Data Analytics in R
Session 8

Maria Kunevich

Homework feedback

- You can see the submitted reports here: https://maria-13.github.io/DataReporting/
- I'll provide my feedback directly to you as comments on your report as a .pdf file
- Any suggestions on how to improve the reports?

  Tip: consider what R code you need to display

Paper summary and work on your project

- Notes on the requirements for paper summary are on GitHub in the Assignments folder
- Feedback on paper selection and project work will be provided through our Miro board
- https://miro.com/welcomeonboard/SUw3RHpXWjBpUDF2V0dwTWhkOFVVUnlUTnZ4Qm1oVGtSTVN0SmNCUmJyODhydU9hUzA3VUpNZVZHRnNBenhqVHwzMDc0NDU3MzYzNjI1MDIzMjY2fDI=?share_link_id=164034942325

Paper summary and work on your project

- Main requirements for the project:
  - designing a project that is executed through **R** language
  - based on data analytics and data analysis
  - encompasses material covered in class: importing data into R, performing Exploratory Data analysis and data visualisation, performing data cleaning and wrangling, providing statistical analysis and hypothesis testing, possibly data modelling
  - communicating your results by creating a report

Assignments deadlines

Assignment	Date of assignment	Deadline (midnight 23:59)
HW1	22 Sept 2022	28 Sept 2022
HW2	29 Sept 2022	5 Oct 2022
HW3	6 Oct 2022	12 Oct 2022
HW4	13 Oct 2022	19 Oct 2022
HW5	20 Oct 2022	2 Nov 2022
Paper summary	20 Oct 2022	20 Nov 2022
HW6	3 Nov 2022	9 Nov 2022
HW7	10 Nov 2022	16 Nov 2022
HW8	17 Nov 2022	23 Nov 2022
HW9	24 Nov 2022	30 Nov 2022
HW10	1 Dec 2022	7 Dec 2022
Project	TBA	14 Dec 2022
Final Presentations		15 Dec 2022

Nov 15 - interim report

Homework assignment 6

- Revision of topics we've covered on DataCamp (each chapter = 1 point):
- https://app.datacamp.com/groups/data-analytics-in-r-db1ae4f4-62a1-4da2-b5d9-94616b38d5d0/assignments

    1. Course Introduction to Statistics in R, Chapter 1: Summary Statistics
    2. Course Introduction to Data Visualization with ggplot2, Chapter 1: Introduction
    3. Course Exploratory Data Analysis in R, Chapter 1: Exploring Categorical Data
    4. Course Reshaping Data with tidyr, Chapter 1: Tidy Data
    5. Course Data Manipulation with dplyr, Chapter 1: Transforming Data with dplyr

Extra: Course Reporting with R Markdown, Chapter 1: Getting started

Plan for today

- Data cleaning - R package from **tidyverse** ‘tidyr’
- Data wrangling - R package from **tidyverse** ‘dplyr’

Tidyr package

tidyr

Some possible problems with data and datasets:

- different data types
- duplicates
- missing values
- out of range values

Important: datasets are often not in the format we want them to be which makes it difficult to perform analysis

Tidyr package

tidyr

Overview

The goal of tidyr is to help you create **tidy data**. Tidy data is data where:

1. Every column is variable.
2. Every row is an observation.
3. Every cell is a single value.

Tidy data describes a standard way of storing data that is used wherever possible throughout the tidyverse. If you ensure that your data is tidy, you'll spend less time fighting with the tools and more time working on your analysis. Learn more about tidy data in `vignette("tidy-data")`.

Tidyr package

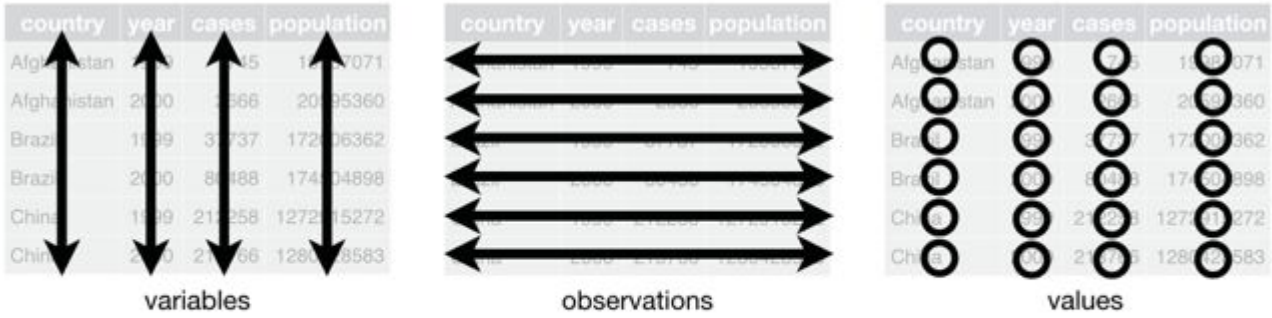

Figure 12.1: Following three rules makes a dataset tidy: variables are in columns, observations are in rows, and values are in cells.

Material is borrowed from **R for Data Science**: Tidy Data

Long and wide format data

Wide Format

Team	Points	Assists	Rebounds
A	88	12	22
B	91	17	28
C	99	24	30
D	94	28	31

Each value is unique in first column

Long Format

Team	Variable	Value
A	Points	88
A	Assists	12
A	Rebounds	22
B	Points	91
B	Assists	17
B	Rebounds	28
C	Points	99
C	Assists	24
C	Rebounds	30
D	Points	94
D	Assists	28
D	Rebounds	31

The values in the first column repeat

In the **wide** format, each value in the first column is **unique**

In the **long** format, the values in the first column **repeat**

Here this difference is visualised:
https://datacarpentry.org/R-ecology-lesson/img/tidyr-pivot_wider_longer.gif

Tidyr package

tidyr

Untidy data often results in two common problems:

1. One variable is spread across multiple columns

country	year	cases
Afghanistan	1999	745
Afghanistan	2000	2666
Brazil	1999	37737
Brazil	2000	80488
China	1999	212258
China	2000	213766

country	1999	2000
Afghanistan	745	2666
Brazil	37737	80488
China	212258	213766

table4

Tidyr package

tidyr

Untidy data often results in two common problems:

2. One observation might be scattered across multiple rows

country	year	key	value
Afghanistan	1999	cases	745
Afghanistan	1999	population	19987071
Afghanistan	2000	cases	2666
Afghanistan	2000	population	20595360
Brazil	1999	cases	37737
Brazil	1999	population	172006362
Brazil	2000	cases	80488
Brazil	2000	population	174504898
China	1999	cases	212258
China	1999	population	1272915272
China	2000	cases	213766
China	2000	population	1280428583

table2

country	year	cases	population
Afghanistan	1999	745	19987071
Afghanistan	2000	2666	20595360
Brazil	1999	37737	172006362
Brazil	2000	80488	174504898
China	1999	212258	1272915272
China	2000	213766	1280428583

Tidyr package

tidyr

Let’s examine the ‘tidyr’ cheat sheet

Four fundamental functions for data tidying that ‘tidyr’ provides:

- **pivot_longer()** makes ‘wide’ data longer
- **pivot_wider()** makes ‘long’ data wider
- **separate()** splits a single column into multiple columns
- **unite()** combines multiple columns into a single column

Tidyr package: reshaping data

tidyr

- turn columns into rows **pivot_longer()**
- turn rows into columns **pivot_wider()**
- turn a character column into multiple columns (`separate()`),
- turn multiple character columns into a single column (`unite()`)

pivot_longer

tidyr: **(cases, "year", "n", 2:4)**

Gather columns into rows.

pivot_wider

tidyr:: **(pollution, size, amount)**

Spread rows into columns.

tidyr::**separate(storms, date, c("y", "m", "d"))**

Separate one column into several.

tidyr::**unite(data, col, ..., sep)**

Unite several columns into one.

%>% Operator

Pipe operator in R originates from the “magrittr” package.

With ‘tidyverse’ packages there is no need to load “magrittr” explicitly, %>% operates automatically

Main advantages:

- ➔ enhances code clarity
- ➔ lowers improvement time
- ➔ makes code easier to maintain

As a result, the %>% operator provides a cleaner, more readable and efficient functions

A more advanced tutorial: Simplify Your Code with %>%

Dplyr package

dplyr

Overview

dplyr is a grammar of data manipulation, providing a consistent set of verbs that help you solve the most common data manipulation challenges:

- `mutate()` adds new variables that are functions of existing variables
- `select()` picks variables based on their names.
- `filter()` picks cases based on their values.
- `summarise()` reduces multiple values down to a single summary.
- `arrange()` changes the ordering of the rows.

These all combine naturally with `group_by()` which allows you to perform any operation "by group". You can learn more about them in `vignette("dplyr")`. As well as these

Dplyr package

dplyr

Resources: **R for Data Science**, 5 Data transformation

All verbs work similarly:

1. The first argument is a data frame
2. The subsequent arguments describe what to do with the data frame, using the variable names (without quotes)
3. The result is a new data frame

Let's practice with both packages in R and create an R Markdown file with examples and explanations.