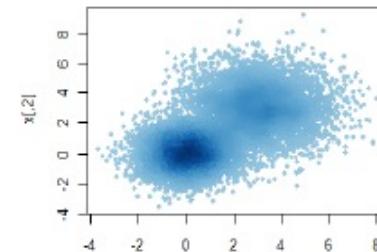
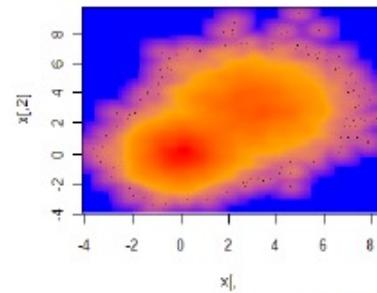
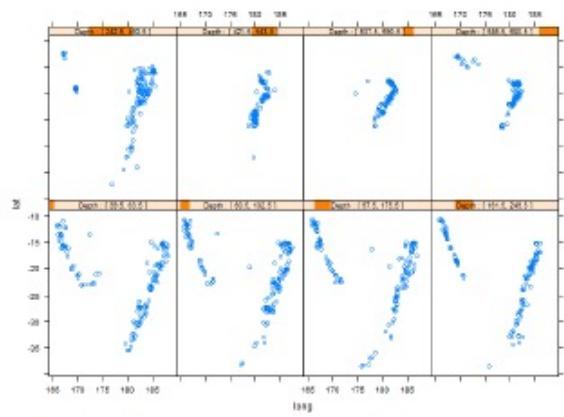
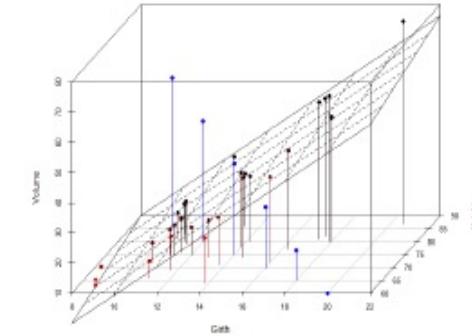
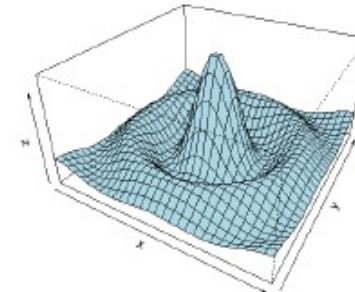
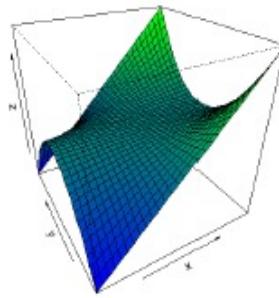
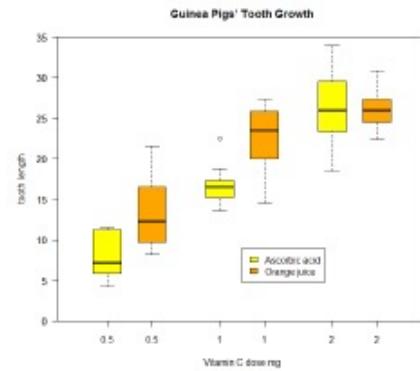


u^b

b
UNIVERSITÄT
BERN

OESCHGER CENTRE
CLIMATE CHANGE RESEARCH

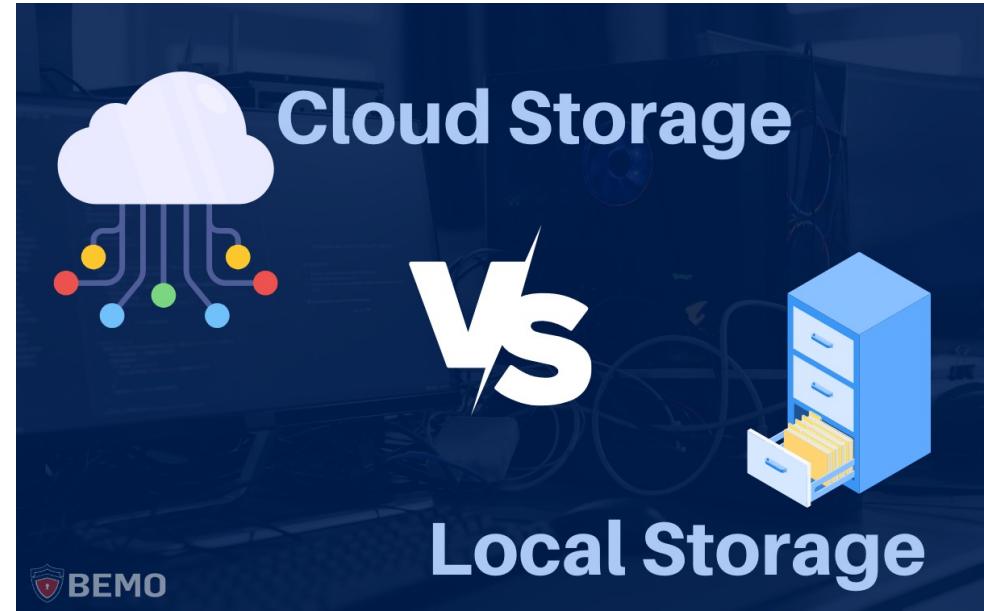
Computer Basics and R



Computer Basics

Dateisystem

- > Dateisystem und Mountpunkte in Finder und Konsole zeigen



Computer Basics

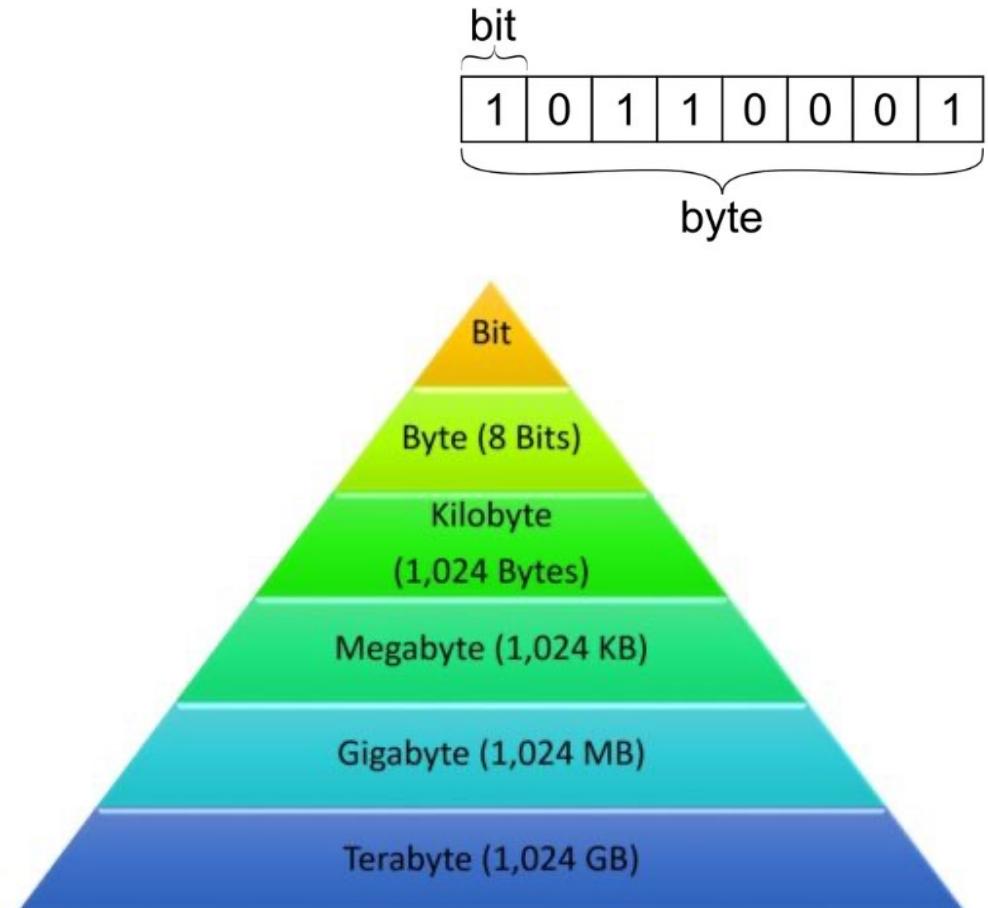
1 Bit: 1/0

1 Byte: kodiert einen Buchstaben

1 Megabyte ?

1 Gigabyte ?

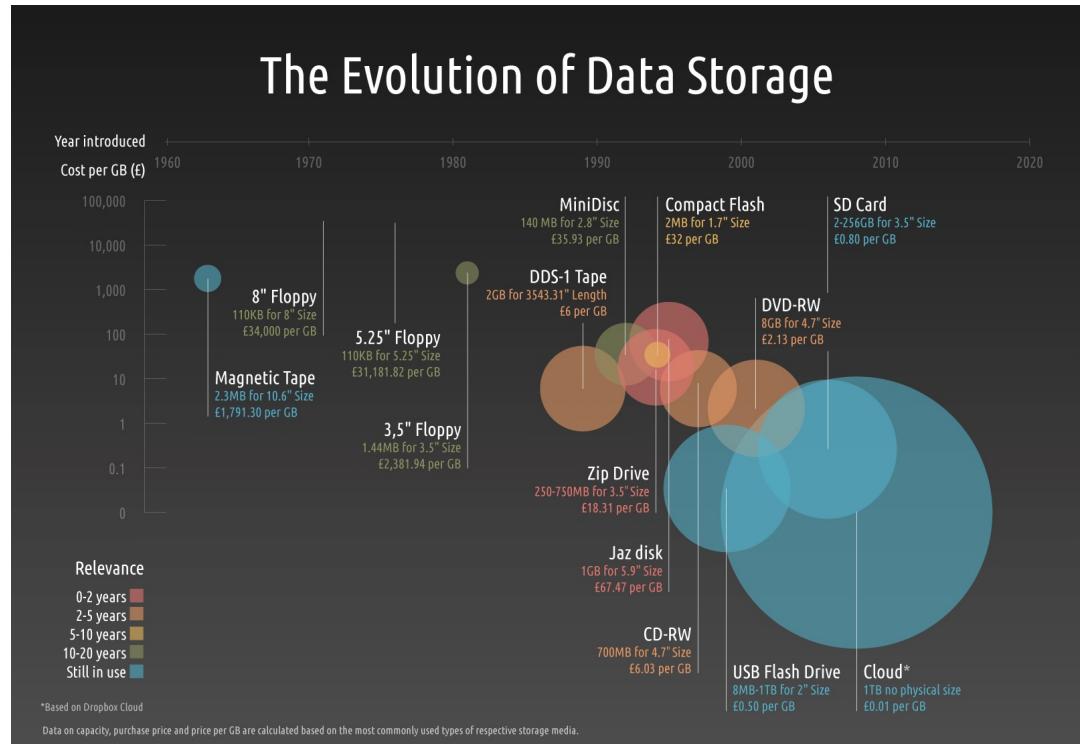
1 Terabyte ?



Computer Basics

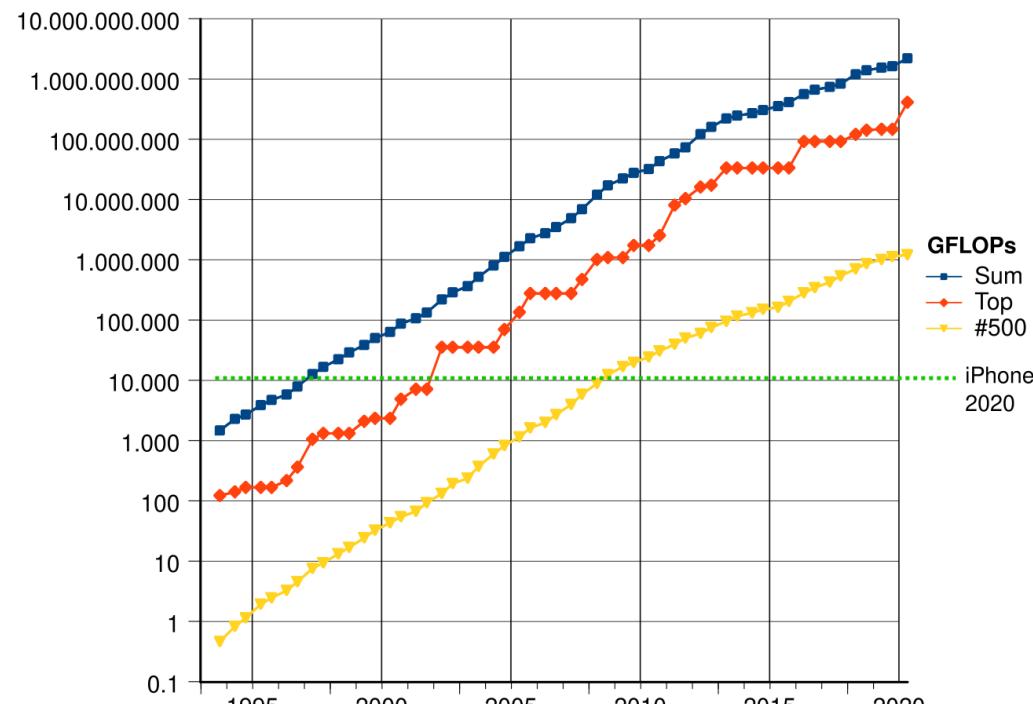
Datengrößen:

- > 100 Jahresmittel der Temperatur an einer Station wenige KB
- > 100 Jahre Tagesmittel einer Station ca. 1 MB
- > Zeitschritt eines Wettervorhersagemodells multivariat bei 1 km räumlicher Auflösung ca. 2 TB

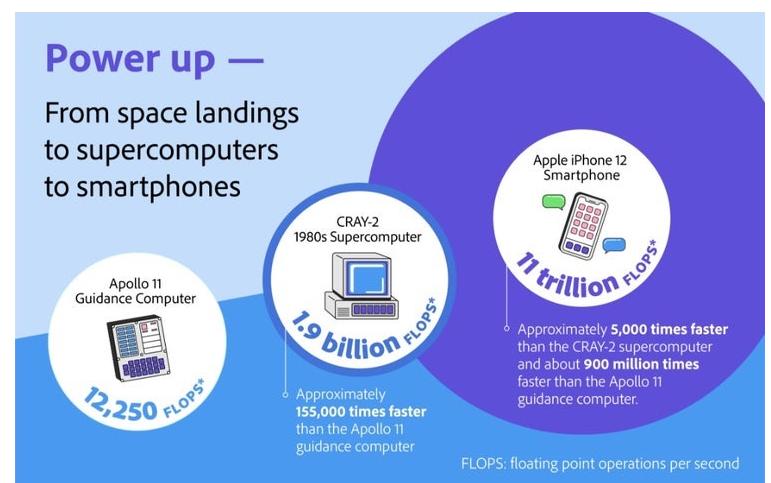


Computer Basics

Computerleistungen einschätzen (iPhone vs Supercomputer)



FLOPS = Floating point operations per second



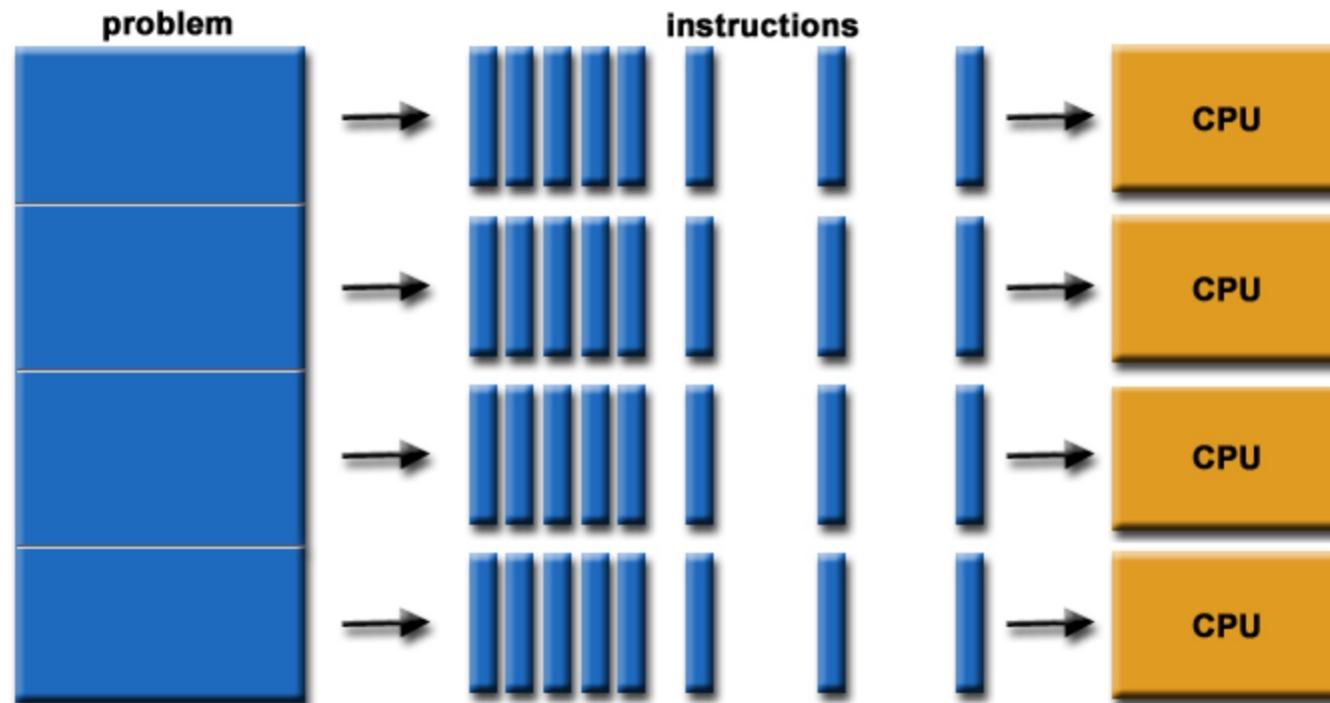
Cloud Computing

- › benutzt ihr die Online Office Version im Browser oder eine auf eurem Computer installierte und wieso?



Computer Basics

CPU Kerne / Parallelisierung



Flaschenhals: Rechenzeit oder Schreibzeit

Computer Basics

Speicherplatzoptimierung?

- > Zwischenschritte speichern?
- > Am Ende wieder löschen?
- > Dateiformate/Komprimierung

Codeoptimierung ?

- > Zeit zum Code schreiben / optimieren
- > Laufzeit des Codes
- > Anzahl der Codeläufe

Computer Basics

Organisation, Dateiformate und Dateinamen

- > KEINE Leerzeichen
- > KEINE Sonderzeichen wie Umlaute
- > Betriebssystemunabhängigkeit
- > ACHTUNG: Länderspezifische Besonderheiten wie Dezimalkomma vs Dezimalpunkt

1 FIGURE OUT LOCATION

Files can be stored in desktop, documents, downloads, or shared drive.



2 ORGANIZE BY FOLDERS

Organize by dates, departments, or events.



3 ORGANIZE BY SUBFOLDERS

Use date or project type.



4 FORMAT AND GROUP FILES

Format based off file type like excel, doc, pdf, jpeg, and png.



5 NAME FILES STRATEGICALLY

Think about how you would search for it in the future.

6 DOCUMENT THE PROCESS

Create a Standard Operating Procedure (SOP's) for consistency.



7 MAINTENANCE IS KING

Have a routine to update, move, and delete files regularly.

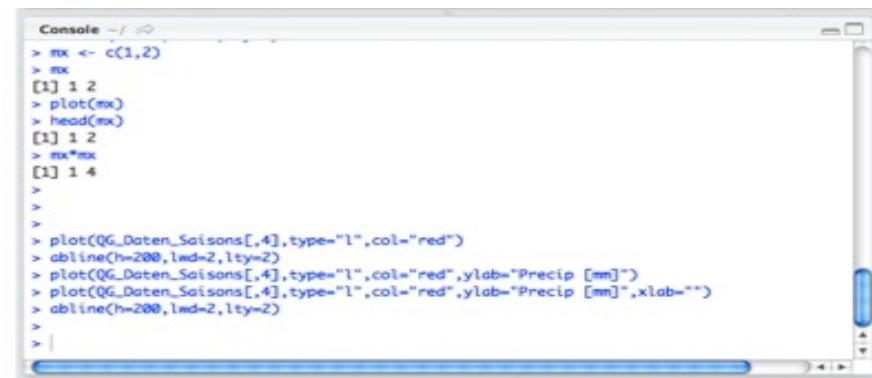


8 OTHER BEST PRACTICES

File immediately and do not store files on your desktop.

Was ist R?

- > Einfache Programmiersprache zur Datenanalyse, statistischen Auswertung und Visualisierung, d.h.
- > geeignet für jegliche Datenbearbeitung
- > inklusive Abbildungen jeder Art, auch Karten
- > "Einfach" da kein Compilieren notwendig ist
- > Freie Software
- > Kein offizielles Menü-System -> Kommando basiert + Text editor
- > 'packages' mit ständig von Benutzer ergänzten Zusatzfunktionen frei im Internet verfügbar
- > Gute Hilfe im Netz



The screenshot shows an R console window with the following text:

```
Console - / 
> mx <- c(1,2)
> mx
[1] 1 2
> plot(mx)
> head(mx)
[1] 1 2
> mx*mx
[1] 1 4
>
>
> plot(QG_Daten_Saisons[,4],type="l",col="red")
> abline(h=200,lwd=2,lty=2)
> plot(QG_Daten_Saisons[,4],type="l",col="red",ylab="Precip [mm]")
> plot(QG_Daten_Saisons[,4],type="l",col="red",ylab="Precip [mm]",xlab="")
>
>
```

Literatur: <https://r4ds.had.co.nz>

Welcome

☰ ⚅ A ⌂ i R for Data Science

1 Introduction

I Explore

2 Introduction

3 Data visualisation

4 Workflow: basics

5 Data transformation

6 Workflow: scripts

7 Exploratory Data Analysis

8 Workflow: projects

II Wrangle

9 Introduction

10 Tibbles

11 Data import

12 Tidy data

13 Relational data

14 Strings

15 Factors

16 Dates and times

III Program

17 Introduction

R for Data Science

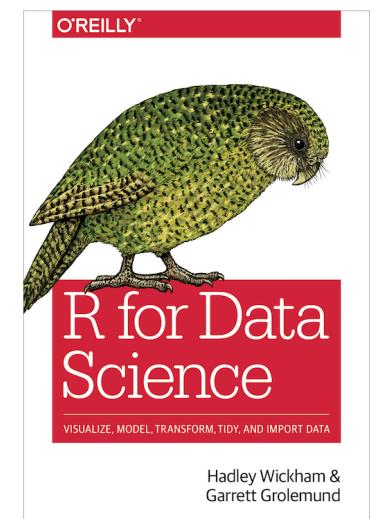
Garrett Grolemund

Hadley Wickham

Welcome

This is the website for “R for Data Science”. This book will teach you how to do data science with R: You’ll learn how to get your data into R, get it into the most useful structure, transform it, visualise it and model it. In this book, you will find a practicum of skills for data science. Just as a chemist learns how to clean test tubes and stock a lab, you’ll learn how to clean data and draw plots—and many other things besides. These are the skills that allow data science to happen, and here you will find the best practices for doing each of these things with R. You’ll learn how to use the grammar of graphics, literate programming, and reproducible research to save time. You’ll also learn how to manage cognitive resources to facilitate discoveries when wrangling, visualising, and exploring data.

This website is (and will always be) **free to use**, and is licensed under the [Creative Commons Attribution-NonCommercial-NoDerivs 3.0](#) License. If you’d like a **physical**



Editor, hier wird programmiert und der Code dann in der Konsole ausgeführt

R-Studio (graphische Benutzeroberfläche)

Konsole / Kommandozeile

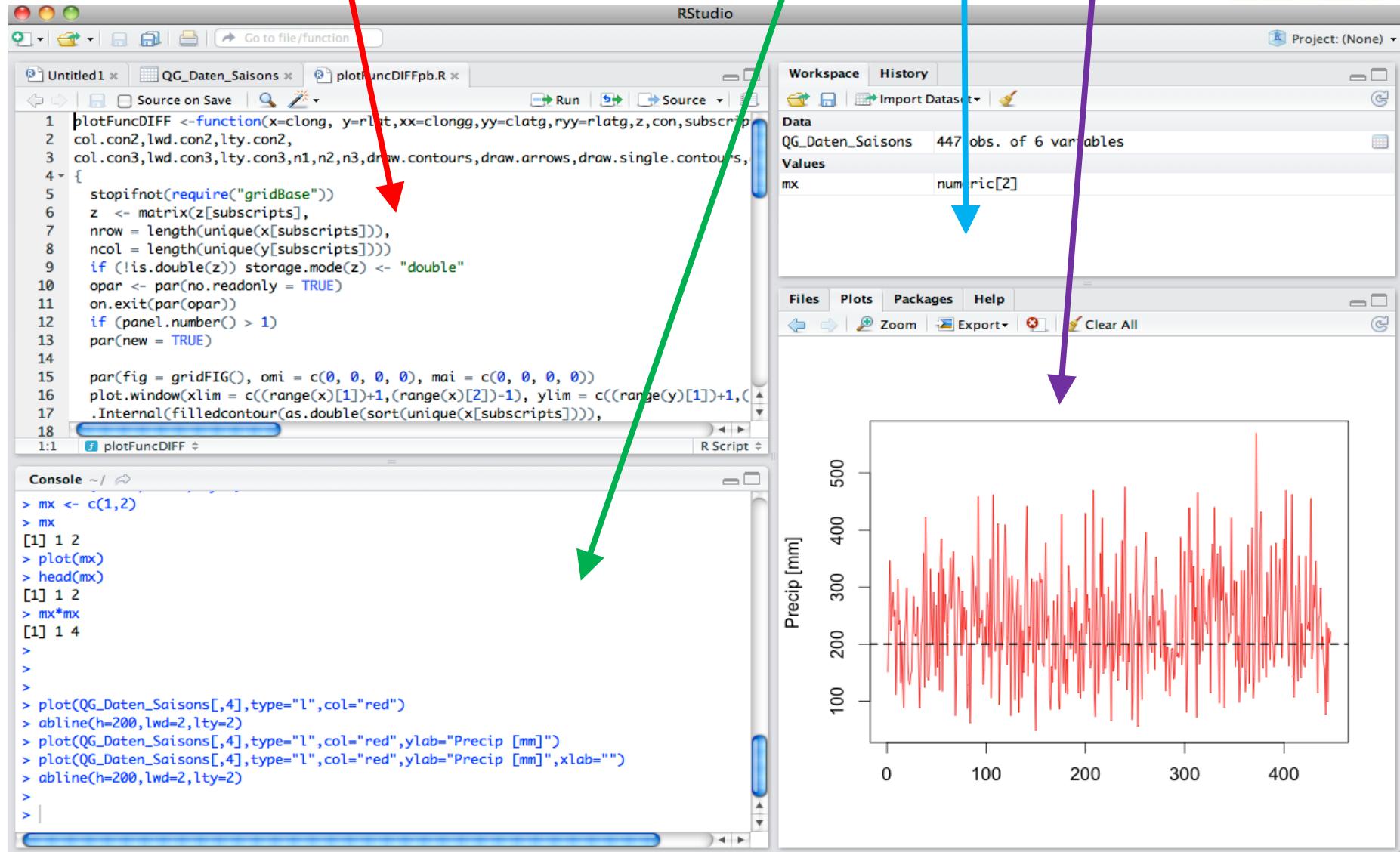
Plots/Hilfe

u^b

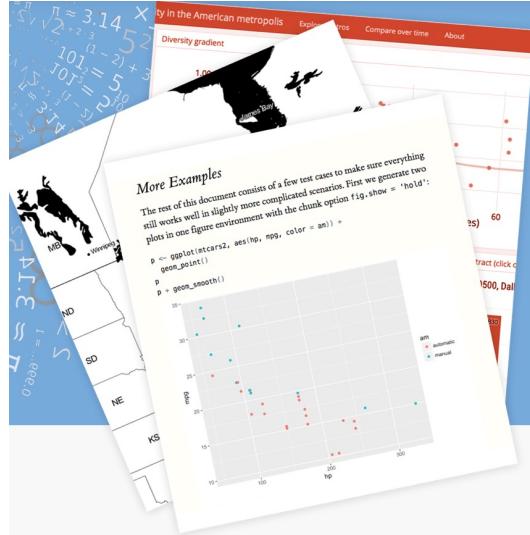
UNIVERSITÄT
BERN

OESCHGER CENTRE

RCH



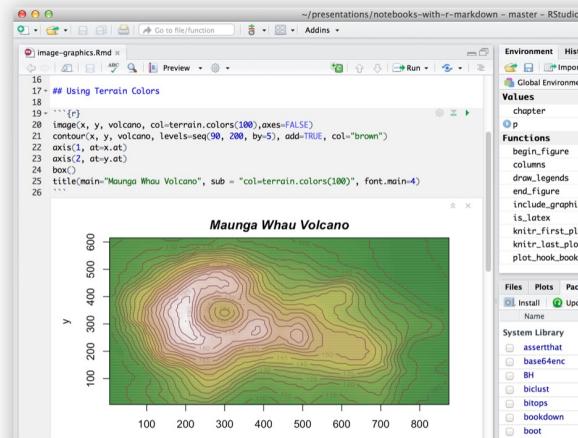
R Markdown or Notebooks



R Markdown documents are fully reproducible. Use a productive [notebook interface](#) to weave together narrative text and code to produce elegantly formatted output. Use [multiple languages](#) including R, Python, and SQL.

Analyze. Share. Reproduce.

Your data tells a story. Tell it with R Markdown.
Turn your analyses into high quality documents, reports, presentations and dashboards.



R Markdown supports dozens of static and dynamic output formats including [HTML](#), [PDF](#), [MS Word](#), [Beamer](#), [HTML5 slides](#),

R Markdown

Cheat Sheet
learn more at rmarkdown.rstudio.com

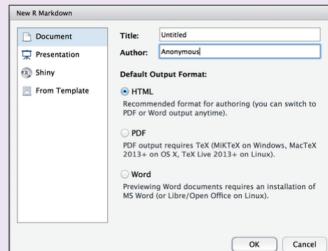
rmarkdown 0.2.50 Updated: 8/14



2. Open File

Start by saving a text file with the extension .Rmd, or open an RStudio Rmd template

- In the menu bar, click **File ▶ New File ▶ R Markdown...**
- A window will open. Select the class of output you would like to make with your .Rmd file
- Select the specific type of output to make with the radio buttons (you can change this later)
- Click OK



4. Choose Output

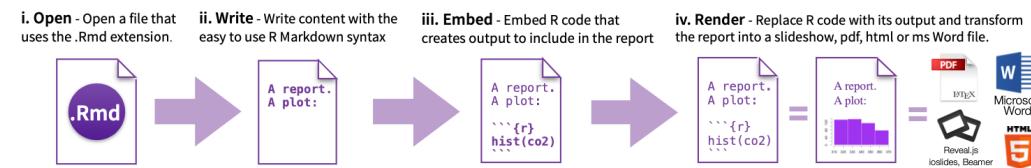
Write a YAML header that explains what type of document to build from your R Markdown file.

YAML

A YAML header is a set of key: value pairs at the start of your file. Begin and end the header with a line of three dashes (---)

```
title: "Untitled"
author: "Anonymous"
output: html_document
---
This is the start of my report. The above is metadata saved in a YAML header.
```

1. Workflow R Markdown is a format for writing reproducible, dynamic reports with R. Use it to embed R code and results into slideshows, pdfs, html documents, Word files and more. To make a report:



3. Markdown

Next, write your report in plain text. Use markdown syntax to describe how to format text in the final report.

syntax

Plain text
End a line with two spaces to start a new paragraph.
italics and _italics_
bold and __bold__
superscript^2^
~~strikethrough~~
[link](www.rstudio.com)

```
# Header 1
## Header 2
### Header 3
#### Header 4
##### Header 5
###### Header 6
```

```
endash: --
emdash: ---
ellipsis: ...
inline equation: $A = \pi r^2$
image: 

horizontal rule (or slide break):
***
```

> block quote

becomes

Plain text
End a line with two spaces to start a new paragraph.
italics and italics
bold and bold
superscript²
strikethrough
[link](#)

Header 1

Header 2

Header 3

Header 4

Header 5

Header 6

endash: –
emdash: —
ellipsis: ...
inline equation: $A = \pi * r^2$

image:

horizontal rule (or slide break):

block quote

R-Struktur, Objektarten

Datentypen: numerisch, alphanumerisch (Text), logisch (TRUE/FALSE)

Objektarten

Vektoren: kein Mischen von Datentypen

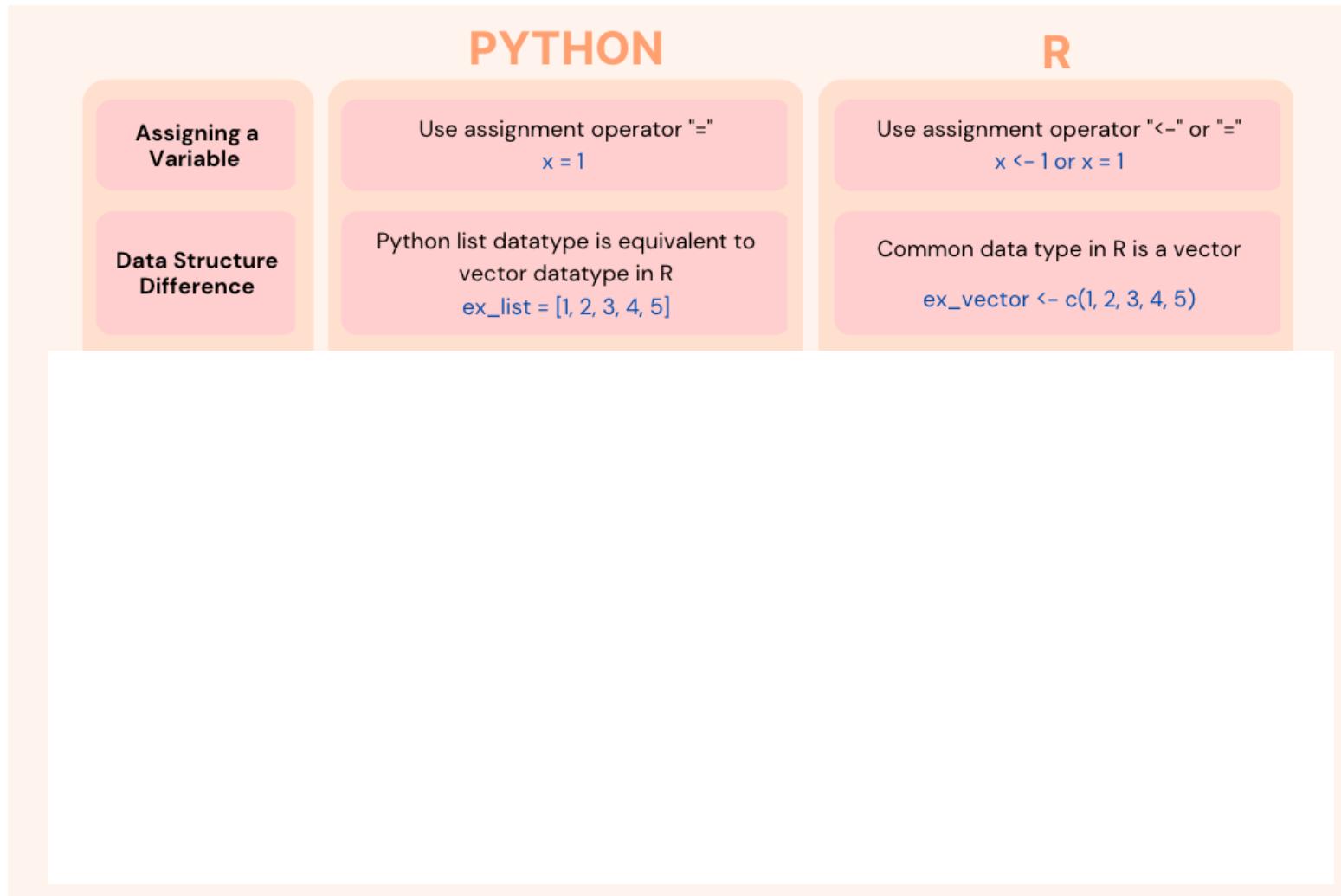
Data Frames ('Tabellen'): Mischen möglich

Matrizen: kein Mischen von Datentypen

Listen: mehrere Vektoren, Data Frames, Matrizen können in Listen zusammengefasst werden

Objekte erzeugt man in dem man einen Objekt-Namen vergibt und Werte mit <- oder = zuweist!

Python vs R



ACHTUNG: copy/paste funktioniert NICHT!!!



b
UNIVERSITÄT
BERN

OESCHGER CENTRE
CLIMATE CHANGE RESEARCH

Objektart: Vektor

Numerische Vektoren

```
> a <- c(4, 2, 7) # c Funktion steht für combine or concatenate  
# Parameter von Funktionen immer mit RUNDEN Klammern
```

```
> a
```

```
[1] 4 2 7
```

```
> b <- c(3.1, 5, -0.7)
```

```
> c <- c(a, b)
```

```
> c
```

```
[1] 4.0 2.0 7.0 3.1 5.0 -0.7
```

Einfache oder doppelte Anführungszeichen funktionieren!

Alphanumerische Vektoren

```
> station <- c('Bern', 'Biel', 'Olten')
```

Logische Vektoren

```
> e <- c(TRUE, FALSE, TRUE)
```

Objektart: Vektor

Logische Vektoren

```
> f <- 1:5
```

```
[1] 1 2 3 4 5
```

```
> f >= 3
```

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

Vergleichsoperationen <, <=, >, >=, ==, !=

Operationen & (und), | (oder), ! (nicht)

```
> g <- (f>2) & (f<5)
```

```
> g
```

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

Objektart: Vektor

```
> seq(0, 3, by=0.5) ... erzeugt Sequenz von 0 bis 3 im 0.5 Abstand  
[1] 0.0 0.5 1.0 1.5 2.0 2.5 3.0
```

```
> rep(5, 3) ... wiederholt 3x den Wert 5  
[1] 5 5 5
```

```
> rep(c(0, 3), length=9) ) ... wiederholt 0 u. 3 bis Vektor 9 Elemente hat  
[1] 0 3 0 3 0 3 0 3 0
```

```
> rep(c(0:3), length=9) ... wiederholt 0,1,2,3 bis Vektor 9 Elem. hat  
[1] 0 1 2 3 0 1 2 3 0 1
```

Objektart: Vektor

- > Wichtige grundlegende Funktionen

Funktion, Beispiel	Bedeutung
<i>length(a)</i>	Länge, Anzahl Elemente
<i>sum(a)</i>	Summe aller Elemente
<i>mean(a)</i>	arithmetisches Mittel der Elemente
<i>var(a)</i>	empirische Varianz
<i>sd(a)</i>	emp. Standardabweichung
<i>range(a)</i>	Wertebereich
<i>min(a), max(a)</i>	Minimum, Maximum Wert des Vektors

Python vs R

PYTHON	R
<p>Chaining operations</p> <p>Method chains with the "." operator <code>df.head(n)</code> or <code>df.describe()</code></p>	<p>Can chain operations using following symbol: "%>%" <code>df %>% head(n)</code> or <code>df %>% summary()</code></p>

Objektart: Data Frame

Vektoren unterschiedlicher Datentypen als Tabelle (=Data Frame) zusammenfassen

> *Dat <- data.frame(station, a, b)* ... Data Frame mit Namen 'Dat' erzeugen

	<i>station</i>	<i>a</i>	<i>b</i>	... Spaltennamen (z.B. <i>a</i> ist Bewölkung, <i>b</i> ist Temperatur zu einem Zeitpunkt)
1	<i>Bern</i>	4	3.1	
2	<i>Biel</i>	2	5.0	
3	<i>Olten</i>	7	-0.7	

Elemente aus Objekten auswählen

Vektor [ELEMENTNUMMER]

> *a[3]; a[1:2]* ... 3. Element; ... 1. bis 2. Element (also 1. und 2.)

Data Frame/Matrix [ZEILE , SPALTE]

> *Dat[1,1]* ... 1. Zeile, 1. Spalte

> *Dat[c(1,2),1]* ... 1. und 2. Zeile der 1. Spalte

> *Dat[1:2,3]* ... 1. bis 2. Zeile der 3. Spalte

> *Dat[,3]* ... alle Zeilen der 3. Spalte (*leer = alle*)

> *Dat[-3,]* ... alle Zeilen bis auf die 3. Zeile, alle Spalten

oder mit Spalten/Zeilennamen

> *Dat[1:2,'b']*

[1] 3.1 5.0

Python vs R

PYTHON

R

Indexing and slicing

Indexing starts from 0, inclusive of the start index and excludes the end index.
`ex_list[0]` and `ex_list[0:2]`
`output: 1` `output: [1, 2]`

Indexing starts from 1, Indexing is inclusive of both start and end index
`ex_vector[1]` and `ex_vector[1:2]`
`output: 1` `output: 1, 2`

Hilfe in R

mit **?Funktionsname** oder **help(Funktionsname)**

```
> ?seq  
> ??histogram  
> help(sum)  
> example(histogram)  
> ...
```

Hilfe im Netz: **GOOGLE, CHATBOTS!**

R als Taschenrechner

> *2 + 5*

[1] 7

> *(2:5)^2* ... auf Vektoren Operationen elementweise angewandt;

[1] 4 9 16 25 ^ oder ** =Potenzfunktion

> *(a+1) * b* ... Klammern wie üblich

[1] 15.5 15.0 -5.6 11.7

> *abs(b)* ... Absolutwert (Betrag)

[1] 3.1 5.0 0.7 1.3

> *exp(a)* ... Exponentialfunktion (e^a)

[1] 54.598150 7.389056 1096.633158 2980.957987

> *log(a)* ... Natürlicher Logarithmus ($\exp(1)^{\log(a)}=a$)

[1] 1.3862944 0.6931472 1.9459101 2.0794415

> *sqrt(a)* ... Wurzel (Square root)

Neues R-Skript oder R Notebook erstellen und umbedingt speichern!!!

u^b

^b
UNIVERSITÄT
BERN

OESCHGER CENTRE

RCH

The screenshot shows the RStudio interface. The top bar displays the title "RStudio". The left pane contains a script editor with the code for "plotFuncDIFF" and a console window below it. The right pane includes a "Workspace" panel showing the dataset "QG_Daten_Saisons" and its variables, and a "Plots" panel displaying a line plot of precipitation over time.

Script Editor (Source)

```
1 plotFuncDIFF <- function(x=clong, y=rlat,xx=clongg,yy=clatg,z,con,subscript
2 col.con2,lwd.con2,lty.con2,
3 col.con3,lwd.con3,lty.con3,n1,n2,n3,draw.contours,draw.arrows,draw.single.contours,
4 {
5 stopifnot(require("gridBase"))
6 z <- matrix(z[subscripts]),
7 nrow = length(unique(x[subscripts])),
8 ncol = length(unique(y[subscripts])))
9 if (!is.double(z)) storage.mode(z) <- "double"
10 opar <- par(no.readonly = TRUE)
11 on.exit(par(opar))
12 if (panel.number() > 1)
13 par(new = TRUE)
14
15 par(fig = gridFIG(), omi = c(0, 0, 0, 0), mai = c(0, 0, 0, 0))
16 plot.window(xlim = c((range(x)[1])+1,(range(x)[2])-1), ylim = c((range(y)[1])+1,(range(y)[2])-1),
17 .Internal(filledcontour(as.double(sort(unique(x[subscripts]))))),
18 )
1:1 f plotFuncDIFF
```

Console

```
> mx <- c(1,2)
> mx
[1] 1 2
> plot(mx)
> head(mx)
[1] 1 2
> mx*mx
[1] 1 4
>
>
>
> plot(QG_Daten_Saisons[,4],type="l",col="red")
> abline(h=200,lwd=2,lty=2)
> plot(QG_Daten_Saisons[,4],type="l",col="red",ylab="Precip [mm]")
> plot(QG_Daten_Saisons[,4],type="l",col="red",ylab="Precip [mm]",xlab="")
> abline(h=200,lwd=2,lty=2)
>
> |
```

Plots

A line plot titled "Precip [mm]" showing precipitation over time. The x-axis ranges from 0 to 450, and the y-axis ranges from 0 to 500. A horizontal dashed line is drawn at 200 mm. The plot shows a highly variable pattern with several sharp peaks exceeding 400 mm.

Verzeichnis finden

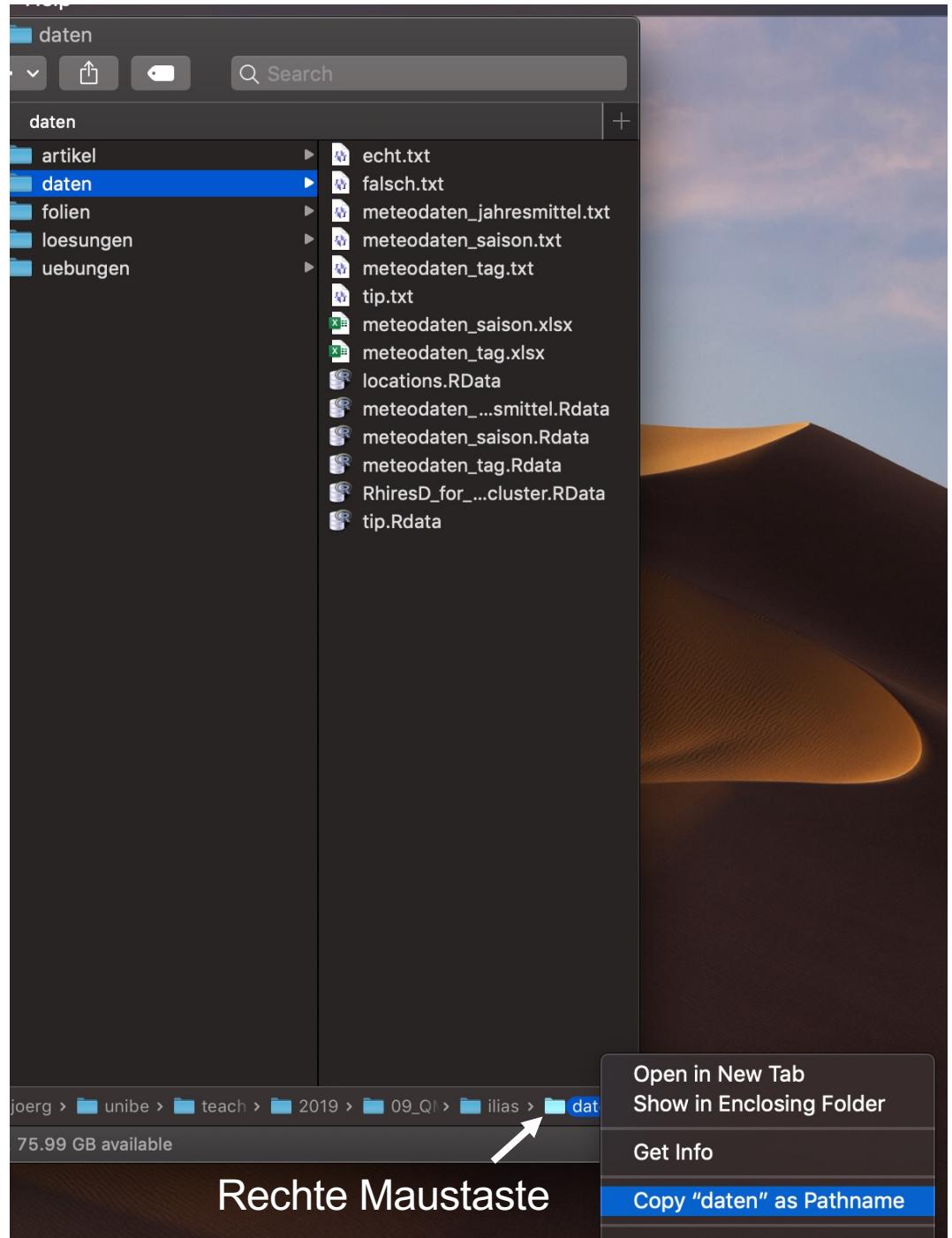
1. Im Datei "Explorer" (Windows) oder "Finder" (Mac) das Verzeichnis finden, wo ihr eure Dateien zu dieser Veranstaltung speichert
2. Beispieldaten von Ilias in dieses R Verzeichnis speichern/runterladen
3. Vollständigen Pfadnamen kopieren (siehe Abb. rechts)

Mac: '/Users/name/quant_meth...')

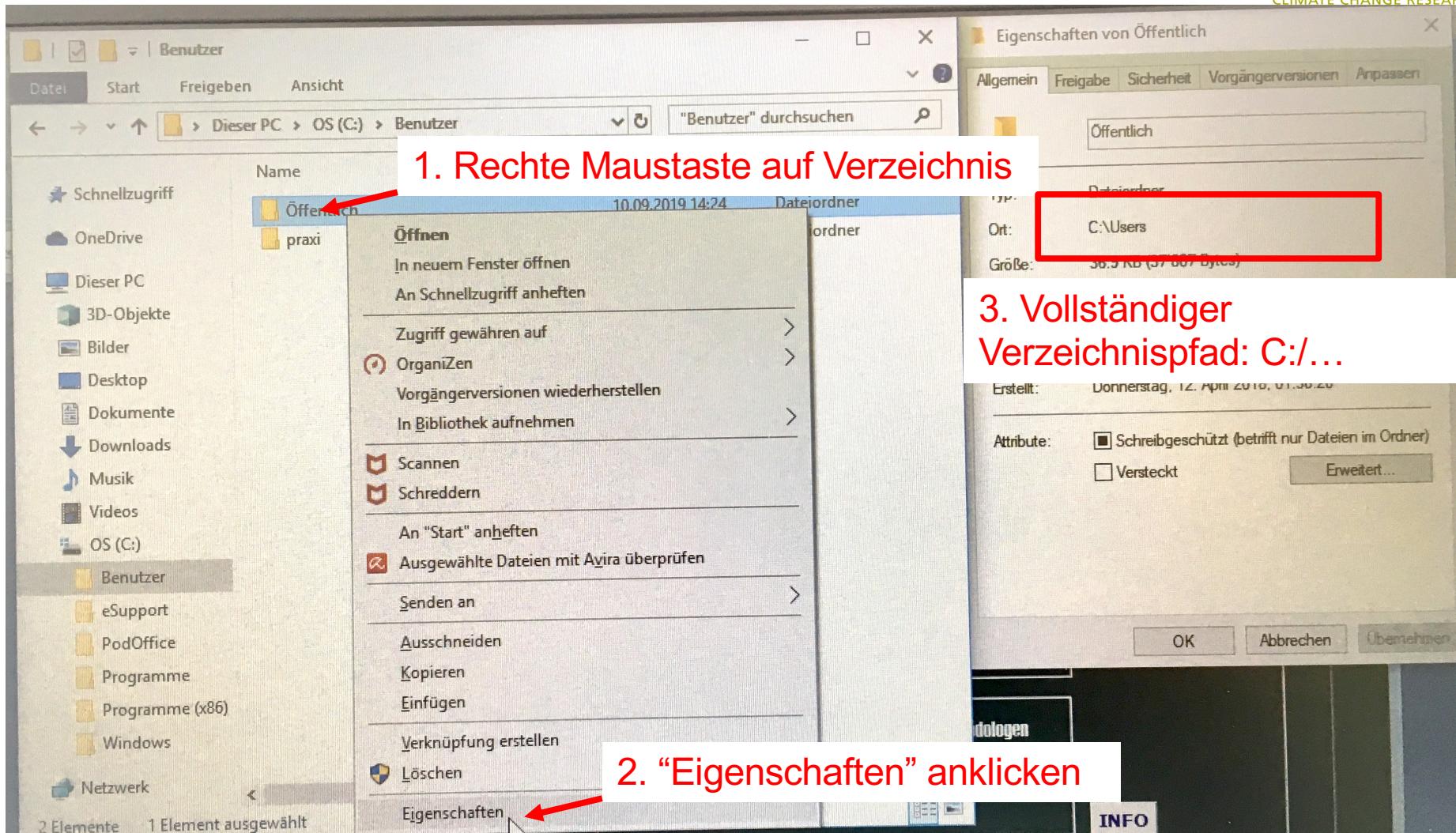
bzw.

Windows: 'C:/Users/...'

ACHTUNG bei Windows wird \ benutzt und muss für R in / verwandelt werden!



Arbeitsverzeichnis im Windows Explorer



Daten einlesen

Dateien im Verzeichnis anzeigen

```
> list.files('/Users/name/quant_meth...')
```

sollte alle Dateien und Unterverzeichnisse im angegeben Verzeichnis anzeigen

ASCII Daten einlesen:

Textdatei 'file.txt' bzw. 'file.csv' aus Excel exportiert

```
> saison <- read.table('/.../.../.../meteodaten_saison.csv',  
    # Pfad zur Datei angeben in ''  
    header=...,          # Erste Zeile Spaltennamen? TRUE oder FALSE  
    na.strings='..',      # wie sind Fehlwerte kodiert? z.B. '-99', NA  
    sep='... '           # optional: wie sind Werte separiert? wenn durch  
    # tabs getrennt nicht notwendig,  
    # sonst z.B. ',' für .csv (Comma separated values)
```

Wie sehen eingelesene Daten aus?

- > *str(saison)* ... zeigt die Struktur des Objekts
- > *head(saison)* ... zeigt die ersten paar Zeilen
- > *tail(saison)* ... zeigt die letzten paar Zeilen
- > *summary(saison)* ... einige statistische Grunddaten
- > *saison[1:10,]* ... zeigt die ersten 10 Zeilen an

Auswahl von Elementen/Spalten/Zeilen

- > *saison[1,1]*
- > *saison[23:24, 'SPALTENNAME']*
- > ...

Euer Skript startet mit:

```
> saison <- read.table('/euer_r_verzeichnis/meteodaten_saison.csv',  
sep=",", header=T) # 'C:/...' unter Windows  
  
> head(saison) # zeigt erste Zeilen der geladenen Daten an  
  
> str(saison) # zeigt an, ob Daten korrekt als numerisch gelesen wurden
```

Skript speichern nicht vergessen!

Auf korrekte GROSS- und klein-Schreibung achten!

Programmierempfehlungen

Kommentare, falls ihr nicht mit “Notebooks/Markdown” arbeitet

```
> # Zeitreihe der Sommertemperatur in Genf, geglättet mit 31-jährigem laufenden  
Mittelwert  
> plot(running.mean(saison[saison[,2]=='Sommer JJA',3],31),ty='l,col='red)
```

Selbsterklärende Objektnamen

```
JJA_temp_genf_31yr-smooth <- running.mean(saison[saison[,2]=='Sommer  
JJA',3],31)  
> plot(JJA_temp_genf_31yr-smooth ,ty='l,col='red)
```

Struktur mit Zeileneinschüben, Leerzeichen, etc.:

```
> for ( i in 3:5 ) {  
    print ( i + 1 )  
}
```

u^b

b
**UNIVERSITÄT
BERN**

OESCHGER CENTRE
CLIMATE CHANGE RESEARCH

ÜBUNGEN 1

R-Übungen 1

ACHTUNG: wenn + statt > am Zeilenanfang steht, ist eine Funktion nicht beendet, d.h. ',),]' oder } fehlen



b
UNIVERSITÄT
BERN

OESCHGER CENTRE
CLIMATE CHANGE RESEARCH

1.1) Vektoren: Überlegt euch die erwarteten Lösungen vor dem Eintippen!

```
x <- c(5,2,1,4)  
xx <- c(1,10,15,18)  
y <- rep(1,5)  
z <- c(TRUE, FALSE, TRUE, TRUE)
```

- a) sum (x)
range(x)
length(x)
sum(x)
- b) c(x,y,13)
- c) x[4] * y[2]
xx[2:4] + x[1:3]
- d) xx <= 12
xx [xx <=12]

- e) plot(x,xx)
plot(x[z],xx[z])

1.2) Zahlenfolgen: Erzeugt mit den *rep* und *seq* Funktionen folgende Zahlenfolgen:

- a) 1 2 3 4 5 6 7 8 9
- b) 'm' 'w' 'm' 'w' 'm' 'w'
- c) 1 2 3 4 1 2 3 4 1 2 3 4
- d) 1 2 2 3 3 3 4 4 4 4

1.3) Lest die Datei
"meteodaten_saison.csv" in R ein:

```
> saison <- read.table('/pfad/.../  
meteodaten_saison.csv', sep=',',  
header=TRUE)
```

Überprüft, ob der Import korrekt verlief?

Welche Spalten sind chr, int, num, ...?

```
> head(), str(), summary(), tail(), class()...
```