GOETHE UNIVERSITÄT FRANKFURT AM MAIN

Dr. Christian Czymara

FORSCHUNGSPRAKTIKUM I UND II: LÄNGSSCHNITTDATENANALYSE IN R

Welcome & introduction to R
session i

AGENDA

- Welcome
- Structure of the seminar
- Term paper
- Software
- Introduction to R

GENERAL INFORMATION

- Thursdays, 14:15 in room PEG 2.G *116*
- Material available at: czymara.com/FoPra
- Join this course's mailing list for communication: https://czymara.com/FoPra-mail

OFFICE HOURS

- After appointment
- Office: 3.G152 (PEG)
- Contact me at cc@soz.uni-frankfurt.de
- Do not hesitate to write me if you have questions, comments, doubts, criticism etc.

OVERVIEW

GOAL OF SEMINAR

- The goal is that students learn how to analyze longitudinal (mostly panel) data with R
- This implies continuous work throughout the semester

THIS MEANS YOU SHOULD HAVE…

- Interest in quantitative social research
- Good working knowledge of descriptive and inductive statistics (i.e., linear and logistic regression)
- Some knowledge of R or another statistics software / language
- I will introduce both in this seminar, but focus will be on advanced methods

WHAT THIS COURSE WILL OFFER

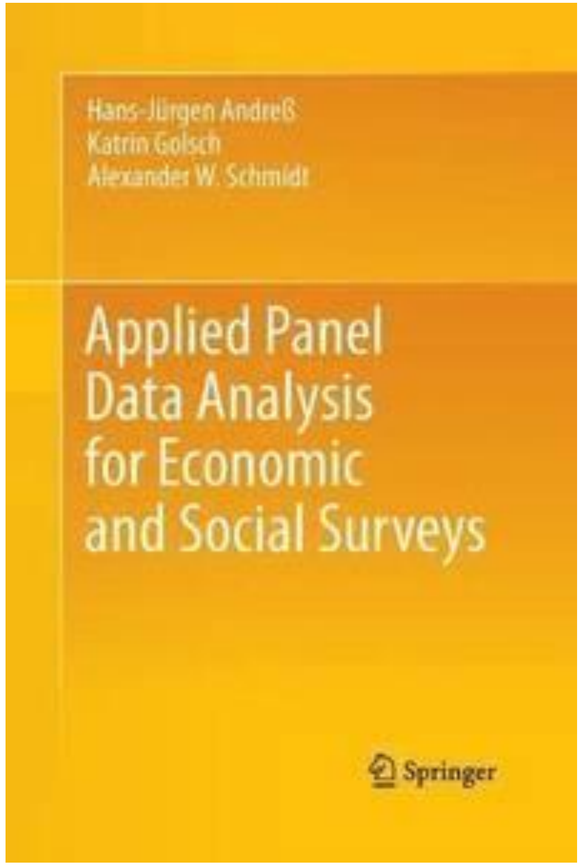
- An introduction to the analysis of different types of longitudinal data
- … and why it may help to tackle the notoriously difficult issue of causality
- The means to conduct your own research
- Hands-on application of methods in tutorials

WHAT THIS COURSE WILL NOT OFFER

- Discussion of substantive theories
- In-depth understanding of mathematical foundation of methods
- Course is less suited as a general introduction into empirical research

LITERATURE

- See syllabus for literature on individual sessions
- Literature on methods
- General textbook for this course: Andreß, Golsch & Schmidt. *Applied panel data analysis for economic and social surveys*. Springer Science & Business Media, 2014



COURSE STRUCTURE

STRUCTURE

- In General, most sessions consist of two parts
  - Lecture part
    - Statistical background
    - Methodology
    - Examples
  - Exercise part
    - Putting things into practice
    - Preparing data
    - Applying method

STRUCTURE: LECTURES

- Lectures will be on the date of each session (starting today)
- I will (try to) upload slides on GitHub beforehand

STRUCTURE: TUTORIALS

- There is a tutorial for each session including
  - An research question consisting of various steps
  - A (more or less prepared) data set
- Your task is to write code to answer the research question
- In many cases, there is more than one correct solution
- I will post the tutorial to GitHub on Thursdays
- You will have one week to work on the tutorial yourselves
- We will discuss the solution in the session of the *following* week, and will upload them afterwards

TERM PAPER

TERM PAPER

- Analysis of longitudinal data on a research question of your interest
- You may chose from any topic (… for which suitable data are available)
- The theoretical part should be based on a *DAG* (see next session)
- *The focus of the paper should be the analysis*

TERM PAPER

- ~15 to 20 pages, incl. tables, graphs, references etc.
- Also add your code so that I can understand and reproduce what you've done
- Send via e-mail to cc@soz.uni-frankfurt.de (no print needed)
- **DEADLINE IS 01 September 2022!**
- Hand in as PDF and R code via e-mail to cc@soz.uni-frankfurt.de

DAG?

- DAG means Directed Acyclic Graph
- DAG does not substitute but complement your theory
- We will discuss DAGs in session four
- ***Your DAG does not have to be rocket science, but not having any DAG in your term paper at all will also result in a downgrade of half a grade***

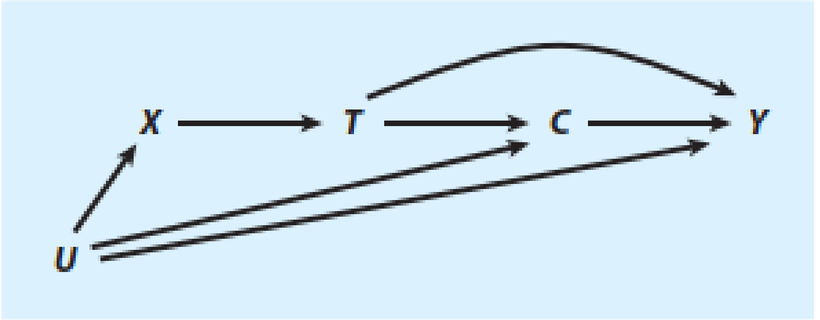


Figure 1

A directed acyclic graph (DAG).

Elwert & Winship 2014: 34

SOFTWARE

R

- You will need R for all tutorials and the term paper
- To work with R, install on your computers
  - R: https://cloud.r-project.org/
  - RStudio: https://www.rstudio.com/products/rstudio/download/

GITHUB

- Material will be uploaded on GitHub
- Link: czymara.com/FoPra
- You can download files without having an account
- For advanced users: Feel free to make an account and download GitHub Desktop to synchronize files every week

OFFICE

- You will probably need to work with *.doc* or *.xls* files
- Get Office 365 ProPlus at: https://www.rz.uni-frankfurt.de/55581873/Informationen_zur_privaten_Nutzung
- Alternatively, you may check the open source software Libre Office: https://www.libreoffice.org/

ONLINE TUTORIUM

- If you have issues with R or general questions, Subin Chang will assist you
- Contact her at s1786518@stud.uni-frankfurt.de

BENEFITS OF LONGITUDINAL DATA ANALYSIS

OPPORTUNITIES

- Monitor social change (e. g. does poverty in a country increase?)
- Examine change at the individual level instead of aggregate trends → May circumvent ecologic fallacy (inference on the individual level based on aggregate relationships)

PROBLEMS OF CROSS-SECTIONAL DATA

- Researchers normally want to make *causal* statements about the association of two variables
- Causal means that the correlation of x and y is not driven by another variable z (spurious correlation)
- The best way to establish this are experiments

→ Randomly assigning individuals in treatment and control group

→ All z are equally distributed between both groups

PROBLEMS OF CROSS-SECTIONAL DATA

- However, experiments often not feasible
- Observational studies thus adjust for z by statistical *controlling* after data collection
- However, z is often not observed in the data at hand
- As a result, estimates based on cross-sectional data are often plagued by *omitted variable bias*, which is bad

SOLUTIONS OF LONGITUDINAL DATA

- With longitudinal data you can deal control *even for (some) unobserved characteristics*!
- Some ≙ all time constant characteristics
- This is because individuals act as "their own controls"
- This does not ensure causality
- But it at least comes closer

THE POWER OF PANEL DATA

"It is hard to overstate the gain in identifying power provided by the beautifully simple method of [Fixed Effects] estimation over standard cross-sectional estimators"

- Gangl 2010: 34

ESTABLISHING CAUSALITY WITH OBSERVATIONAL DATA

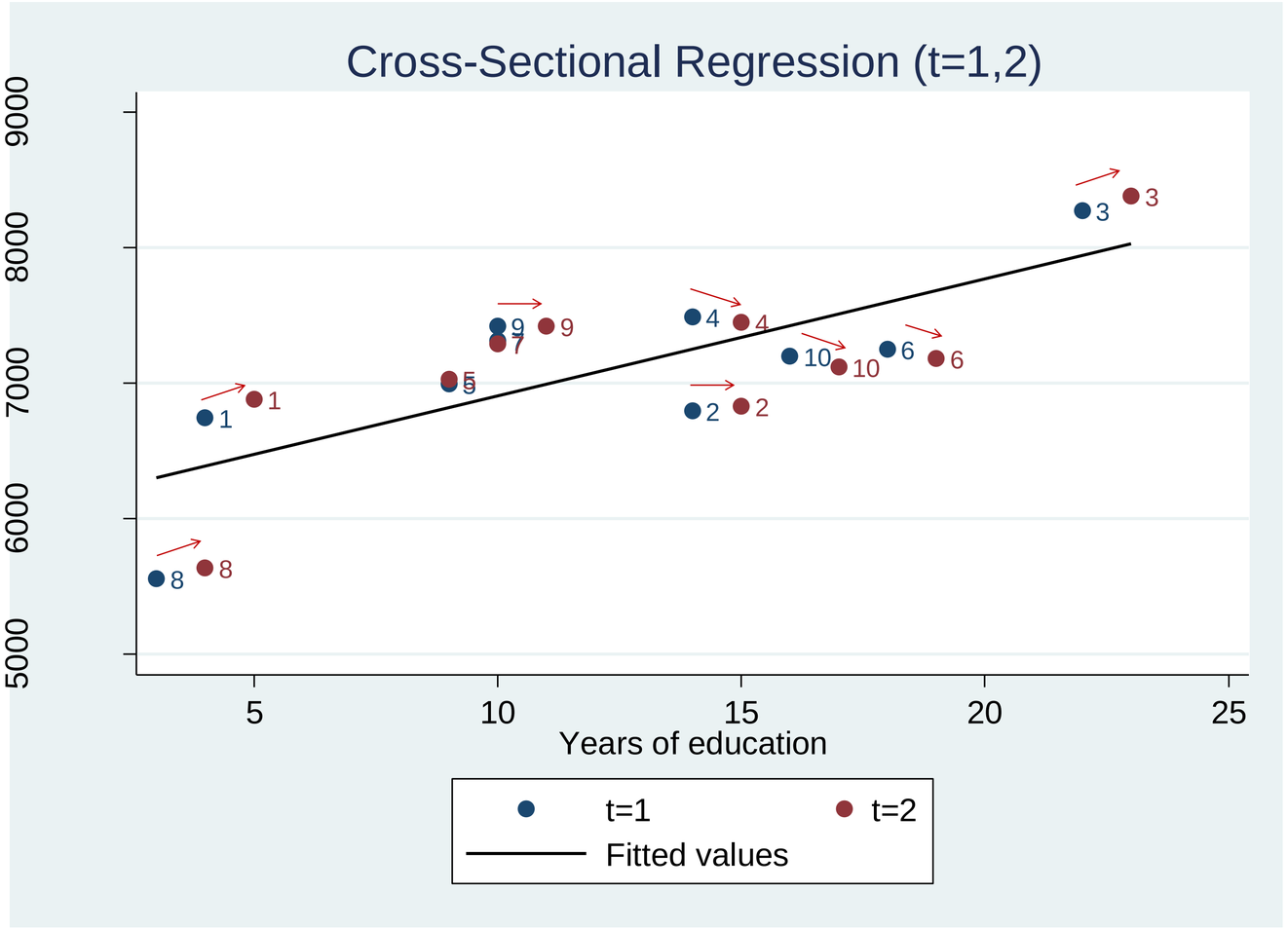
- There are other methods which mimic the logic of a randomized experiment drawing upon observational data
  - Natural experiments
  - Difference in Differences
  - Regression discontinuity
  - Etc.
- These methods have a temporal aspect, but not all require panel data
- See session on natural experiments

EXAMPLE

QUESTION

- We are interested in the returns to education (“*How much does additional education financially pay off?*”)
- y = income
- x = years of education
- Both measured at two time points (t)

$$y_{it} = \beta_0 + \beta_1 x_{1it} + \varepsilon_i$$



WHAT DID WE LEARN?

- On average, those with more years of education have higher income
- But…
- Additional education does not pay off equally for everyone
- More for those with lower levels of education
- Not really for those with medium-high levels

FINALLY

- Please participate in this survey:

  https://www.soscisurvey.de/longanalysisSS22/

- It will give me an overview on your interest, expectations and existing knowledge
- Moreover, you can analyze your own data in the first exercise and play around with it (all personal information and open-end answers will be removed)
- Join this course's mailing list for communication: https://czymara.com/FoPra-mail

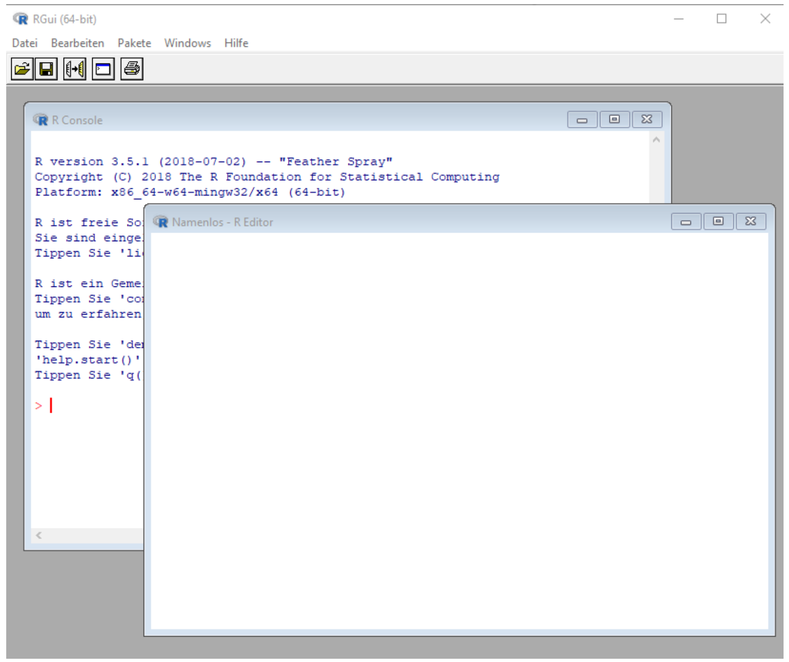
R

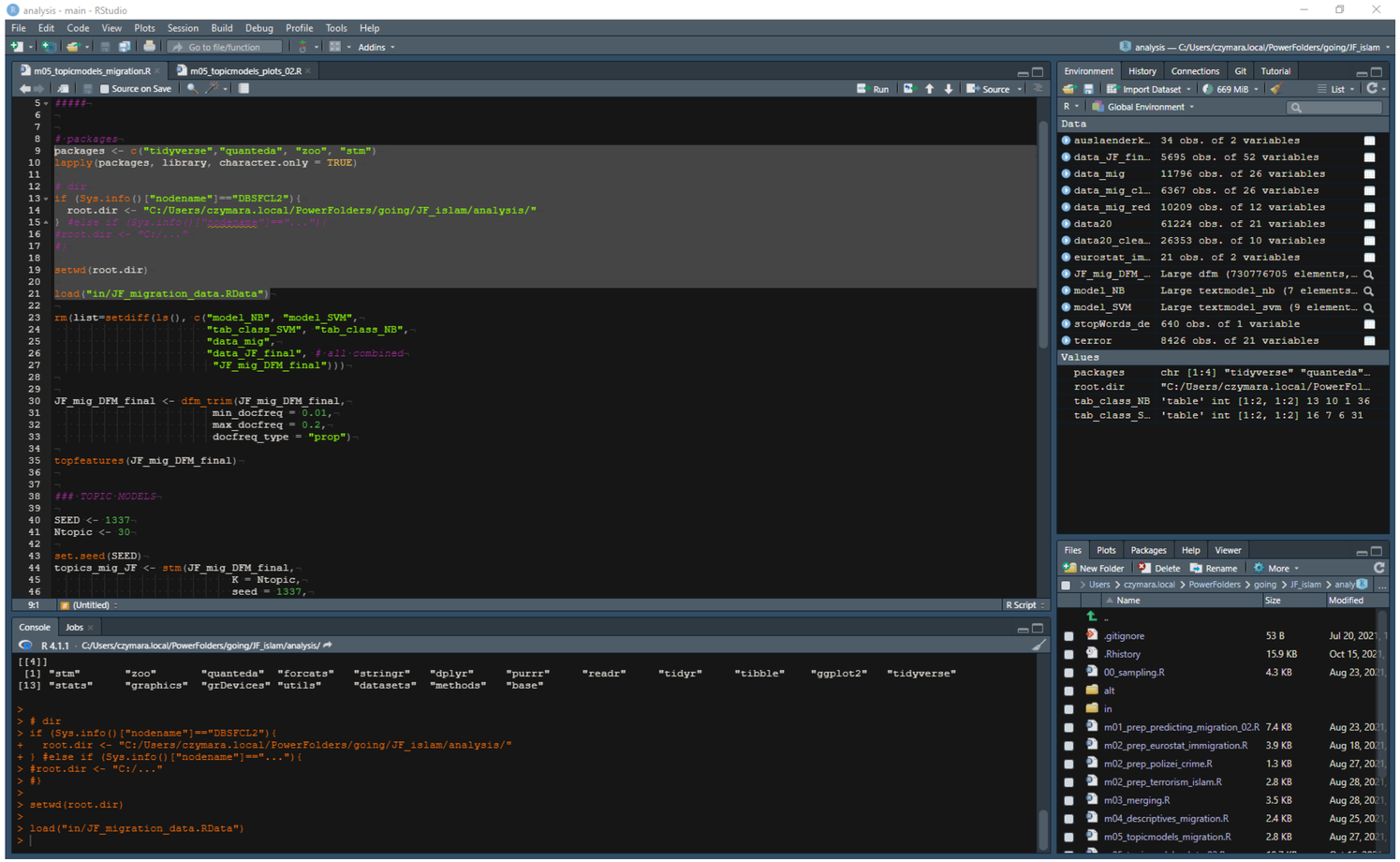
WHAT IS R?

R

- Why "R"? → *"R is an implementation of the S programming language"* (Wikipedia)
- R is a programming language for data analysis
- Rstudio is a so called Integrated Development Environment (IDE), making your work a lot easier
  - Writing and running R Code
  - Overview of stored objects
  - Projects containing multiple files
  - Git connection
  - Etc.

R VS. RSTUDIO


RGui (64-bit)
Datei Bearbeiten Pakete Windows Hilfe
R Console
R version 3.5.1 (2018-07-02) -- "Feather Spray"
Copyright (C) 2018 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)
R ist freie So
Sie sind einge
Tippen Sie 'li
R ist ein Geme
Tippen Sie 'co
um zu erfahren
Tippen Sie 'de
'help.start()'
Tippen Sie 'q(
Namenlos - R Editor


analysis - main - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
m05_topicmodels_migration.R
m05_topicmodels_plots_02.R
Source on Save
Run
Source
#####
# packages
packages <- c("tidyverse","quanteda", "zoo", "stm")
lapply(packages, library, character.only = TRUE)
# dir
if (Sys.info()["nodename"]=="DBSFCL2"){
root.dir <- "C:/Users/czymara.local/PowerFolders/going/JF_islam/analysis/"
} #else if (Sys.info()["nodename"]=="..."){
#root.dir <- "C:/..."
#}
setwd(root.dir)
load("in/JF_migration_data.RData")
rm(list=setdiff(ls(), c("model_NB", "model_SVM",
"tab_class_SVM", "tab_class_NB",
"data_mig",
"data_JF_final", # all combined
"JF_mig_DFM_final")))
JF_mig_DFM_final <- dfm_trim(JF_mig_DFM_final,
min_docfreq = 0.01,
max_docfreq = 0.2,
docfreq_type = "prop")
topfeatures(JF_mig_DFM_final)
TOPIC MODELS
SEED <- 1337
Ntopic <- 30
set.seed(SEED)
topics_mig_JF <- stm(JF_mig_DFM_final,
K = Ntopic,
seed = 1337,
Console
Jobs
R 4.1.1 · C:/Users/czymara.local/PowerFolders/going/JF_islam/analysis/
[[4]]
[1] "stm" "zoo" "quanteda" "forcats" "stringr" "dplyr" "purrr" "readr" "tidyr" "tibble" "ggplot2" "tidyverse"
[13] "stats" "graphics" "grDevices" "utils" "datasets" "methods" "base"
> # dir
> if (Sys.info()["nodename"]=="DBSFCL2"){
+ root.dir <- "C:/Users/czymara.local/PowerFolders/going/JF_islam/analysis/"
+ } #else if (Sys.info()["nodename"]=="..."){
> #root.dir <- "C:/..."
> #}
> setwd(root.dir)
> load("in/JF_migration_data.RData")
Environment History Connections Git Tutorial
Import Dataset
669 MiB
Global Environment
Data
auslaenderk… 34 obs. of 2 variables
data_JF_fin… 5695 obs. of 52 variables
data_mig 11796 obs. of 26 variables
data_mig_cl… 6367 obs. of 26 variables
data_mig_red 10209 obs. of 12 variables
data20 61224 obs. of 21 variables
data20_clea… 26353 obs. of 10 variables
eurostat_im… 21 obs. of 2 variables
JF_mig_DFM_… Large dfm (730776705 elements,…
model_NB Large textmodel_nb (7 elements…
model_SVM Large textmodel_svm (9 element…
stopWords_de 640 obs. of 1 variable
terror 8426 obs. of 21 variables
Values
packages chr [1:4] "tidyverse" "quanteda"…
root.dir "C:/Users/czymara.local/PowerFol…
tab_class_NB 'table' int [1:2, 1:2] 13 10 1 36
tab_class_S… 'table' int [1:2, 1:2] 16 7 6 31
Files Plots Packages Help Viewer
New Folder Delete Rename More
Users › czymara.local › PowerFolders › going › JF_islam › analy…
Name Size Modified
.gitignore 53 B Jul 20, 2021
.Rhistory 15.9 KB Oct 15, 202…
00_sampling.R 4.3 KB Aug 23, 21…
alt
in
m01_prep_predicting_migration_02.R 7.4 KB Aug 23, 21…
m02_prep_eurostat_immigration.R 3.9 KB Aug 18, 21…
m02_prep_polizei_crime.R 1.3 KB Aug 27, 21…
m02_prep_terrorism_islam.R 2.8 KB Aug 28, 20…
m03_merging.R 3.5 KB Aug 28, 21…
m04_descriptives_migration.R 2.4 KB Aug 25, 21…
m05_topicmodels_migration.R 2.8 KB Aug 27, 21…

R BENEFITS

- Free and open source
- Large and very helpful community
- Plethora of user-written packages on basically everything
- Very powerful tools for data manipulation and data visualization
- In addition to analyzing data, you can write programs, websites, books, and much more with R (and R Markdown)
- … and integrate with other languages

R BASICS

MATH OPERATORS

- Addition (+), subtraction (–), multiplication (*), Division (/), exponentiation (^), exponential (`exp()`), logarithm (`log()`), and basically everything else
- For example: `3+2` will return `5`
- Operators can also be combined: `(3+5)/(4*2)` will return `1`
- But we wouldn't need R for that…

OBJECTS

- Crucially, R allows to store information (e. g., numbers or text) in an object
- To create an object, use the *assignment operator*: <-
  - E. g.: `result_1 <- 3+5`
  - `result_2 <- 4*2`
- These objects can be recalled: `result_3 <- result_1 / result_2` will again return `1`
- Note that object names can be anything, better avoid generic names such as `result_1, result_2, result_3` ☺

LOGICAL OPERATORS

- Tests whether something is `True` or `False`
- For example: `result_1 == result_2` will be `True` (because 8=8)
- `result_1 == result_3` will be `False` (because 8≠1)
- But `result_1 != result_3` will again be `True`
- Similarly, `result_1 > result_3` will be `True`

LOGICAL OPERATORS

- Logical operators can be combined using `&` (“and”) and `|` (“or”)
- Example 1: `8 == 8 & 8 > 1`
- Example 2: `8 == 8 & 8 == 1`
- Example 3: `8 == 8 | 8 == 1`
- Example 3: `8 != 8 | 8 > 1`
- This will often be relevant when you create new variables or define your sample of analysis (e. g.: relevant age range from 18 to 65 and only natives)

VECTORS

- Storing a single value (as in object `result_1`) is not very interesting in most cases
- If you want to store many values simultaneously, you work with vectors
- For example: `variable_num <- c(8, 8, 1)`
- Importantly, vectors don't have to be numerical but can also be strings ("text") `variable_char <- c("a", "b", "c")`
- We can also combine the numerical objects we created before: `variable_num <- c(result_1, result_2, result_3)`

INDEXING

- What if we want to recall a certain value of `variable_num`?
- Use indexing, which is done via `[]`
- `vector[elements]`
- Let's say we want to access the first element of `variable_num`
  - → `variable_num[1]` will return `8`
  - → `Variable_num[2]` will return `8`
  - → `variable_num[3]` will return `1`
- You can also nest these: What will `variable_num[variable_num[3]]` return?

VARIABLE TYPES

3.5 IMPORTANT TYPES OF VARIABLES

1. *Logical*: Binary variable with values `True` and `False` → `class(True)`
2. *Character* (string): Text (including symbols and numbers that are treated as text) → `class("this is 1 character")`
3. *Numeric*: Numbers for mathematical operations → `class(123)`
4. *Factor*: Categories → `class(factor(c("male", "female")))`

- Most important for regression analysis: *Factor* and *numeric* correspond to *categorical* and *continuous* variables
- `NA` is an value that means *missing value* ("not available"), important if you work with real-world data

DATA FRAMES

VARIABLES AND DATA FRAMES

- So far, we learned about single variables (logical, numerical, character, factor)
- However, in most cases we won't analyze a bunch of unrelated variables but rather several variables of one (or more) data sets
- Data sets (called data frames in R) are a collection of variables that are organized in a two-dimensional table
  - Column: variable
  - Row: observations
  - Cells: values
- You can turn variables into a data frame using `as.data.frame()`

EXAMPLE OF DATA FRAME

Filter

	trstplc Trust in the police	trstprl Trust in country's parliament	trstlgl Trust in the legal system	trstplt Trust in politicians	trstprt Trust in political parties	trstep Trust in the European Parliament	trstun Trust in the United Nations	discrim	blgetmg Belong to minorit
1	10	9	10	0	*NA*	*NA*	9	no	2
2	5	0	8	0	*NA*	0	6	yes	2
3	8	6	4	2	*NA*	7	*NA*	no	*NA*
4	9	8	10	4	*NA*	7	8	no	2
5	4	6	7	4	*NA*	4	5	no	2
6	6	0	5	0	*NA*	2	*NA*	yes	2
7	6	6	6	5	*NA*	5	*NA*	no	2
8	7	9	7	4	*NA*	6	6	no	2
9	8	5	7	3	*NA*	2	0	no	2
10	5	0	3	5	*NA*	*NA*	*NA*	no	1
11	7	2	2	0	*NA*	0	0	no	*NA*
12	3	5	5	3	*NA*	3	3	no	2
13	2	0	0	0	*NA*	0	0	no	*NA*
14	8	5	10	5	*NA*	5	5	no	*NA*
15	7	8	5	5	*NA*	5	8	no	*NA*
16	4	6	6	4	*NA*	6	5	no	2
17	8	9	9	6	*NA*	7	6	no	2
18	6	5	8	5	*NA*	6	5	no	2
19	8	5	7	3	*NA*	2	2	no	2

HOW TO ACCESS A VARIABLE WITHIN A DATA FRAME

- Use the `$` operator
- For example, to access the variable `discrim` in the data frame `ESS`, type `ESS$discrim`

```
> ESS$discrim
  [1] no   yes  no   no   no   yes  no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   n
 [34] no   no   no   no   no   no   no   no   no   no   no   no   <NA> no   no   no   no   no   yes  no   no   no   no   no   no   no   no   no   n
 [67] no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   n
[100] no   no   <NA> <NA> no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   yes  no   no   no   no   no   no   no   n
[133] no   no   no   no   <NA> no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   <NA> no   n
[166] no   no   no   <NA> no   no   yes  no   no   no   no   yes  no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   y
[199] no   no   no   no   yes  no   no   no   no   no   no   no   no   no   no   no   no   no   no   yes  no   no   yes  no   no   no   no   no   <NA> n
[232] yes  no   no   no   no   yes  no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   n
[265] no   no   no   no   yes  no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   n
[298] yes  no   no   yes  no   no   no   no   no   no   no   no   no   no   no   no   yes  no   no   no   no   no   no   no   no   no   no   no   n
[331] no   no   no   no   no   yes  no   no   no   no   no   no   no   no   no   no   <NA> yes  no   yes  yes  no   no   no   no   no   no   no   n
[364] no   no   no   no   no   no   no   yes  no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   n
[397] no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   yes  no   no   no   no   no   no   no   no   no   no   no   no   n
[430] no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   n
[463] no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   <NA> no   no   no   no   no   no   no   no   n
[496] no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   <NA> <NA> no   no   no   no   no   n
[529] no   no   no   no   yes  no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   no   yes  no   yes  no   no   <NA> n
```

- Better overview:

```
> table(ESS$discrim)

    no    yes
361710  25988
>
```

RECODING VARIABLES

- Accessing existing variables is nice, but often we need to create *new* variables based on old ones
- Example: What year did an immigrant arrive in a country? → create new variable `migr_year` based on `livecnta` ("What year you first came to live in country") and `inwyye` (year of interview)
- `ESS$migr_year <- ESS$inwyye - ESS$livecnta`

INDEXING

- Like indexing for one-dimensional vectors, there is also indexing for the two-dimensional data frames

→`dataframe[rows, columns]`

- The first value refers to the *rows* you want to access
- The second value refers to the *columns* you want to access
- So `dataframe[1, 1]` will show the first observation's value of the first variable
- If we are interested in all values of the first observation (row), we can use `dataframe[1, ]`
- If we are interested in all values of the first variable, we can use `dataframe[, 1]`

SUBSETTING DATA

- Subsetting means reducing the data, either drop columns (variables) or rows (observations)
- Example code for keeping variables:

```
    modelvars <- c("trstplc", "discrim", "blgetmg",
"livecnty_comb1", "migr", "continent", "educ")
    ESS_reduc <- ESS[modelvars]
```

- Example code for keeping observations (here: listwise deletion):

```
    ESS_reduc <- ESS__reduc[complete.cases(ESS_reduc[modelvars]), ]
```

FUNCTIONS

WHAT ARE FUNCTIONS?

- You will mostly work with functions in R
- Functions (often) require an input (often between `()`)and will create an output
- Example: `mean()` function: `mean(variable_num)` will return `5.666667` (the mean of 8, 8 and 1)
- Or `range(variable_num)`
- To access R's help, type `?function` (e.g. `?mean`)
- Even better: Google

PACKAGES

- There are several functions included in “base R” (e. g. `mean()`)
- But a lot of the things that make R *really* interesting have to be loaded into your working environment as *packages*
- Packages are a collection of (user-written) functions
- To install packages, use the `install.packages()` function
- You have to *install* a package only once, but you will have to *load* it every time you want to use it
- To load a package, use `library()`


tidyverse
www.rstudio.com

TIDYVERSE

- "The tidyverse is an opinionated collection of R packages designed for data science. All packages share an underlying design philosophy, grammar, and data structures." (https://www.tidyverse.org/)
- The goal of the Tidyverse packages is to make data "tidy"
- Tidy is here defined as
  - Variables in columns
  - Observations in rows
  - Values in cells
- This is how we organized our data frame before

UNTIDY DATA: EUROSTAT

First instance decisions on applications by citizenship, age and sex - annual aggregated data (rounded)
(online data code: MIGR_ASYDCFSTA)
Source of data: Eurostat

Settings: Default

Table | Line | Bar | Map

GEO	SEX	TIME	2017	2018	2019	2020
European Union - 27 countries (from 2020)	Total		504 800	345 575	334 805	309 185
European Union - 27 countries (from 2020)	Males		373 780	247 975	235 635	206 075
European Union - 27 countries (from 2020)	Females		130 680	97 550	99 135	103 085
European Union - 28 countries (2013-2020)	Total		524 070	364 470	348 335	:
European Union - 28 countries (2013-2020)	Males		386 765	260 650	244 905	:
European Union - 28 countries (2013-2020)	Females		136 950	103 760	103 390	:
Belgium	Total		11 460	9 340	10 640	10 650
Belgium	Males		7 795	6 430	7 285	7 270
Belgium	Females		3 665	2 915	3 355	3 380
Bulgaria	Total		3 045	1 370	850	1 375
Bulgaria	Males		2 400	1 190	750	1 300
Bulgaria	Females		640	180	100	70
Czechia	Total		1 045	1 230	1 255	855
Czechia	Males		750	850	935	630
Czechia	Females		290	380	320	225
Denmark	Total		4 510	1 315	1 455	765
Denmark	Males		3 130	845	875	450
Denmark	Females		1 380	470	580	310
Germany (until 1990 former territory of the FRG)	Total		262 575	103 175	83 855	66 120
Germany (until 1990 former territory of the FRG)	Males		193 815	68 315	52 550	42 660
Germany (until 1990 former territory of the FRG)	Females		68 430	34 815	31 285	23 445
Estonia	Total		60	55	45	45
Estonia	Males		35	40	35	25
Estonia	Females		25	15	10	20
Ireland	Total					

WHY SHOULD DATA BE TIDY?

- Easier to read and process
- Standardized workflows of many functions
- A lot of possibilities to manipulate tidy data with the Tidyverse

(SOME) IMPORTANT FUNCTIONS OF THE TIDYVERSE

DPLYR()

- One of the most useful packages of the Tidyverse is `dplyr()`
- It includes
  - `filter()`: Filters observations, e. g.: `filter(ESS, discrim == "yes")`
  - `mutate()`: Create variables, e. g.: `ESS <- mutate(ESS, migr_year = inwyye - livecnta)`
  - `rename()`: Changes name of variable, e. g.: `rename(ESS, discrim = dscrgrp)`
  - `summarize()`: Get some aggregate statistic (example later)
  - …
- `mutate()` most potent when combined `ifelse()` to make conditional statements
- Idea: `ifelse(logical test, value if TRUE, value if FALSE)`
- Example: `ifelse(1 == 1, "This is TRUE", "This is FALSE")`
- Or: `mutate(ESS, discrim = ifelse(dscrgrp == 1, "yes", "no"))`

DEFINE MIGRATION BACKGROUND USING MUTATE()

```
mutate(ESS, migr =
    ifelse(brncntr == 1 & mocntr == 1 & facntr
==  1, "native",
        ifelse(brncntr == 2,
        "first gen immigrant",
            ifelse(brncntr == 1
            & (mocntr == 2 | facntr == 2),
            "second gen immigrant",
    NA))))

# 1=yes, 2=no
```

PIPING

%>%

- Let's say we want to get the logarithm of 4, and round it to the first decimal:
  - `x <- log(4)`
  - `round(x, 1)`
- To have less code, R allows *nesting* functions: `round(log(4), 1)`
- But that's hard to read (from inside out), especially when nesting many functions
- Piping allows to read from start to end:

```
4 %>%
 log() %>%
 round(1)
```

- Note: `magrittr` package must be loaded to use piping

COMBINING IT ALL

```
ESS_allwav %<>%
  mutate(migr = ifelse(brncntr == 1
                       & mocntr == 1
                       & facntr == 1, # 1=Yes
                       "native",
                     ifelse(brncntr == 2, # 2=no
                            "first gen immigrant",
                            ifelse(brncntr == 1
                                   & (mocntr == 2
                                   | facntr == 2),
                                   "second gen immigrant",
                                   NA)))) %>%
  mutate(unempl = ifelse(uempla == 1 | # actively looking for job
                           uempli == 1, # not actively looking for job, 1 = marked
                         "Unemployed",
                         ifelse(is.na(uempla) == T |
                                  is.na(uempli) == T,
                                NA,
                                "Not unemployed"))) %>%
  mutate(educ = ifelse(eisced <= 2,
                       "Low (<= ISCED 2)",
                       ifelse(eisced == 3 | eisced == 4,
                              "Medium low (ISCED 3)",
                              ifelse(eisced == 5,
                                     "Medium high (ISCED 4)",
                                     ifelse(eisced >= 6 & eisced <= 50,
                                            "High (>= ISCED 5)",
                                            NA))))) %>%
  mutate(minority = ifelse(blgetmg == 1,
                           "yes",
                           ifelse(blgetmg == 2,
                                  "no",
                                  NA))) %>%
  mutate(discrim = ifelse(dscrgrp == 1,
                          "yes",
                          ifelse(dscrgrp == 2,
                                 "no",
                                 NA)))
```

NOT COVERED HERE, BUT AWESOME

- A bazillion more functions
- RStudio projects (incl. version control using GitHub)
- R Markdown: Create documents, websites, presentations etc. that automatically update numbers and figures if you change the data (no manual copy & pasting anymore!)

LITERATURE

- The slides are inspired by this great intro from Fabio Votta
- Wickham & Grolemund (2017). R for Data Science. O'Reilly