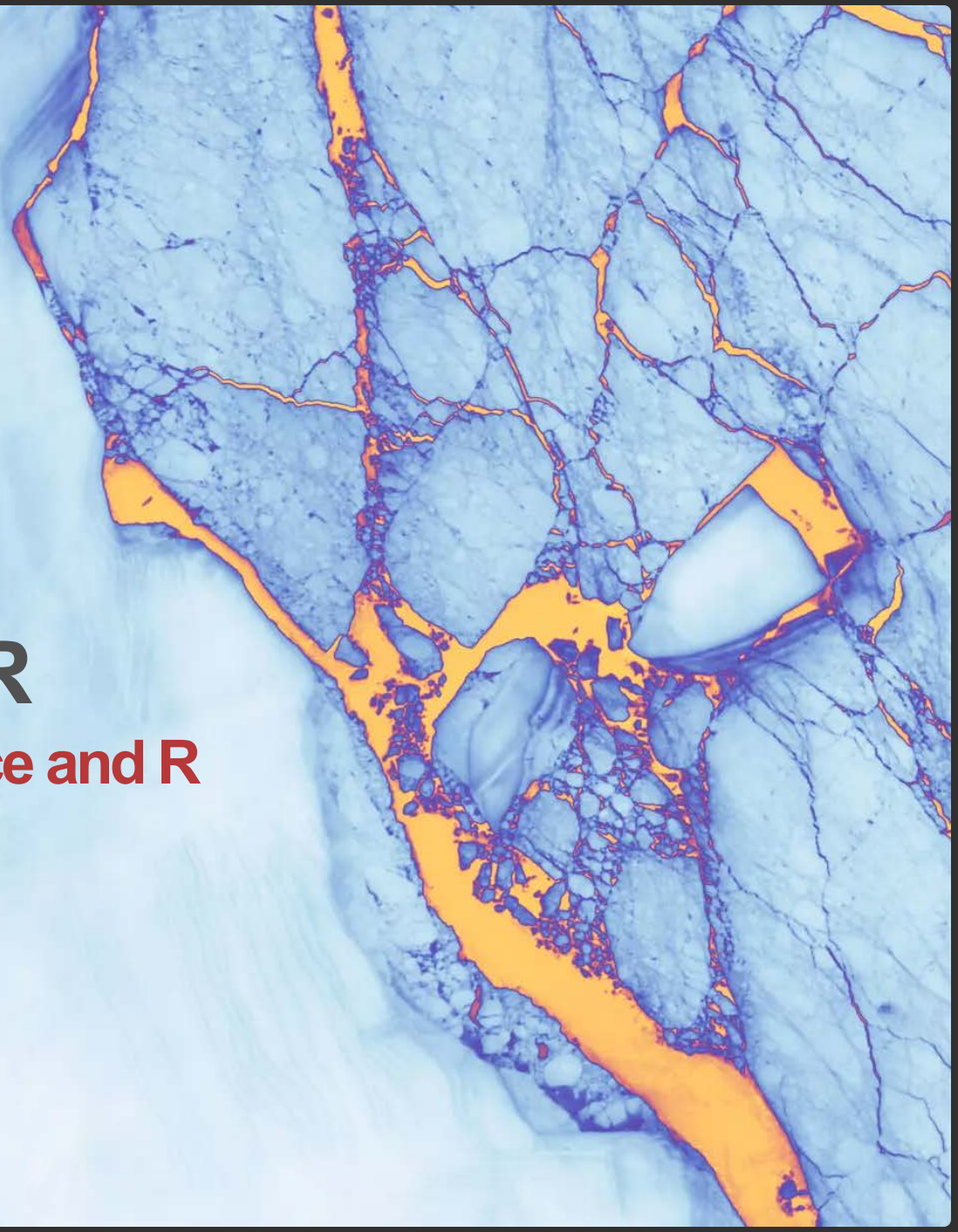
Marine Data Science

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Data Analysis with R

1 - Introduction to data science and R

Saskia A. Otto
Postdoctoral Researcher



What is 'Data Analysis' or 'Data Science'?

Data science is

- all about **uncovering findings from data**. Diving in at a granular level to mine and understand complex behaviors, trends, and inferences.

Data science is

- all about **uncovering findings from data**. Diving in at a granular level to mine and understand complex behaviors, trends, and inferences.

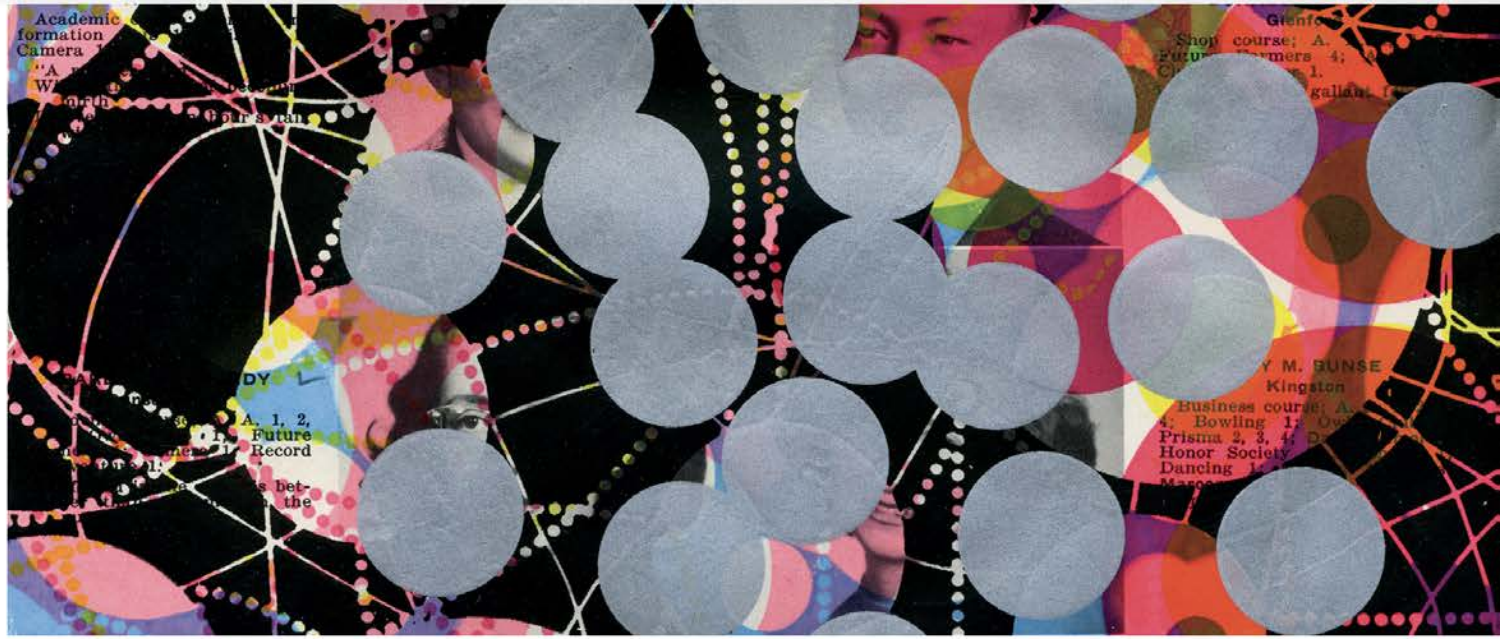
How do data scientists mine out insights?

- It starts with data exploration.
- When given a challenging question, data scientists become **detectives** and investigate leads and try to **understand patterns** within the data.
- Data scientists may **apply quantitative technique** in order to get a level deeper → e.g. inferential models, segmentation analysis, time series forecasting, synthetic control experiments, etc.
- The intent is to scientifically **piece together a forensic view** of what the data is really saying.
- THATS WHY IT IS SO EXCITING!

MENU

Subscribe | Sign In | Register

Harvard Business Review



ARTWORK: TAMAR COHEN, ANDREW J BUBOLTZ, 2011, SILK SCREEN ON A PAGE FROM A HIGH SCHOOL YEARBOOK, 8.5" X 12"

DATA

Data Scientist: The Sexiest Job of the 21st Century

by Thomas H. Davenport and D.J. Patil

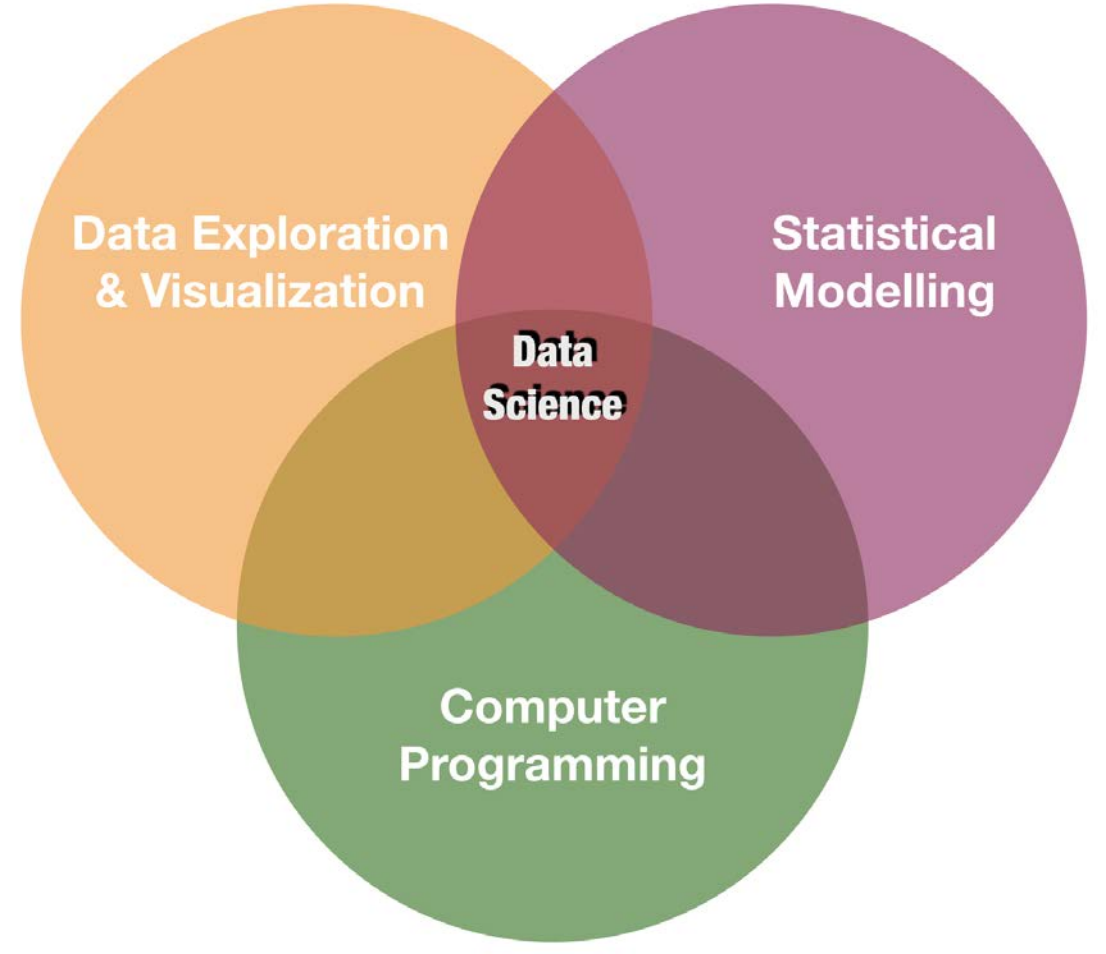
FROM THE OCTOBER 2012 ISSUE

WHAT TO READ NEXT

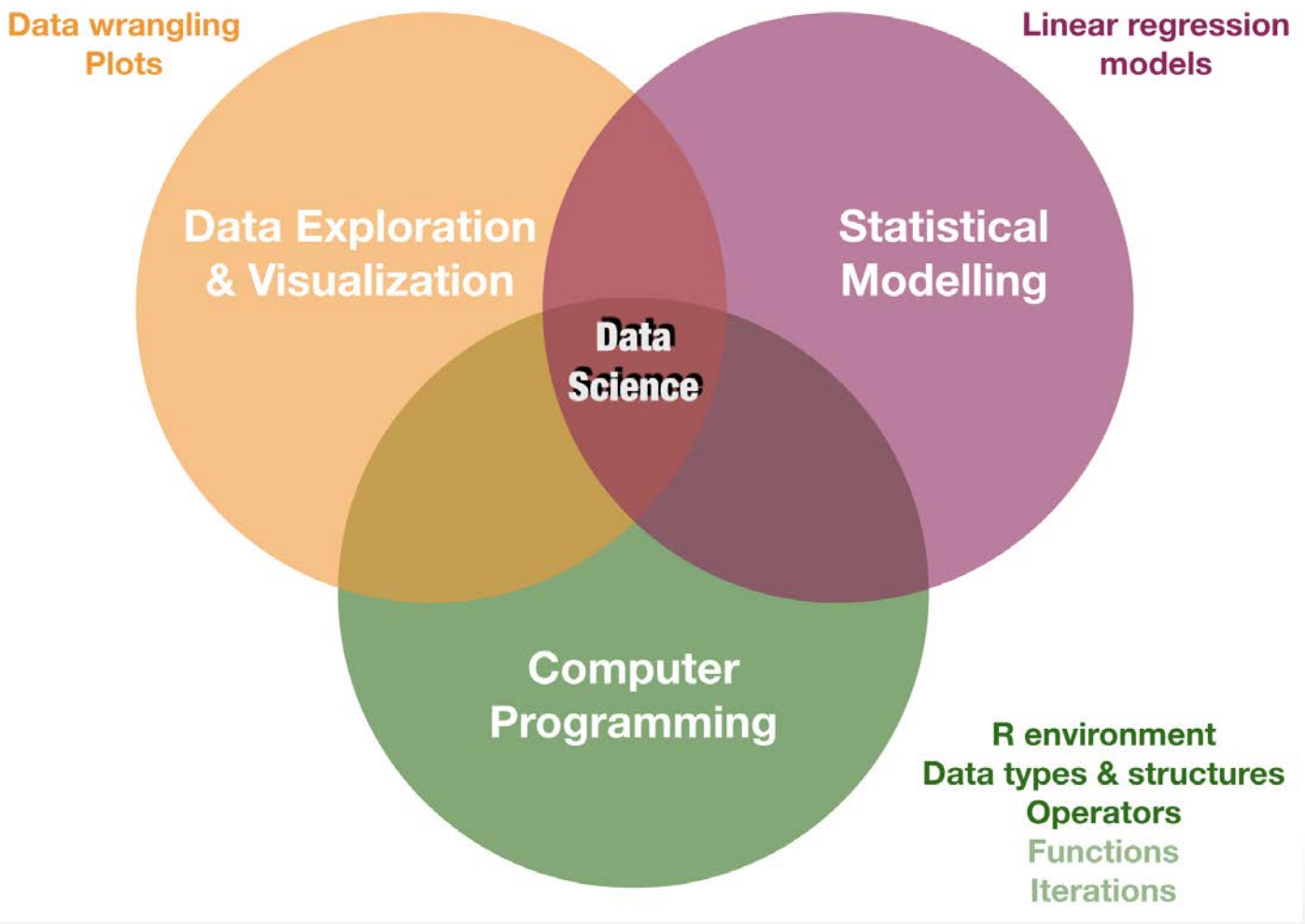
Big Data: The Management Revolution

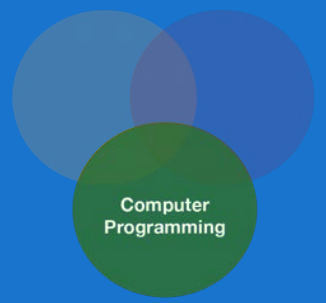
Screenshot of Harvard Business Review website, taken November 11, 2017.

Data science is a blend of skills in three major areas:



You will learn in this course:





Intro to the R environment

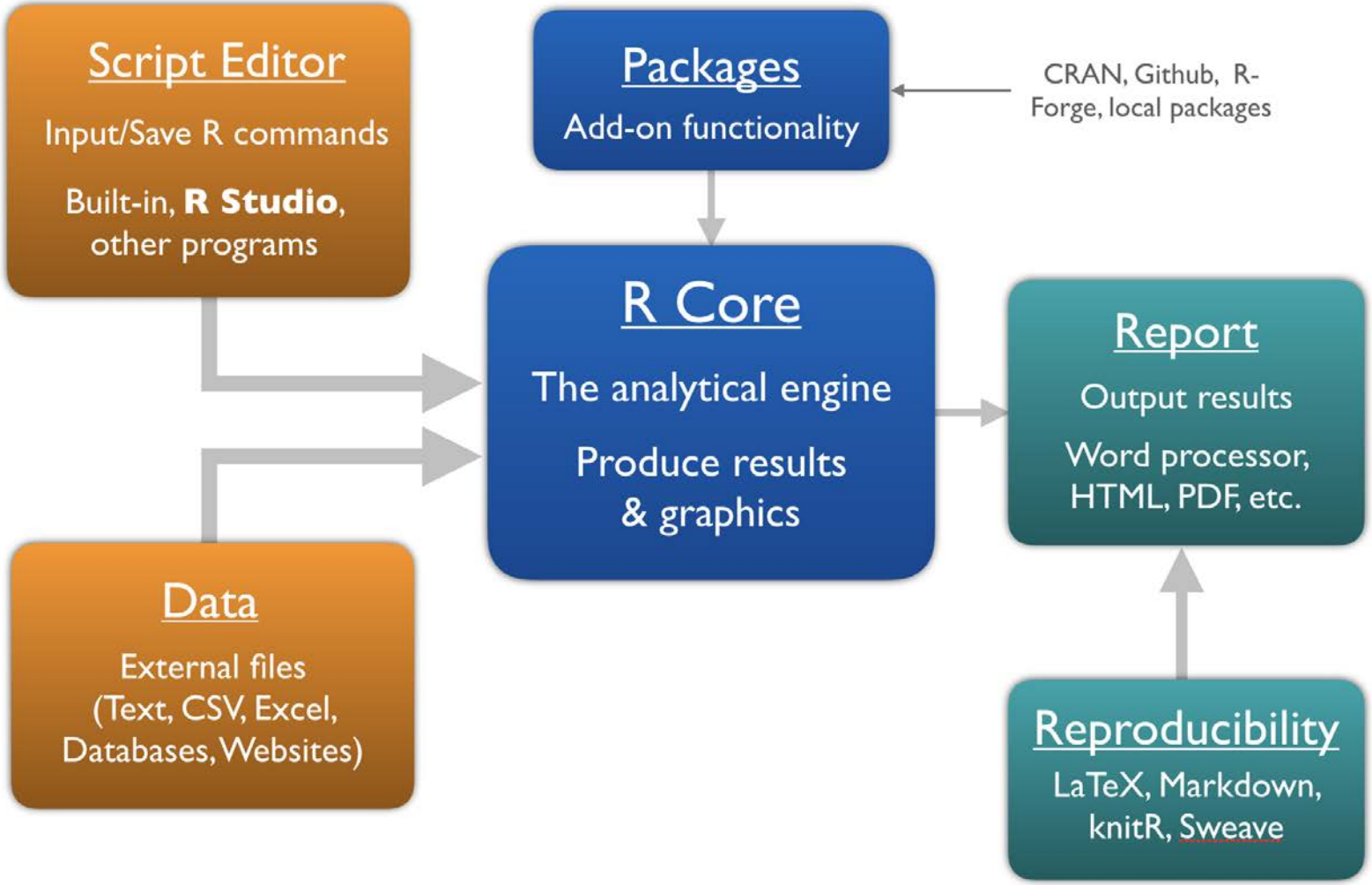
What is R?

R is a programming language

"R is a **system for statistical computation and graphics**. It is a GNU project which is similar to the S language and environment which was developed at Bell Laboratories by John Chambers and colleagues. R can be considered as a different implementation of S…..R is available as **Free Software** under the terms of the Free Software Foundation's GNU General Public License in source code form. It compiles and runs on a wide variety of **UNIX** platforms, **Windows** and **MacOS**." (from http://r-project.org/)

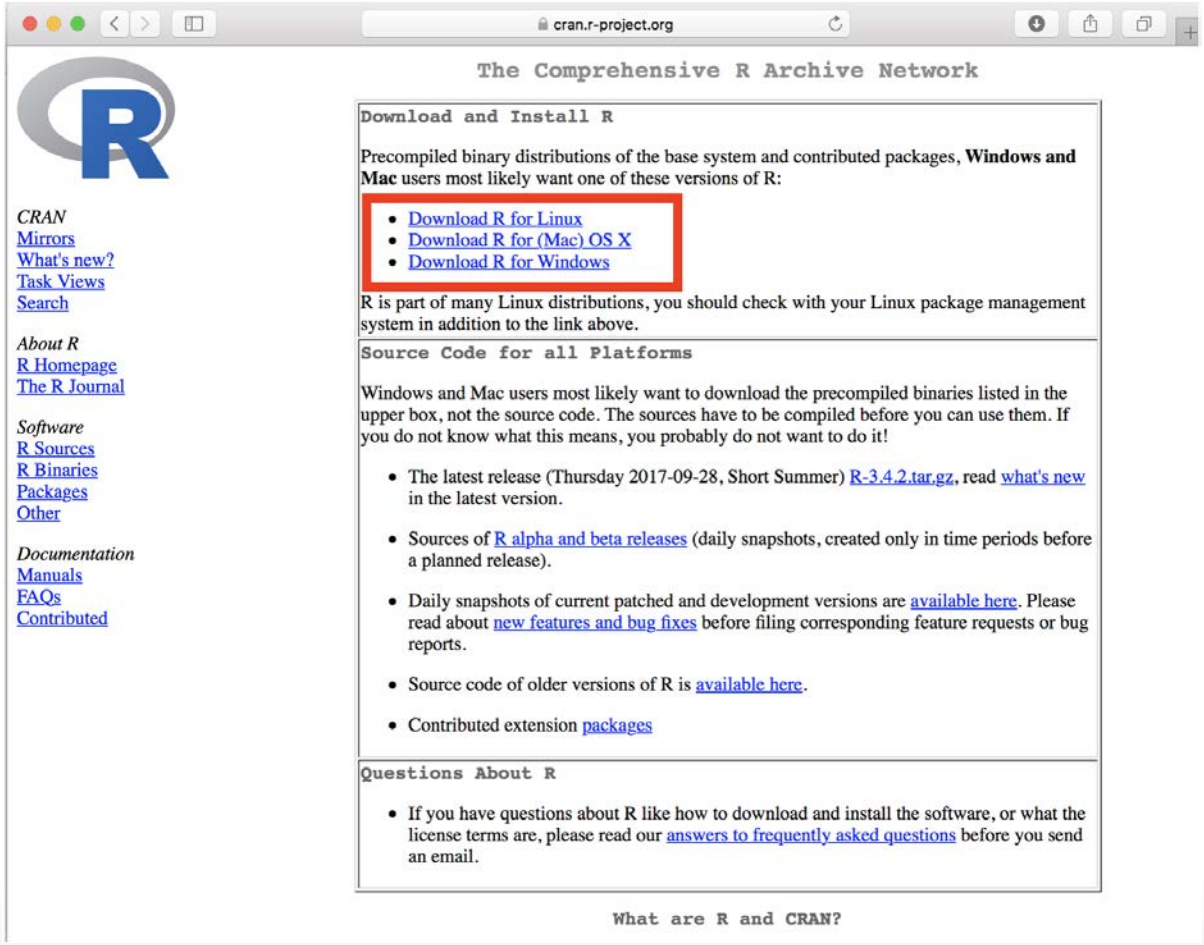
Why use R?


R The most powerful and most widely used statistical software
Robert Gentleman
Ross Ihaka
Später ansehen
Teilen


Script Editor
Input/Save R commands
Built-in, R Studio, other programs
Packages
Add-on functionality
CRAN, Github, R-Forge, local packages
R Core
The analytical engine
Produce results & graphics
Report
Output results
Word processor, HTML, PDF, etc.
Data
External files (Text, CSV, Excel, Databases, Websites)
Reproducibility
LaTeX, Markdown, knitR, Sweave

Where do you get R?

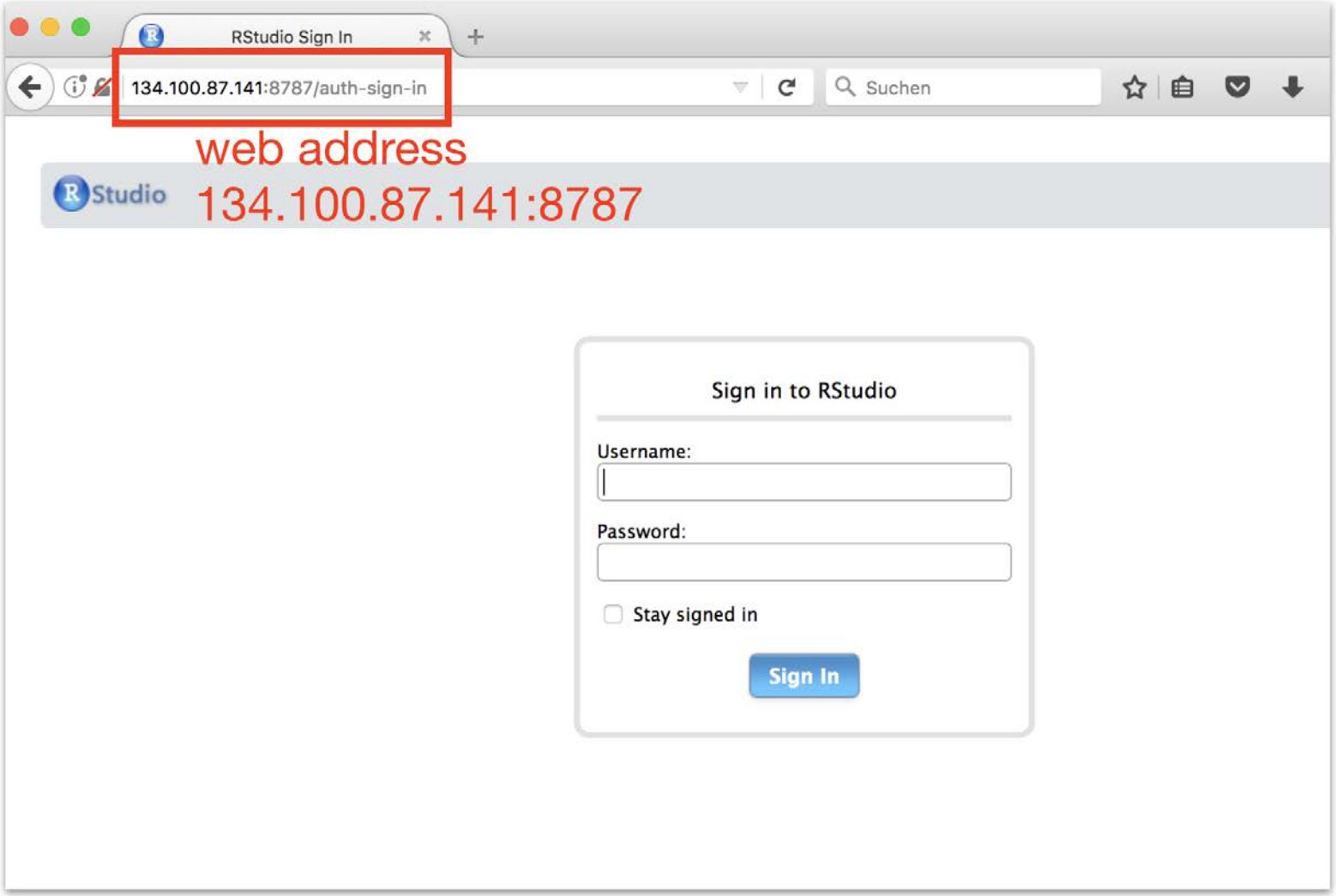
Directly from the website https://cran.r-project.org

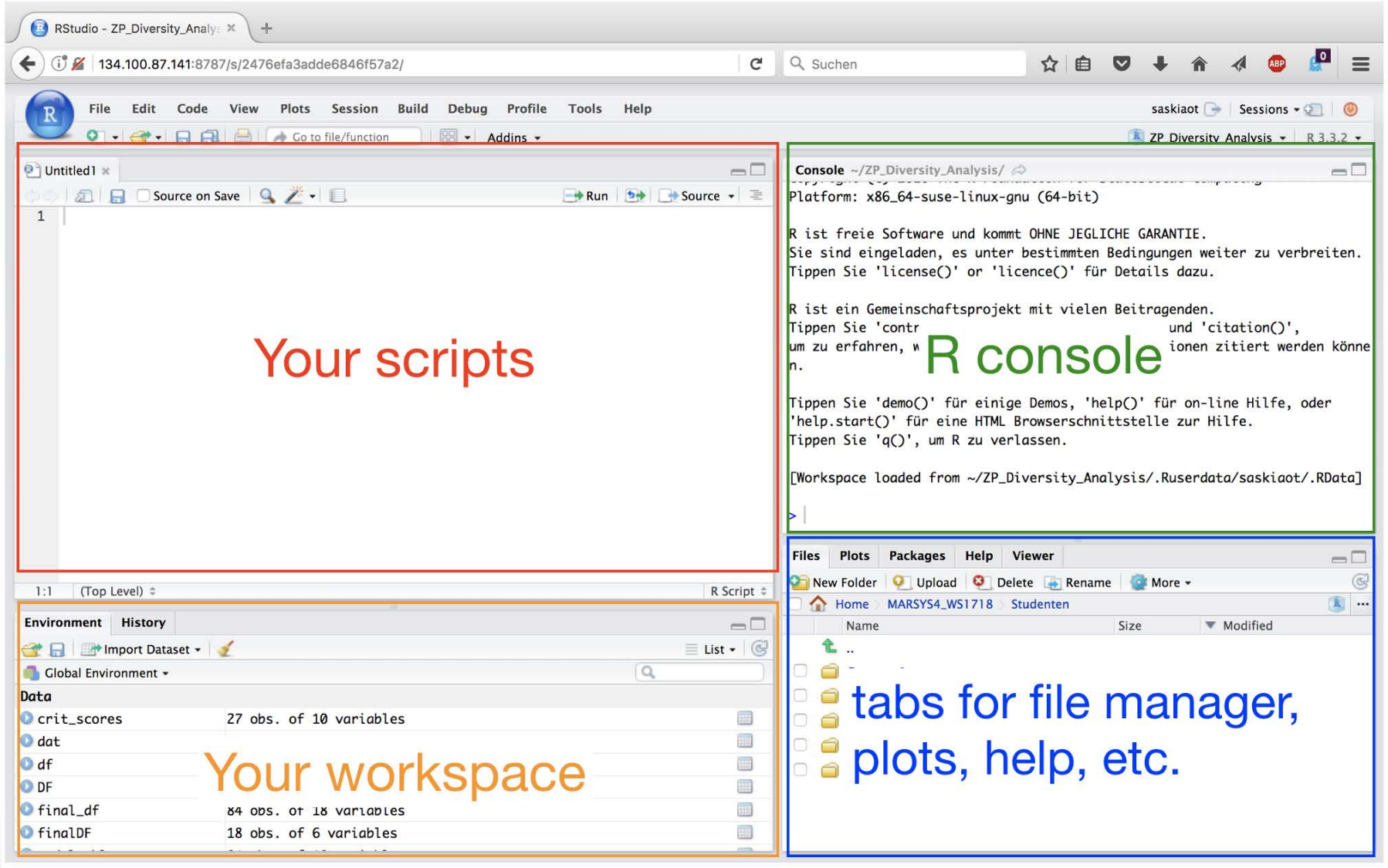


Use R from within RStudio

- A software program, which provides a GUI for R (trademark of RStudio, Inc.)
- Helps writing and executing R code and analyzing data with R.
- Integrated text editor and data and package manager.
- Provides version control, LaTeX integration, keyboard shortcuts, and debugging tools.
- Has become standard amongst R users.
- Open source and commercial editions available: www.rstudio.com
- Runs on desktops (Windows, Mac, and Linux) or in a web browser connected to RStudio Server → **We will use R Studio Server during the course**!

RStudio Server




RStudio - ZP_Diversity_Analys
134.100.87.141:8787/s/2476efa3adde6846f57a2/
Suchen
File Edit Code View Plots Session Build Debug Profile Tools Help
saskiaot
Sessions
Go to file/function
Addins
ZP_Diversity_Analysis
R 3.3.2
Untitled1
Source on Save
Run
Source
1
Your scripts
1:1
(Top Level)
R Script
Console ~/ZP_Diversity_Analysis/
Platform: x86_64-suse-linux-gnu (64-bit)
R ist freie Software und kommt OHNE JEGLICHE GARANTIE.
Sie sind eingeladen, es unter bestimmten Bedingungen weiter zu verbreiten.
Tippen Sie 'license()' or 'licence()' für Details dazu.
R ist ein Gemeinschaftsprojekt mit vielen Beitragenden.
Tippen Sie 'contr
und 'citation()',
um zu erfahren, w
ionen zitiert werden könne
n.
R console
Tippen Sie 'demo()' für einige Demos, 'help()' für on-line Hilfe, oder
'help.start()' für eine HTML Browserschnittstelle zur Hilfe.
Tippen Sie 'q()', um R zu verlassen.
[Workspace loaded from ~/ZP_Diversity_Analysis/.Ruserdata/saskiaot/.RData]
Environment History
Import Dataset
List
Global Environment
Data
crit_scores 27 obs. of 10 variables
dat
df
DF
final_df 84 obs. of 18 variables
finalDF 18 obs. of 6 variables
Your workspace
Files Plots Packages Help Viewer
New Folder Upload Delete Rename More
Home MARSYS4_WS1718 Studenten
Name Size Modified
..
tabs for file manager,
plots, help, etc.

Advantages of writing scripts

- **Transparency and reproducibility** - Not only the results but each step of the analysis are visible.
- **Flexibility** - Some analyses need only a few code tweaks of an existing R script.
- **Exchange** - In theory every R user should understand your script to allow easy sharing of code.

Work with scripts in R Studio

- Open a new script (**File ➔ New File ➔ R Script)**
- Write code into the empty script (in the editor pane)
- Send the code to the R console:
  - mark the code chunk and copy and paste it into the console (NOT recommended!)
  - press **ctrl + enter** ➔ code of current line (where the cursor is) is executed; cursor jumps then automatically to the next line of code
  - mark the code chunk ➔ ctrl + enter ➔ entire code chunk is executed

Run current line
Re-Run the previous section (ctrl + shift + p)
Run the whole script

Style guide - Some general recommendations

- Use a style guide and stick to it.
- Every script should be **as small as possible** and **as complex as needed**.
- Every script should be run by the console from start to end **without any error messages**.
- Use **"#"** to comment your code.
- Comment **why** you do something, not what you do.
- Never use `attach()`.
- Assignment operator: `<-` (do not use "=")

Style guide - Object names

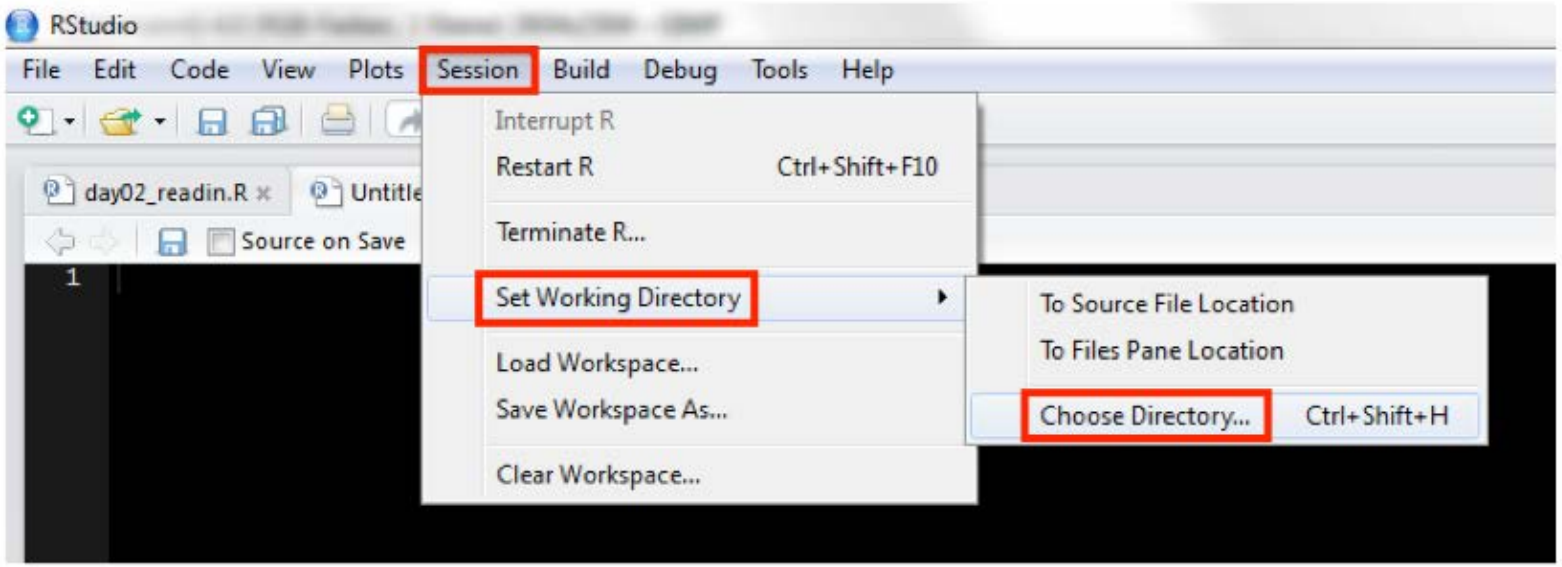
- Variable names should be **nouns** and function names should be **verbs**.
- Object names **cannot** begin with **numbers**.
- Don't uses special characters (e.g. !,/,%)
- Wise to **avoid** names already in use (for **functions**).
- Use **lowercase**
- Do **not** use **empty spaces** in names, instead combine with **underscore**.
- Ok to use:
    - a, x, my_list, my_dat, dat1
- **Not** recommended:
    - 1_a, c, list, _mydat, $dat

Style guide - Spacing

Put a space

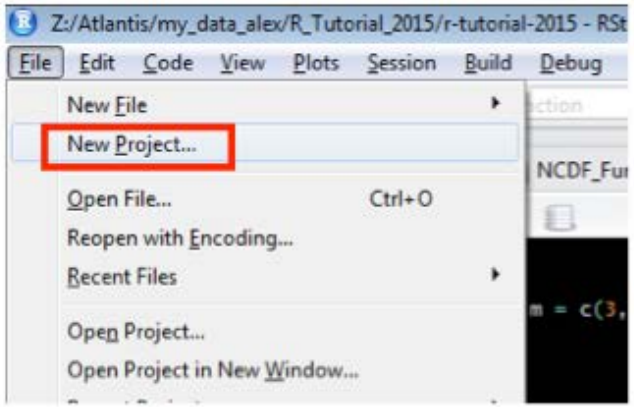
- **before** and **after** all infix operators (=, +, -, <-, etc.)
- when naming arguments in function calls
- after a comma, but never before

Before you start: set the working directory

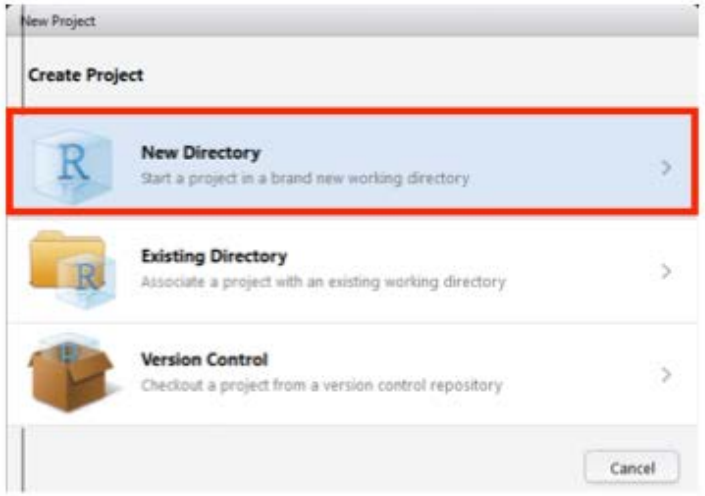


Better: Use R Projects

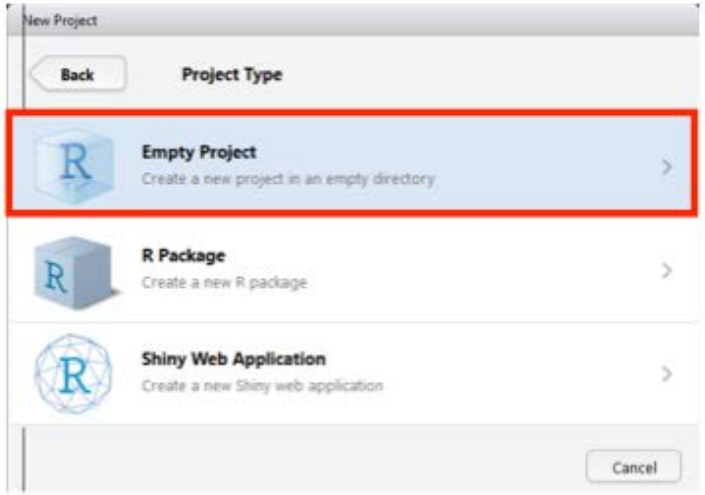
1.



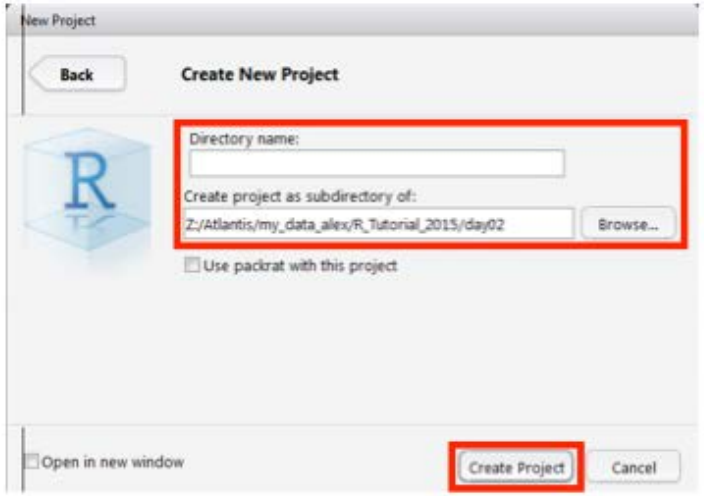
2.



3.

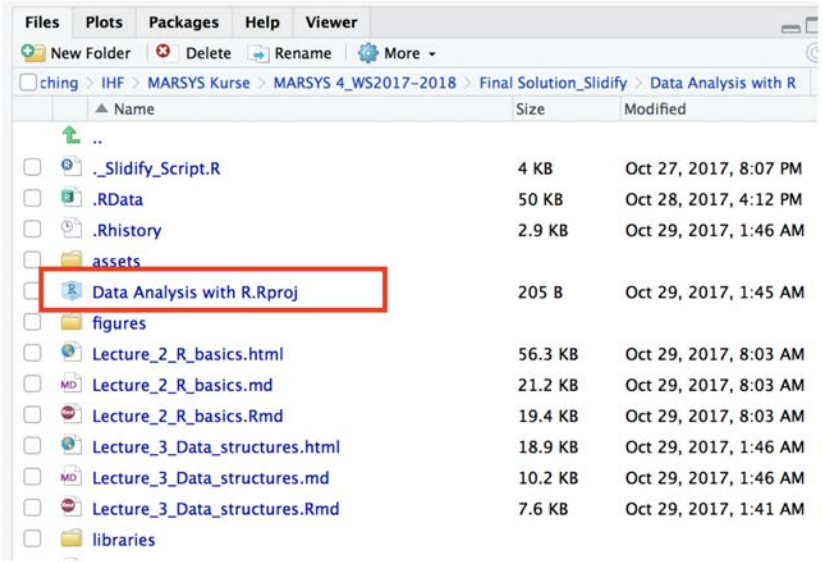


4.



R Projects (cont)

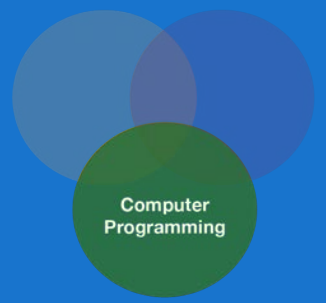
New folder stucture



Simply open the project file (here „Data Analysis with R.Rproj") and you are all set.

Advantages of projects

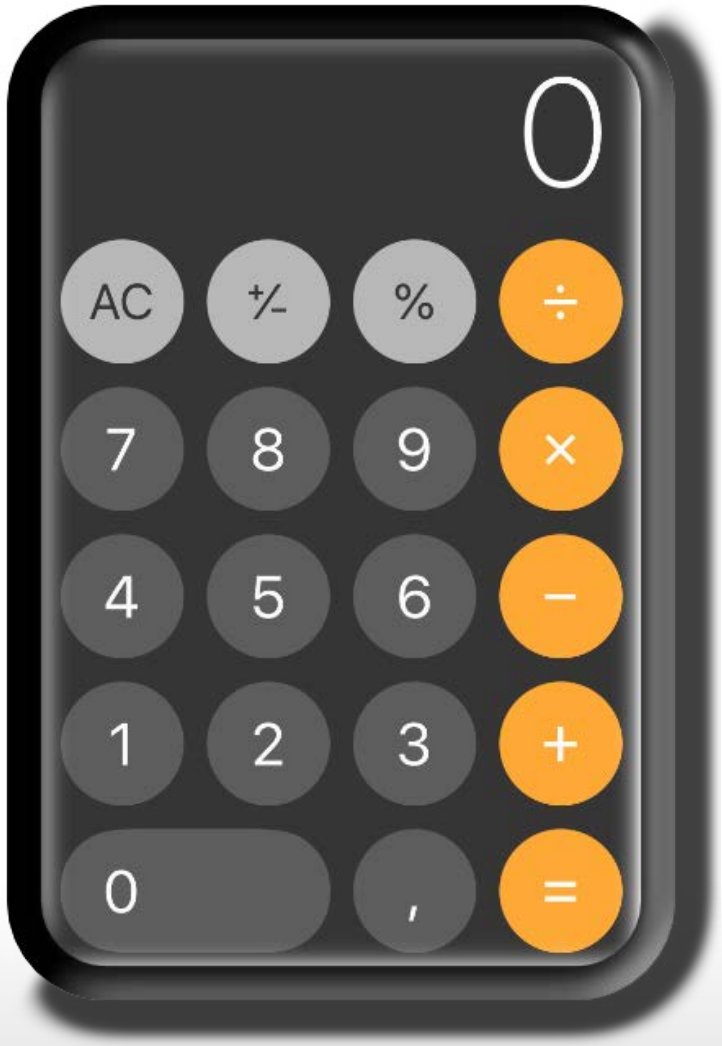
- **Pre-defined** folder structure
- Working directory is set **automatically**
- All **scripts** in this projects are **immediately** available
- Opens a **new R instance** so that one can switch between several instances



Arithmetics and functions in R

Basic calculations

In its most basic form, R can be used as a **simple calculator**.



Basic calculations

Consider the following **arithmetic operators**:

- Addition: `+`
- Subtraction: `-`
- Multiplication: `*`
- Division: `/`
- Exponentiation: `^`

Basic calculations

Consider the following **arithmetic operators**:

- Addition: `+`
- Subtraction: `-`
- Multiplication: `*`
- Division: `/`
- Exponentiation: `^`

```
5 + 5
5 - 5
3 * 5 + 2
(5 + 5) / 2
```

Basic calculations

Consider the following **arithmetic operators**:

- Addition: `+`
- Subtraction: `-`
- Multiplication: `*`
- Division: `/`
- Exponentiation: `^`

```
5 + 5
```

```
## [1] 10
```

```
5 - 5
```

```
## [1] 0
```

```
3 * 5 + 2 # multipl. then add.
```

```
## [1] 17
```

```
(5 + 5) / 2 # add. then div.
```

```
## [1] 5
```

Basic calculations (cont)

It also has **functions** that let you do more **sophisticated** manipulations, which you can **combine** by using **parentheses**:

```
a <- c(1,2,3,4)
c <- (a + sqrt(a))/(exp(2)+1)
```

Basic calculations (cont)

It also has **functions** that let you do more **sophisticated** manipulations, which you can **combine** by using **parentheses**:

```
a <- c(1,2,3,4)
c <- (a + sqrt(a))/(exp(2)+1)
```

Order of calculations (from the innermost to outermost parenthesis - just like a calculator).

Basic calculations (cont)

It also has **functions** that let you do more **sophisticated** manipulations, which you can **combine** by using **parentheses**:

```
a <- c(1,2,3,4)
c <- (a + sqrt(a))/(exp(2)+1)
```

Order of calculations (from the innermost to outermost parenthesis - just like a calculator).

1. `sqrt(a)` and `exp(2)`

Basic calculations (cont)

It also has **functions** that let you do more **sophisticated** manipulations, which you can **combine** by using **parentheses**:

```
a <- c(1,2,3,4)
c <- (a + sqrt(a))/(exp(2)+1)
```

Order of calculations (from the innermost to outermost parenthesis - just like a calculator).

1. `sqrt(a)` and `exp(2)`
2. then `a` added to `sqrt(a)` and `1` added to `exp(2)`

Basic calculations (cont)

It also has **functions** that let you do more **sophisticated** manipulations, which you can **combine** by using **parentheses**:

```
a <- c(1,2,3,4)
c <- (a + sqrt(a))/(exp(2)+1)
```

Order of calculations (from the innermost to outermost parenthesis - just like a calculator).

1. `sqrt(a)` and `exp(2)`
2. then `a` added to `sqrt(a)` and `1` added to `exp(2)`
3. then the division

   (sqrt = square root, exp = exponent)

+ prompt

```
>
>
>
>
> (mean(c(2,5,7,8,13)) / log(10)
+
```

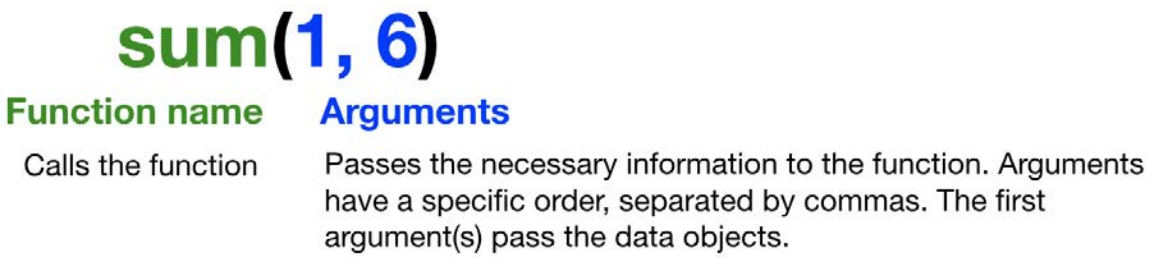
If your prompt turns into a **"+"**, R thinks you haven't finished your previous command. Either **finish the command**, or press **escape**.

A short introduction to functions in R

- Functions are the heart and soul of R.
- A function is a **block of code** that gives **instructions** to R to carry out.
- Some functions come in R's **base** package others in **additional packages**.
- They work similar to functions in other packages:

A short introduction to functions in R

- Functions are the heart and soul of R.
- A function is a **block of code** that gives **instructions** to R to carry out.
- Some functions come in R's **base** package others in **additional packages**.
- They work similar to functions in other packages:
  - in **Excel**, to sum up over cells A1 to A10 you write **sum(A1:A23)**.
  - in R there is an equivalent function named `sum()`, which takes the elements it should sum over as argument:



Getting help

Getting help for a **specific** function:

```
help("mean")
?mean
```

Search the help pages:

```
??mean
help.search("mean")
```

List **all** functions, which contain "mean":

```
apropos("mean")
```

R Documentation for the function `mean`

Files | Plots | Packages | Help | Viewer

mean

R: Arithmetic Mean Find in Topic

mean {base} **Function - and package name** R Documentation

Arithmetic Mean

Description

Generic function for the (trimmed) arithmetic mean.

Tells you in short what the function does

Usage

```
mean(x, ...)

## Default S3 method:
mean(x, trim = 0, na.rm = FALSE, ...)
```

How to use the functions (which arguments are in which order)

Arguments

`x`	An `R` object. Currently there are methods for numeric/logical vectors and date, date-time and time interval objects. Complex vectors are allowed for `trim = 0`, only.
`trim`	the fraction (0 to 0.5) of observations to be trimmed from each end of `x` before the mean is computed. Values of trim outside that range are taken as the nearest endpoint.
`na.rm`	a logical value indicating whether `NA` values should be stripped before the computation proceeds.
`...`	further arguments passed to or from other methods.

More information on each argument (or parameter) that goes into the function

R Documentation for the function `mean`

2nd part of the documentation

Value

If `trim` is zero (the default), the arithmetic mean of the values in `x` is computed, as a numeric or complex vector of length one. If `x` is not logical (coerced to numeric), numeric (including integer) or complex, `NA_real_` is returned, with a warning.

If `trim` is non-zero, a symmetrically trimmed mean is computed with a fraction of `trim` observations deleted from each end before the mean is computed.

What the function returns

References

Becker, R. A., Chambers, J. M. and Wilks, A. R. (1988) *The New S Language*. Wadsworth & Brooks/Cole.

See Also

`weighted.mean`, `mean.POSIXct`, `colMeans` for row and column means.

Related function → can be useful to look them up

Examples

```
x <- c(0:10, 50)
xm <- mean(x)
c(xm, mean(x, trim = 0.10))
```

The last section shows how the function can be used with real data. Always worth to take a look and play around with the examples!

[Package *base* version 3.4.2 Index]

Your turn...

Quiz 1: Simple calculations

Open a new script in your R Studio Server environment and save it before you start writing anything. Now calculate the following and write the result in the boxes below:

1. Substract 10 from 23, then multiply with 2.

2. Substract 10 from 23, then multiply with 2, then add 100, then divide all by 5.

3. Substract 23 from 10, then multiply with -10, then take the square root (use `sqrt()` function).

Submit | Show Hint | Show Answer | Clear

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Thank You

For more information contact me: saskia.otto@uni-hamburg.de

http://www.researchgate.net/profile/Saskia_Otto
http://www.github.com/saskiaotto

Image on title and end slide: Section of an infrared satallite image showing the Larsen C ice shelf on the Antarctic Peninsula - USGS/NASA Landsat: A Crack of Light in the Polar Dark, Landsat 8 - TIRS, June 17, 2017