Intro to the **Tidyverse** and importing/cleaning tables

Raphael Eisenhofer

2022_04_07

Outline for today:

- 1. Intro to the Tidyverse
- 2. Importing table data into R (excel, tsv, csv, etc.)
- 3. Cleaning data

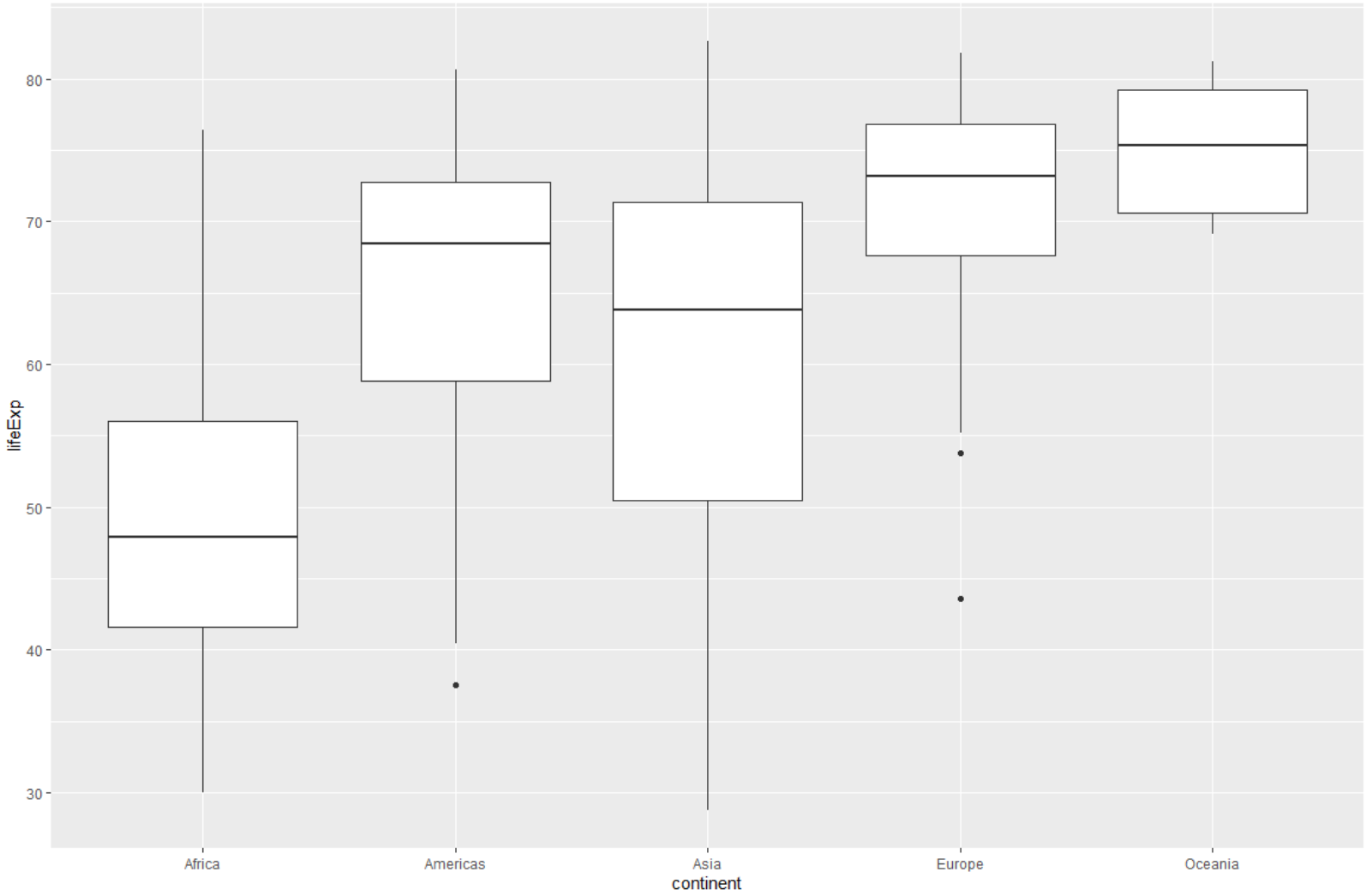
1. The Tidyverse

The Tidyverse

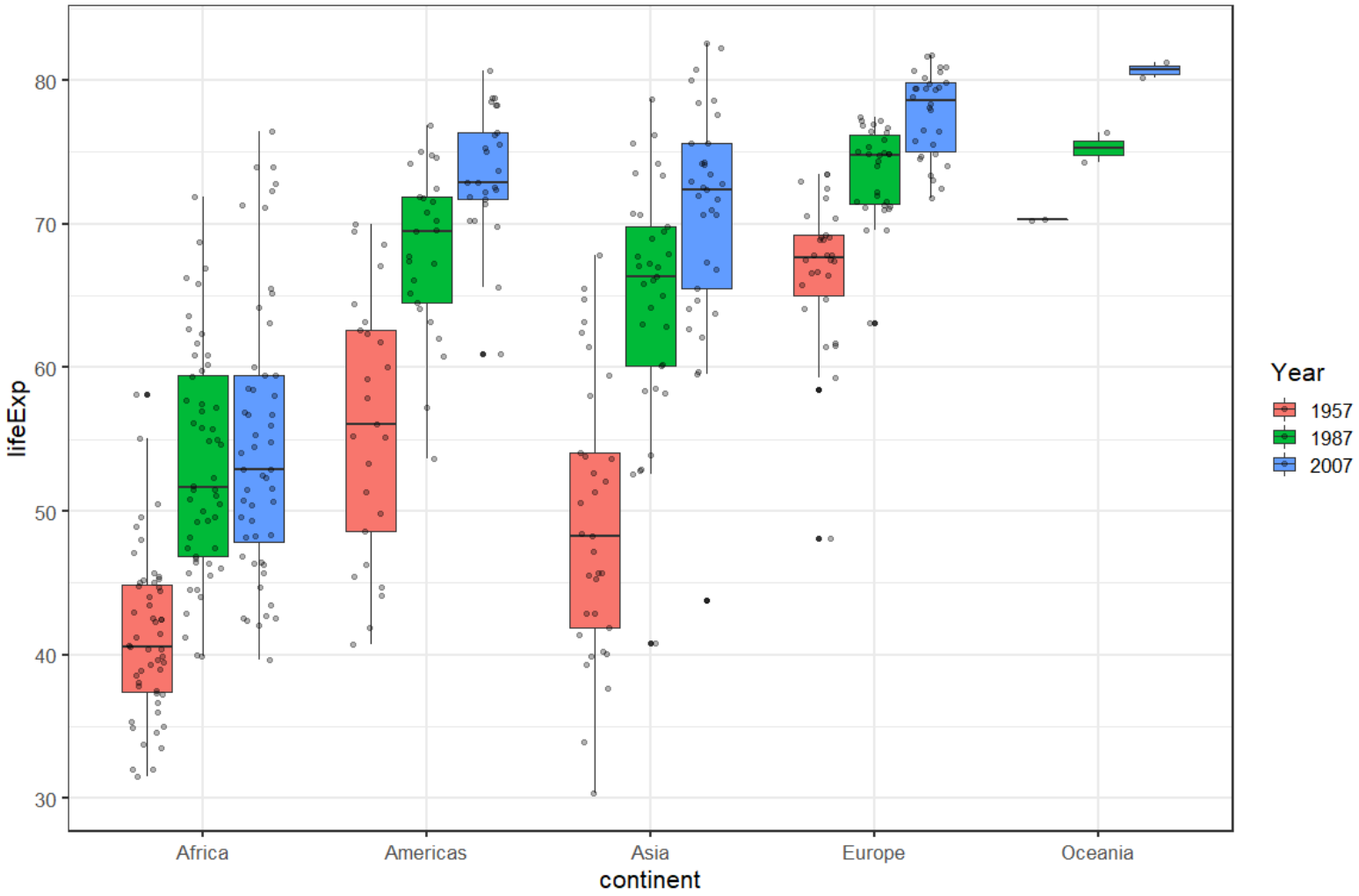
- Group of R packages designed for data science
- Common design/grammar structures
- More user friendly than base R

Wickham et al., (2019). Welcome to the Tidyverse.
Journal of Open Source Software, 4(43), 1686,
https://doi.org/10.21105/joss.01686

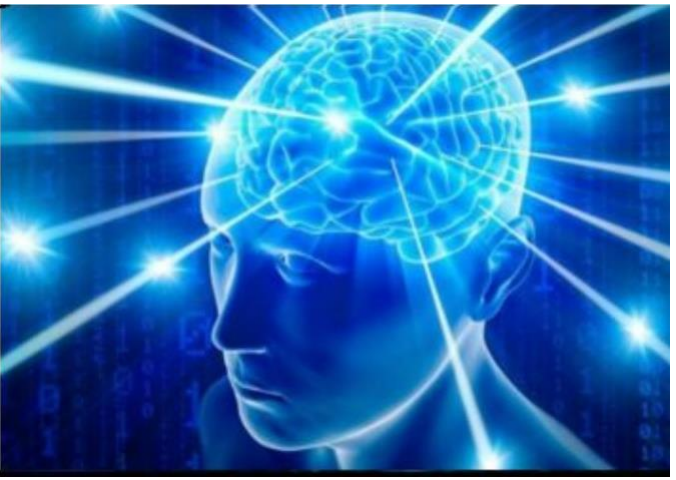
ggplot2


80
70
60
50
40
30
lifeExp
Africa
Americas
Asia
Europe
Oceania
continent

ggplot2


lifeExp
80
70
60
50
40
30
Africa
Americas
Asia
Europe
Oceania
continent
Year
1957
1987
2007

ggplot2



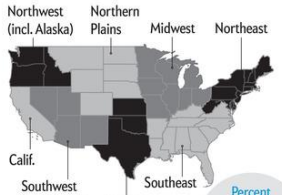
GRAPHIC SCIENCE

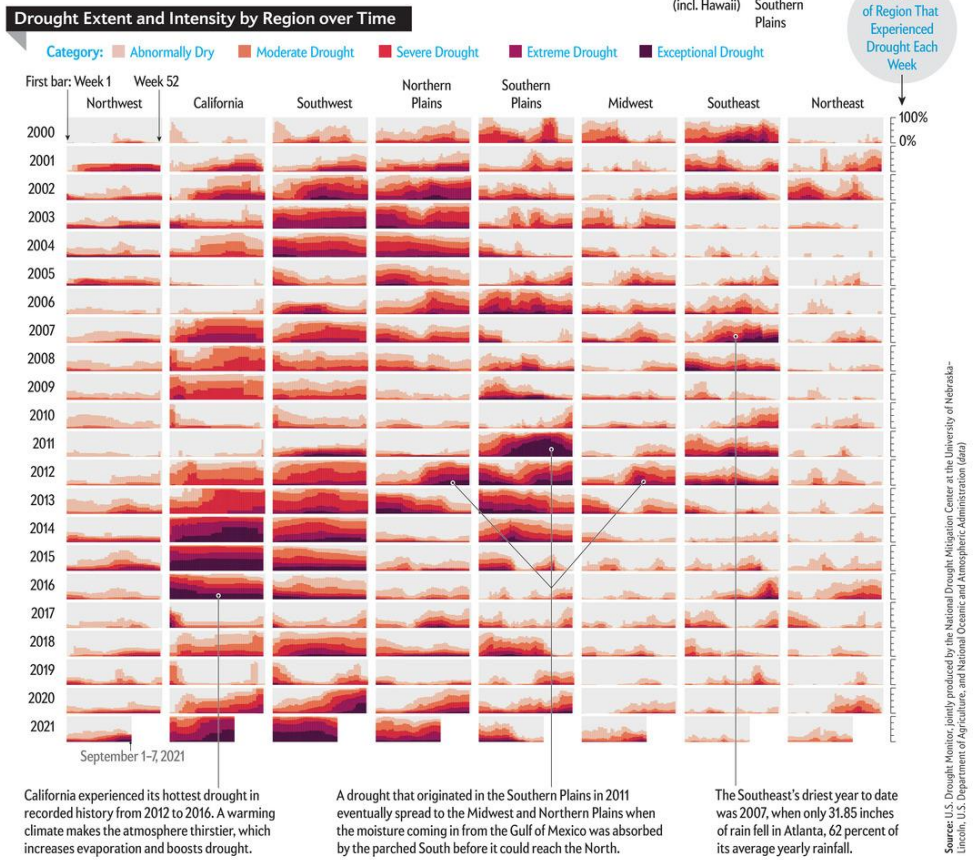
Text by Clara Moskowitz | Graphic by Cédric Scherer and Georgios Karamanis

Escalating Drought

Climate change is intensifying periods of extreme dryness, particularly in the U.S. West

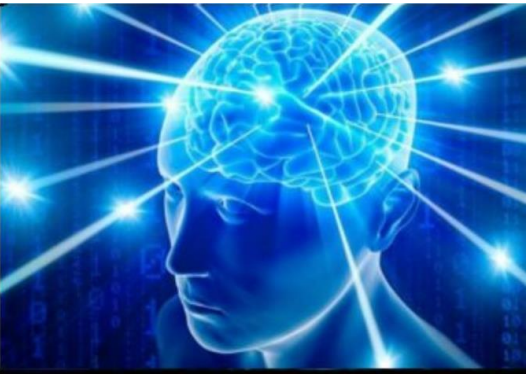
For more than 20 years the National Drought Mitigation Center (NDMC) has been monitoring dozens of indices of drought around the country, including satellite measurements of evaporation and color in vegetation, soil-moisture sensors, rainfall estimates, and river and streamflow levels. Although the agency's weekly assessments have identified periods of exceptional drought before, lately dryness has been ramping up. "The changing climate is definitely contributing to more natural disasters, drought being one of them," says Brian Fuchs, a climatologist who oversees the weekly report at the NDMC. "We're seeing more frequent and high-intensity episodes. This year some of these areas in the West have been in drought more than they have been without drought."

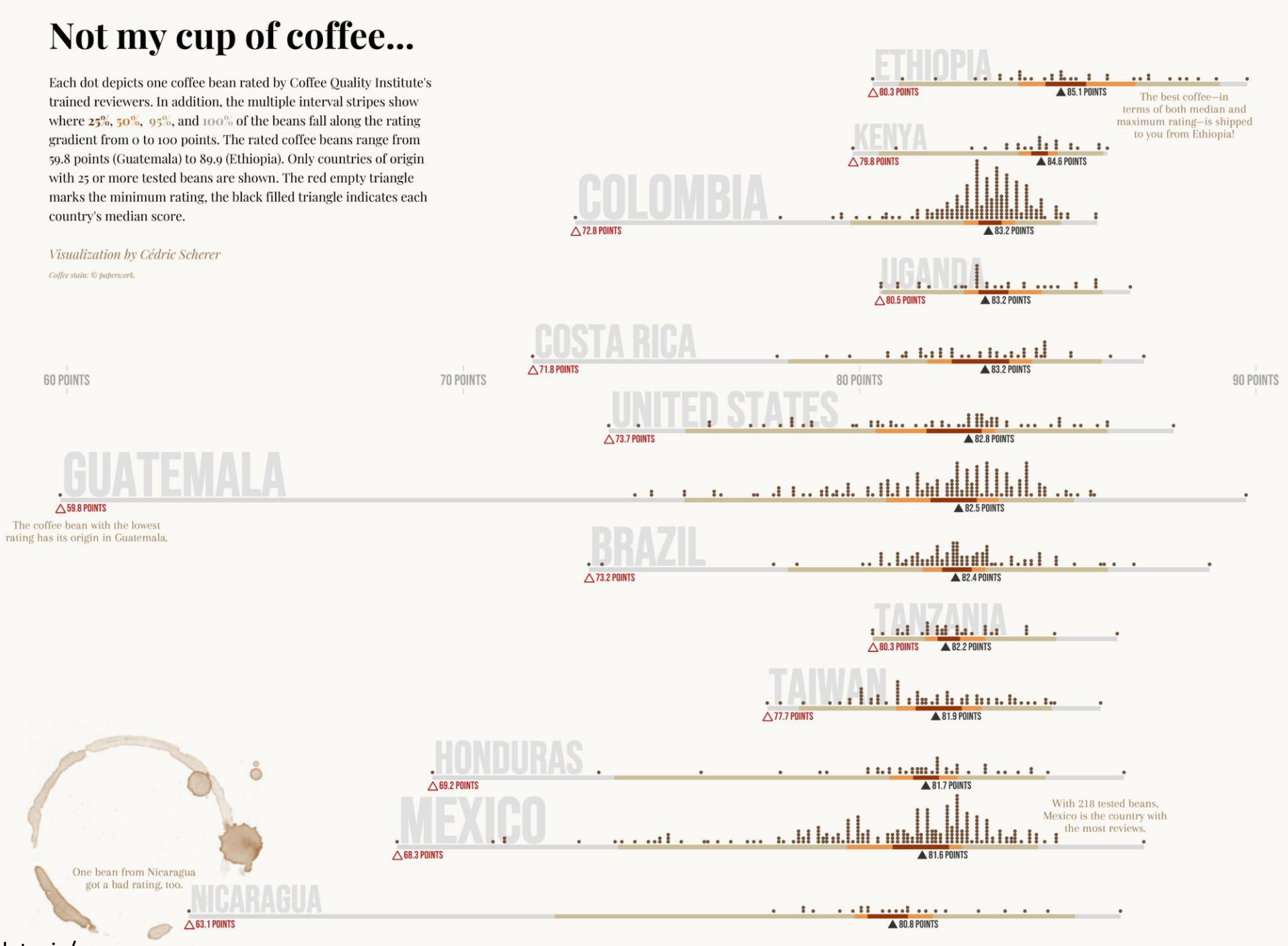




https://www.cedricscherer.com/top/dataviz/

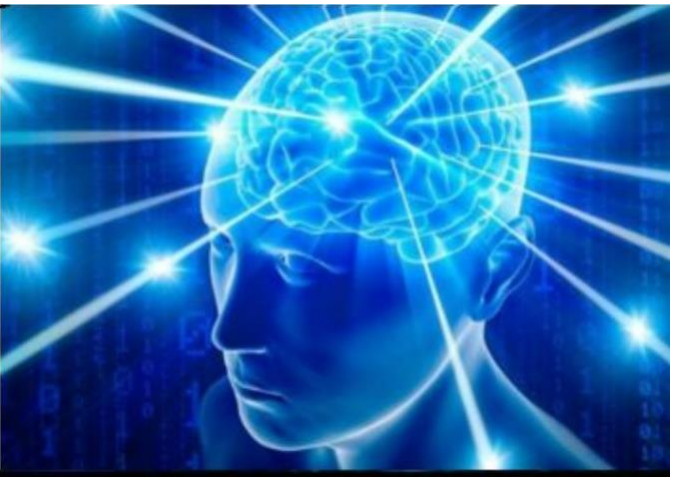
ggplot2

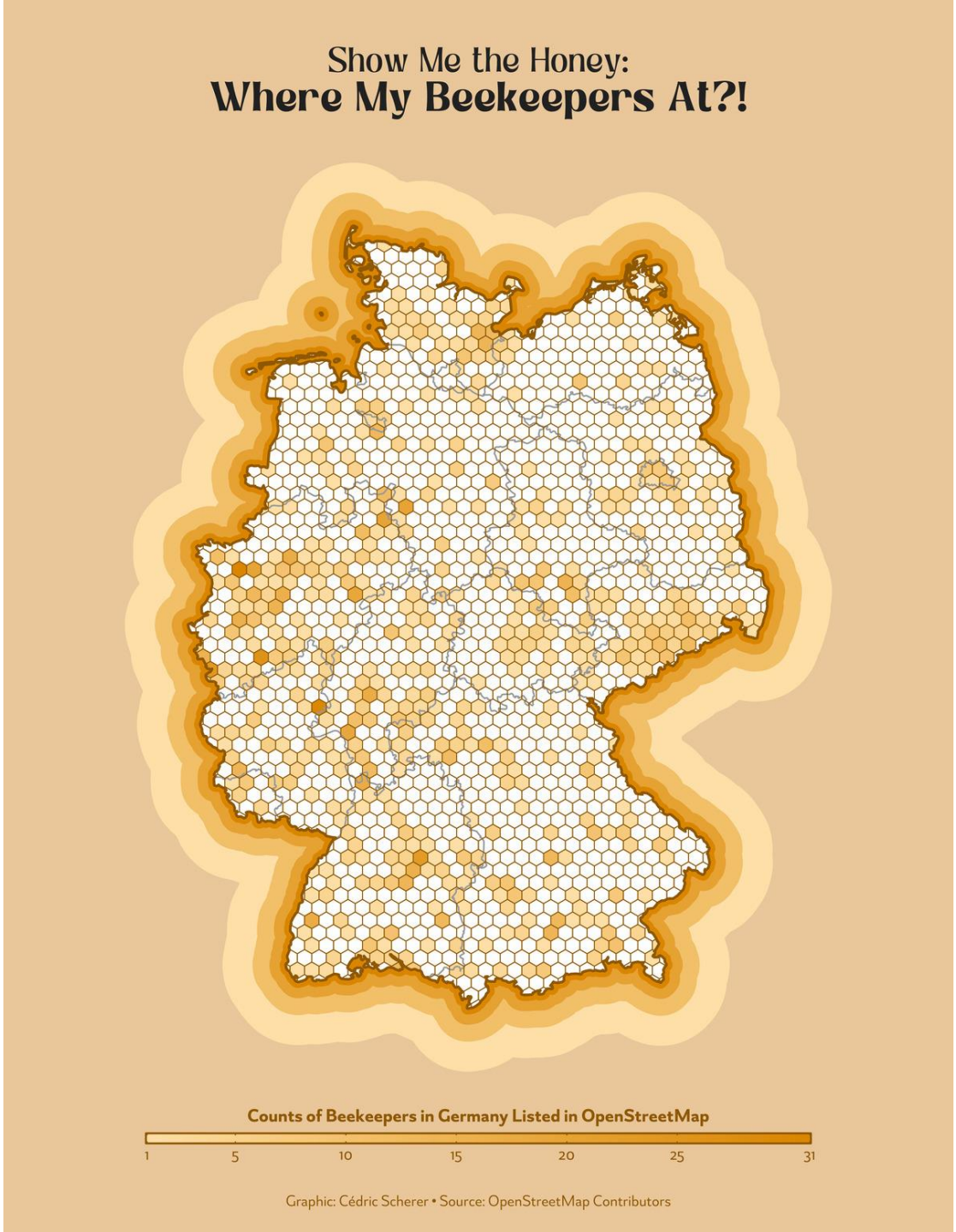




https://www.cedricscherer.com/top/dataviz/

ggplot2





https://www.cedricscherer.com/top/dataviz/

Pipes! %>%

- Sends the output of one function to the input of another

Pipes! %>%

- Sends the output of one function to the input of another

```r
vector <- c(5, 5, 5, 5, 5)
sum(vector)


c(5, 5, 5, 5, 5) %>%
  sum()
```

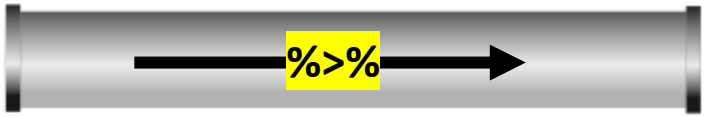
Pipes! %>%

- Sends the output of one function to the input of another

```
vector <- c(5, 5, 5, 5, 5)
sum(vector)


c(5, 5, 5, 5, 5) %>%
  sum()
```

- Create vector



- Sum()

Pipes! %>%

- Can simplify code and make it more intuitive/readable

```r
vector <- c(5, 5, 5, 5, 5)
vector_sum <- sum(vector)
sqrt(vector_sum)


c(5, 5, 5, 5, 5) %>%
  sum() %>%
  sqrt()
```

2. Importing table data into R

Functions for importing table data

- **read_xlsx()**

- read_xlsx(**path** = "*path/to/your/table.xlsx*", **sheet** = "*sheet 1*")

Readxl cheatsheet: https://raw.githubusercontent.com/rstudio/cheatsheets/main/data-import.pdf

Functions for importing table data

- **read_delim()**

- read_delim(**file** = "*path/to/your/file*.filetype", **delim** = '*delimiter*')

Readr cheatsheet: https://raw.githubusercontent.com/rstudio/cheatsheets/main/data-import.pdf

Functions for importing table data

- **read_delim()**

- read_delim(**file** = "*path/to/your/file*.filetype", **delim** = '*delimiter*')

- Common delimiters:
  - comma separated (.csv) **delim** = ','
  - tab separated (.tsv) **delim** = '\t'

Readr cheatsheet: https://raw.githubusercontent.com/rstudio/cheatsheets/main/data-import.pdf

Data import with the tidyverse : : CHEAT SHEET

Read Tabular Data with readr

read_*(file, col_names = TRUE, col_types = NULL, col_select = NULL, id = NULL, locale, n_max = Inf, skip = 0, na = c("", "NA"), guess_max = min(1000, n_max), show_col_types = TRUE) See **?read_delim**

A|B|C
1|2|3
4|5|NA → | A | B | C |
|---|---|---|
| 1 | 2 | 3 |
| 4 | 5 | NA |

read_delim("file.txt", delim = "|"**)** Read files with any delimiter. If no delimiter is specified, it will automatically guess.
To make file.txt, run: write_file("A|B|C\n1|2|3\n4|5|NA", file = "file.txt")

A,B,C
1,2,3
4,5,NA → | A | B | C |
|---|---|---|
| 1 | 2 | 3 |
| 4 | 5 | NA |

read_csv("file.csv"**)** Read a comma delimited file with period decimal marks.
write_file("A,B,C\n1,2,3\n4,5,NA", file = "file.csv")

A;B;C
1,5;2;3
4,5;5;NA → | A | B | C |
|---|---|---|
| 1.5 | 2 | 3 |
| 4.5 | 5 | NA |

read_csv2("file2.csv"**)** Read semicolon delimited files with comma decimal marks.
write_file("A;B;C\n1,5;2;3\n4,5;5;NA", file = "file2.csv")

A B C
1 2 3
4 5 NA → | A | B | C |
|---|---|---|
| 1 | 2 | 3 |
| 4 | 5 | NA |

read_tsv("file.tsv"**)** Read a tab delimited file. Also **read_table()**.
read_fwf("file.tsv", fwf_widths(c(2, 2, NA))**)** Read a fixed width file.
write_file("A\tB\tC\n1\t2\t3\n4\t5\tNA\n", file = "file.tsv")

USEFUL READ ARGUMENTS

A	B	C
1	2	3
4	5	NA

No header
read_csv("file.csv", col_names = FALSE)

x	y	z
A	B	C
1	2	3
4	5	NA

Provide header
read_csv("file.csv",
col_names = c("x", "y", "z"))

Read multiple files into a single table
read_csv(c("f1.csv", "f2.csv", "f3.csv"),
id = "origin_file")

1	2	3
4	5	NA

Skip lines
read_csv("file.csv", skip = 1)

A	B	C
1	2	3

Read a subset of lines
read_csv("file.csv", n_max = 1)

A	B	C
NA	2	3
4	5	NA

Read values as missing
read_csv("file.csv", na = c("1"))

A;B;C
1,5;2;3,0

Specify decimal marks
read_delim("file2.csv", locale =
locale(decimal_mark = ","))

Save Data with readr

write_*(x, file, na = "NA", append, col_names, quote, escape, eol, num_threads, progress)

A	B	C
1	2	3
4	5	NA

→ A,B,C
1,2,3
4,5,NA

write_delim(x, file, delim = " "**)** Write files with any delimiter.

write_csv(x, file**)** Write a comma delimited file.

write_csv2(x, file**)** Write a semicolon delimited file.

One of the first steps of a project is to import outside data into R. Data is often stored in tabular formats, like csv files or spreadsheets.

The front page of this sheet shows how to import and save text files into R using **readr**.

The back page shows how to import spreadsheet data from Excel files using **readxl** or Google Sheets using **googlesheets4**.

OTHER TYPES OF DATA

Try one of the following packages to import other types of files:

- **haven** - SPSS, Stata, and SAS files
- **DBI** - databases
- **jsonlite** - json
- **xml2** - XML
- **httr** - Web APIs
- **rvest** - HTML (Web Scraping)
- **readr::read_lines()** - text data

Column Specification with readr

Column specifications define what data type each column of a file will be imported as. By default readr will generate a column spec when a file is read and output a summary.

spec(x**)** Extract the full column specification for the given imported data frame.

```
spec(x)
# cols(
#   age = col_integer(),
#   sex = col_character(),
#   earn = col_double()
# )
```

age is an integer
sex is a character
earn is a double (numeric)

COLUMN TYPES

Each column type has a function and corresponding string abbreviation.

- **col_logical()** - "l"
- **col_integer()** - "i"
- **col_double()** - "d"
- **col_number()** - "n"
- **col_character()** - "c"
- **col_factor(**levels, ordered = FALSE**)** - "f"
- **col_datetime(**format = ""**)** - "T"
- **col_date(**format = ""**)** - "D"
- **col_time(**format = ""**)** - "t"
- **col_skip()** - "-", "_"
- **col_guess()** - "?"

USEFUL COLUMN ARGUMENTS

Hide col spec message
read_*(file, show_col_types = FALSE)

Select columns to import
Use names, position, or selection helpers.
read_*(file, col_select = c(age, earn))

Guess column types
To guess a column type, read_ *() looks at the first 1000 rows of data. Increase with **guess_max**.
read_*(file, guess_max = Inf)

DEFINE COLUMN SPECIFICATION

Set a default type

```
read_csv(
  file,
  col_type = list(.default = col_double())
)
```

Use column type or string abbreviation

```
read_csv(
  file,
  col_type = list(x = col_double(), y = "l", z = "_")
)
```

Use a single string of abbreviations

```
# col types: skip, guess, integer, logical, character
read_csv(
  file,
  col_type = "_?ilc"
```


googlesheets4
www.rstudio.com

3. Cleaning table data


Real world data
iris &
mtcars

Janitor

> Data scientists, according to interviews and expert estimates, spend from 50 percent to 80 percent of their time mired in this more mundane labor of collecting and preparing unruly digital data, before it can be explored for useful nuggets.
>
> – "For Big-Data Scientists, 'Janitor Work' Is Key Hurdle to Insight" *(New York Times, 2014)*

https://www.nytimes.com/2014/08/18/technology/for-big-data-scientists-hurdle-to-insights-is-janitor-work.html

Janitor

- **clean_names()**
  - As the name suggests! E.g. SaMpLe.NaMe% -> sample.name

Janitor

- **clean_names()**
  - As the name suggests! E.g. SaMpLe.NaMe% -> sample.name

- **remove_empty()**
  - Removes empty columns

Janitor

- **clean_names()**
  - As the name suggests! E.g. SaMpLe.NaMe% -> sample.name
- **remove_empty()**
  - Removes empty columns
- **remove_constant()**
  - Removes useless columns

Janitor

- **clean_names()**
  - As the name suggests! E.g. SaMpLe.NaMe% -> sample.name
- **remove_empty()**
  - Removes empty columns
- **remove_constant()**
  - Removes useless columns
- **get_dupes()**
  - Finds duplicate entries (rows)

Dplyr

- **distinct()**
  - Prints only distinct rows in a data frame

tibble

- **as.data.frame (base R)**

- **tibble()**
  - Tidyverse functions automatically import tabular data to tibbles
  - Provides a more succinct overview of your data!
  - Automatically prints the types of data for each column (e.g. chr, dbl)

Exercise time!

- See the exercises in “Exercises.R”

Summary

- The Tidyverse 

- Importing data into R 

- Cleaning data