Session 5. A date with EDA: Bring your own data

Antonio Paez
My Name

28 June, 2022

Highlights:

This is my mini-reflection. Paragraphs must be indented.
It can contain multiple paragraphs.

Threshold Concepts:

threshold concept 1
threshold concept 2
threshold concept 3
threshold concept 4

> "Errors using inadequate data are much less than those using no data at all."
>
> — Charles Babbage

> "Data that is loved tends to survive."
>
> — Kurt Bollacker

Session Outline

- Reading external data
- Hands-on practice
- Some additional notes on working with Rmarkdown

Reminder

The human brain is a highly evolved machine with a specialization in visual recognition of spatial patterns, including shapes, areas, lengths, directions, and colors. This is why well-designed statistical plots are so effective for conveying complex information.

Preliminaries

Clear the workspace from *all* objects:

```
rm(list = ls())
```

Load packages. Remember, packages are units of shareable code that augment the functionality of base `R`. For this session, the following package/s is/are used:

```
library(dplyr) # A Grammar of Data Manipulation
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(edashop) # A Package for a Workshop on Exploratory Data Analysis
library(ggplot2) # Create Elegant Data Visualisations Using the Grammar of Graphics
library(haven) # Import and Export 'SPSS', 'Stata' and 'SAS' Files
library(readr) # Read Rectangular Text Data
library(readxl) # Read Excel Files
library(sf) # Simple Features for R
```

```
## Linking to GEOS 3.9.1, GDAL 3.3.2, PROJ 7.2.1; sf_use_s2() is TRUE
```

Reading external data

In this workshop we have used data that was meticulously prepared and documented to make it analysis-ready. But often we need to read data from so-called external sources, that is, files that are not in native `R` format, including Excel files, csv files, Stata files, shape files, and so on.

The original files of the data sets provided in {edashop} are shared with the package. For example, check the following help file:

```
?i40_index_rank
```

This is one of two data frames created using original files from Honti et al. (2020).

The path to the source file is:

```
external_csv <- system.file("extdata",
                            "rankings.csv",
                            package = "edashop")
```

You can check that this object is just a string with the path to the file:

```
external_csv
```

```
## [1] "C:/Users/paezha/AppData/Local/R/win-library/4.2/edashop/extdata/rankings.csv"
```

One way to read files in csv format is with function `read.csv`:

```
imported_csv <- read.csv(external_csv)
```

Once the file is imported it can be saved as a file of type rda for ease of access in the future. For reproducibility purposes, it is highly recommended to keep the source files intact and document any data processing done in a script or Rmarkdown document. For example, here we change the names of the columns in the table we just imported:

```
imported_csv <- imported_csv |>
  rename(NUTS_ID = Regio,
         gdp_rank = GDPrank,
         promethee_rank = PrometheeRank,
         regional_innovation_index = RII)
```

Then we save the file:

```
save(external_csv,
     file = "external_csv.rda",
     compress = "xz")
```

This makes it easy to retrieve our prepared data.
An external file in Stata's dat format is also shared:

```
external_dta <- system.file("extdata",
                            "phd_italy.dta",
                            package = "edashop")
```

As before `external_dta` is a string with the path to the file:

```
external_dta
```

```
## [1] "C:/Users/paezha/AppData/Local/R/win-library/4.2/edashop/extdata/phd_italy.dta"
```

Package {haven} has utilities for reading external data in various common formats, including dta:

```
imported_dta <- read_dta(external_dta)
```

Package {readxl} has utilities to read Excel files:

```
external_xlsx <- system.file("extdata",
                            "italy-nuts-codes.xlsx",
                            package = "edashop")

imported_xlsx <- read_xlsx(external_xlsx)
```

As a last example, package {sf} includes utilities to read shape files (and much, much more besides!):

```
external_shp <- system.file("extdata",
                            "nuts2.shp",
                            package = "edashop")

imported_shp <- st_read(external_shp)
```

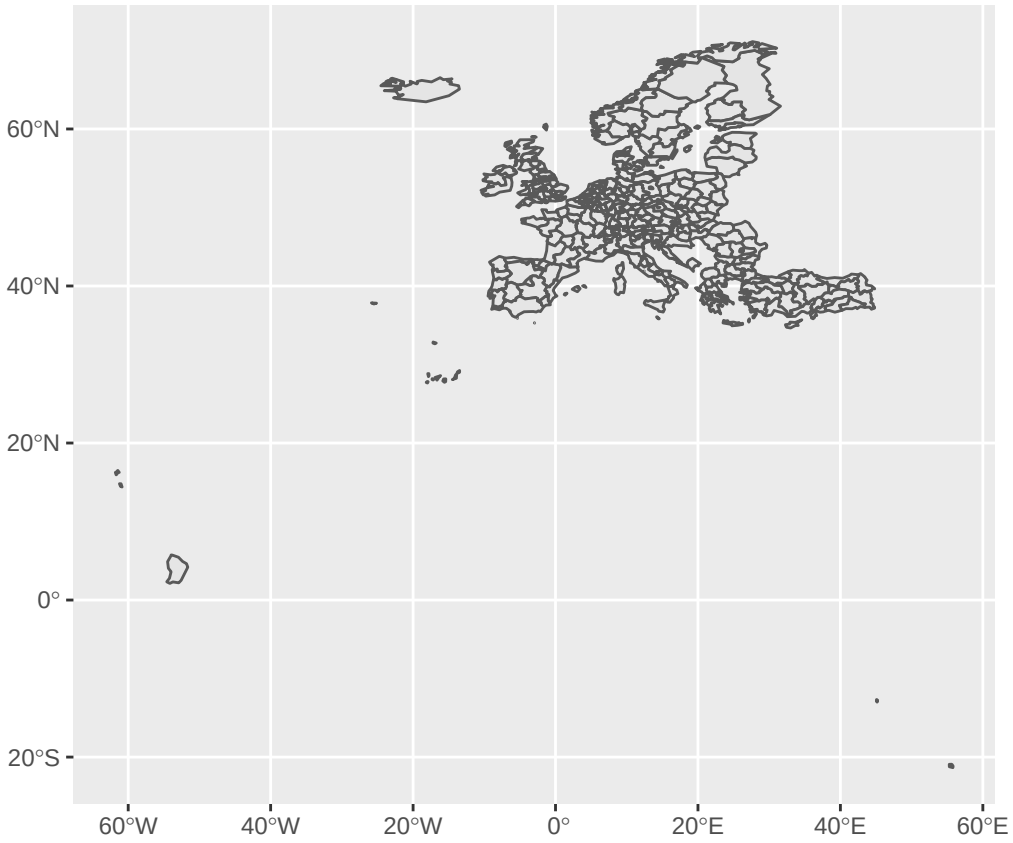
```
## Reading layer 'nuts2' from data source
##   'C:\Users\paezha\AppData\Local\R\win-library\4.2\edashop\extdata\nuts2.shp'
##   using driver 'ESRI Shapefile'
## Simple feature collection with 320 features and 6 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: -61.806 ymin: -21.35 xmax: 55.826 ymax: 71.127
## Geodetic CRS:  WGS 84
```

Hands-on practice

If you brought a data file of your own in an external format, try reading it here. Then, you can play with it.

As an example, I am going to plot the shape file that we just read before. Notice that spatial data in simple features (sf) format are a type of geometric object in the grammar of {ggplot}:

```
ggplot() +
  geom_sf(data = imported_shp)
```

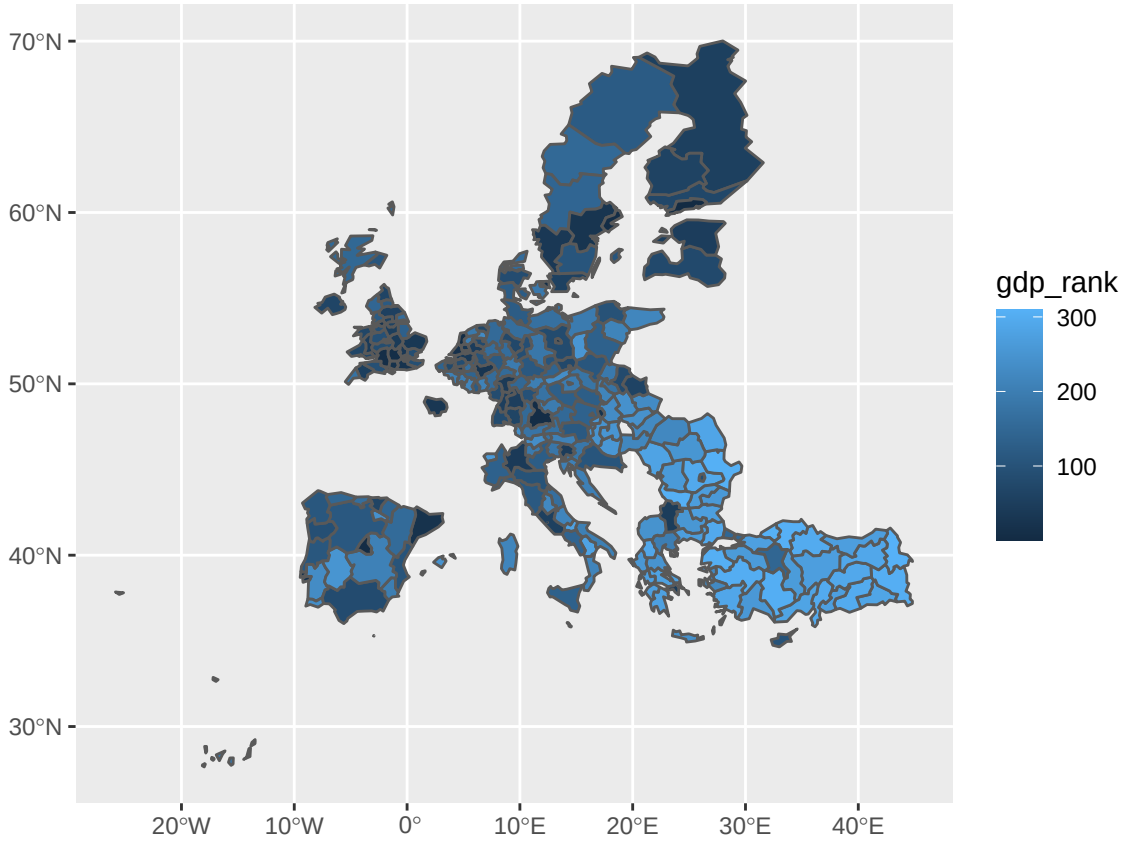


Next we can join the geometry of the boundaries to the rankings of industrial readinness, and convert to simple features:

```
i40_rankings <- imported_csv |>
  left_join(imported_shp,
            by = "NUTS_ID") |>
  st_sf()
```

Now we can map something else, for instance, by encoding it to the *fill* property of the polygons (the color of the areas):

```
ggplot() +
  geom_sf(data = i40_rankings,
          aes(fill = gdp_rank))
```



A great resource to learn about working with spatial data in R is Lovelace et al. (2019).

Some additional notes on working with Rmarkdown

Rmarkdown is a great invention. It implements principles of literate programming and enforces good discipline when writing and documenting not only code but also analysis processes.

Here is some additional information of value. The chunks of code that we use to communicate with the computer accept a number of options that stipulate the behavior of the chunk. Chunk options are written inside the curly brackets. They can also be named:

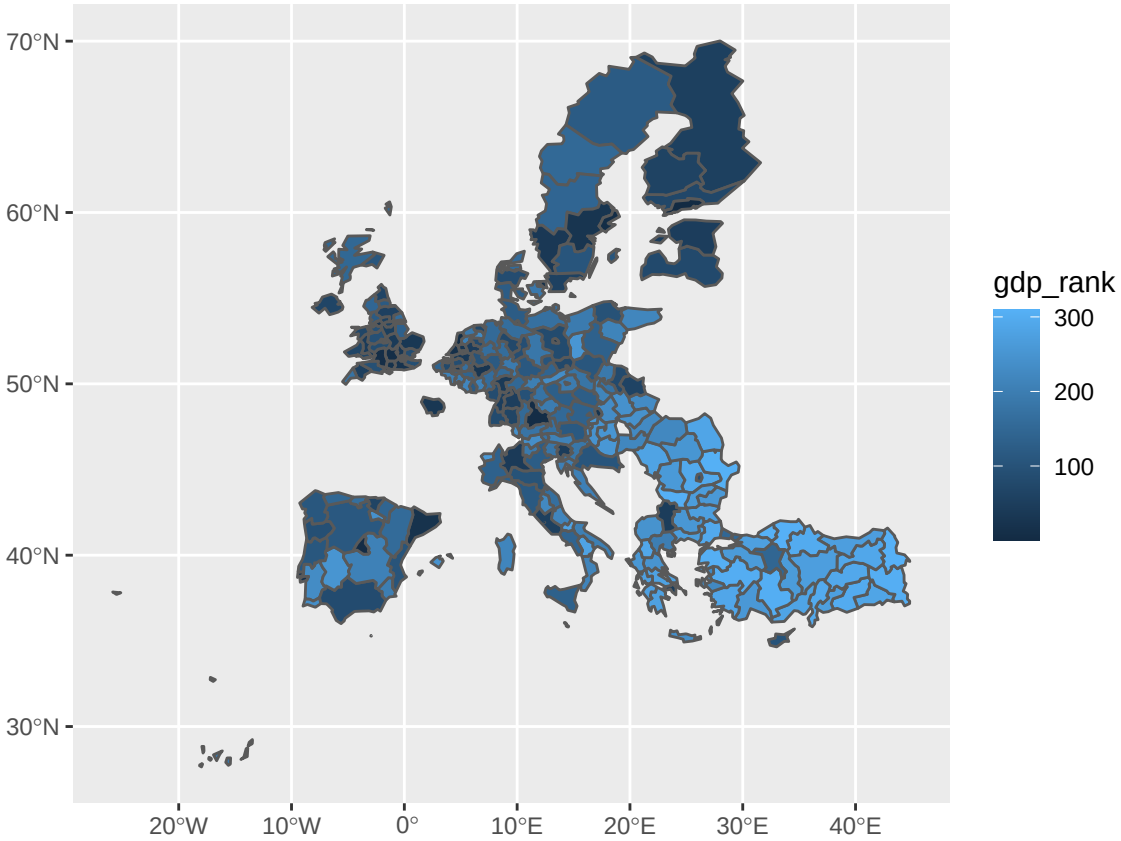
```
# Option eval controls whether the chunk is evaluated (i.e., run).
# This chunk of code can only be run manually, by clicking the play icon,
# but will be ignored when running the document or when knitting
?edashop
```

Some chunks, like eval, control whether the code is run. Others, like echo, control whether the chunk itself is printed in the output document when knitting:

```
## [1] "This chunk of code will not be printed in the output. The result of running the code will"
```

Chunk include is used to include code and results of the code, or to suppress all. For example, when reading files or loading data, we may not wish to have those things printed in the output document:

For example, let us say that we wish to show a plot in a paper or report, but we do not need to show the



code:

Rmarkdown is a great format for creating reproducible research reports and papers. These are some examples:

- Paez, Antonio, et al. "A spatio-temporal analysis of the environmental correlates of COVID-19 incidence in Spain." Geographical analysis 53.3 (2021): 397-421. The repository with the reproducible document is here: https://github.com/paezha/covid19-environmental-correlates
- Paez A, Higgins CD (2021) The Accessibility Implications of a Pilot COVID-19 Vaccination Program in Hamilton, Ontario. Findings (doi.org/10.32866/001c.24082). The repository with the reproducible document is here: https://github.com/paezha/Accessibility-Pharmacies-Hamilton-Vaccines
- Higgins, C.D., Páez, A., Kim, G., Wang, J. (2021) Changes in accessibility to emergency and community food services during COVID-19 and implications for low income populations in Hamilton, Ontario. Social Science and Medicine, 291:114442 (doi.org/10.1016/j.socscimed.2021.114442). The repository with the reproducible document is here: https://github.com/paezha/Accessibility-Food-Banks-Hamilton

For your practice, you can create a new template from package {rticles} and write a small exploratory data analysis report using your data set or one of the data sets provided.

Happy analysis!

Practice

1. Install package {rticles}.
2. Create a new Rmarkdown file from am {rticles} template of your choice.
3. Transfer your data analysis exercise to the template and write a small report.