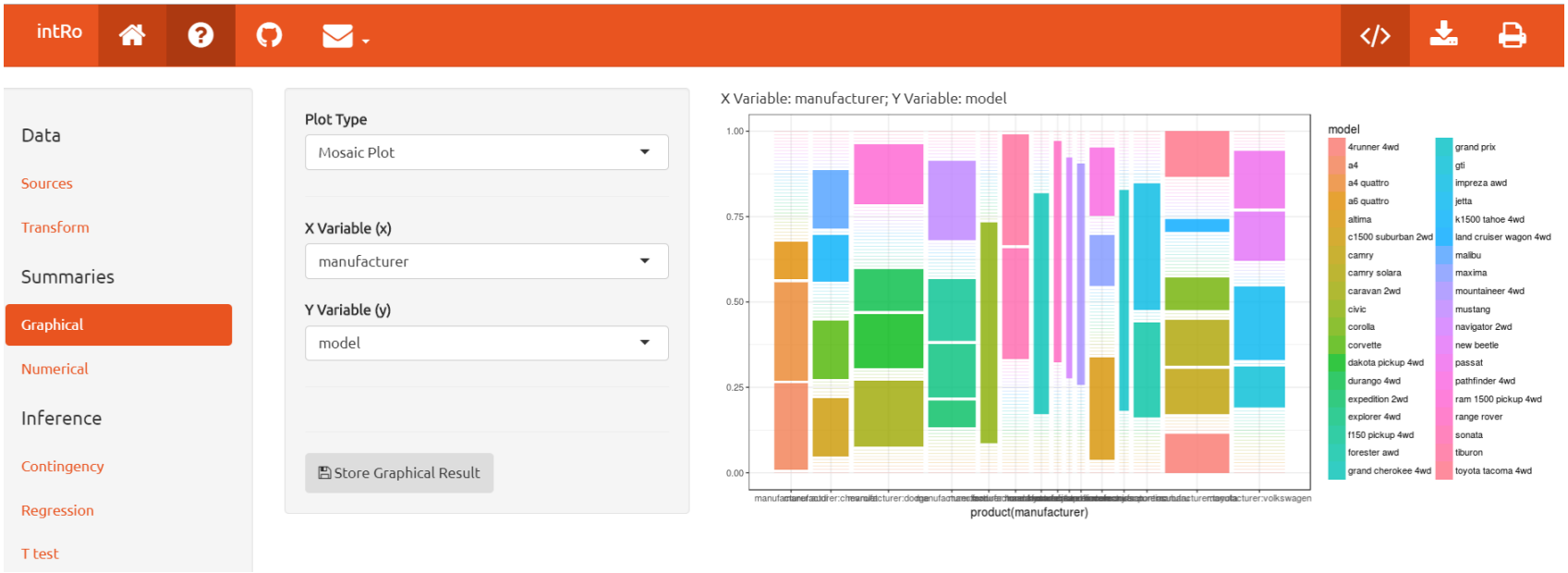
- for descriptive data analysis.
- with functions to work with date-times and time-spans.
- to use an interface to `python`.
- to import foreign data (e.g. SPSS data).
- to handle large amounts of data

How to learn after this workshop



Shiny App - Intro R

http://www.intro-stats.com/



Some links to read on

- Six **reasons** to use **Rstudio**.
- **Why you should learn R first for data science**
- **RStudio – Infoworld 2015 Technology of the Year Award Recipient!**
- **Why the R programming language is good for business?**
- **Have a look at R-bloggers**
- **Comparisson between python and R**
- **R and Stata Side-by-side**
- **AWESOME R**
- **1000 R tutorials/Links**
- **Learn R by watching two-minute videos**

R for stata users

- Oscar Torres-Reyna - **Exploring Data and Descriptive Statistics (using R)**