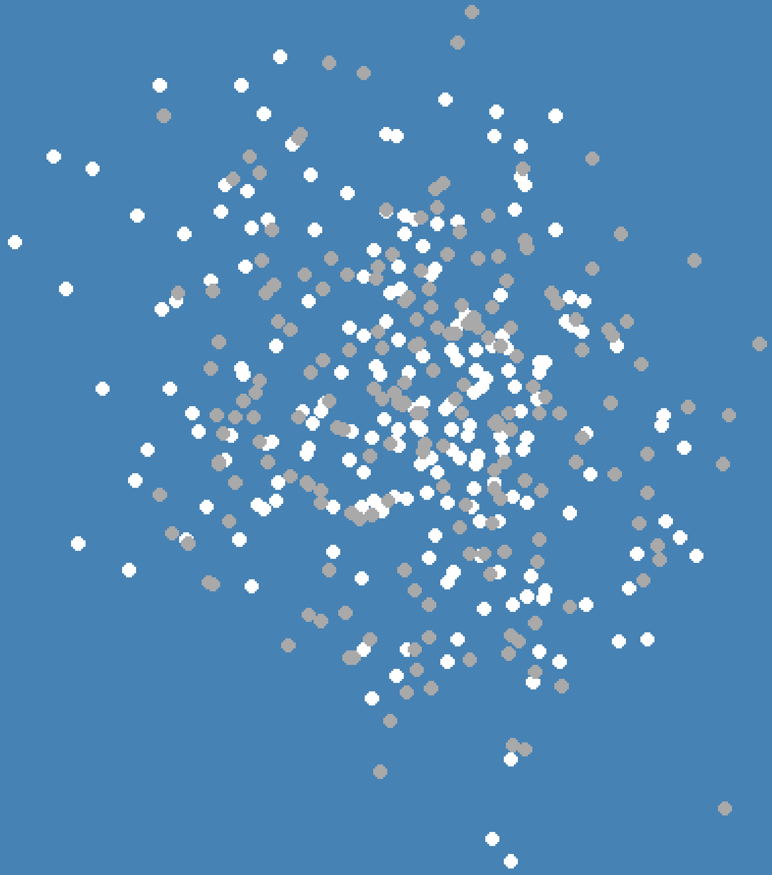
Introduction to R

2.2 Loading and Storing Data

Lion Behrens, M.Sc.

UNIVERSITY OF MANNHEIM

University of Mannheim
Chair of Social Data Science and Methodology
Chair of Quantitative Methods in the Social Sciences



Overview

Good news: R is data-agnostic, you can import data of a large range of shapes & types in R.

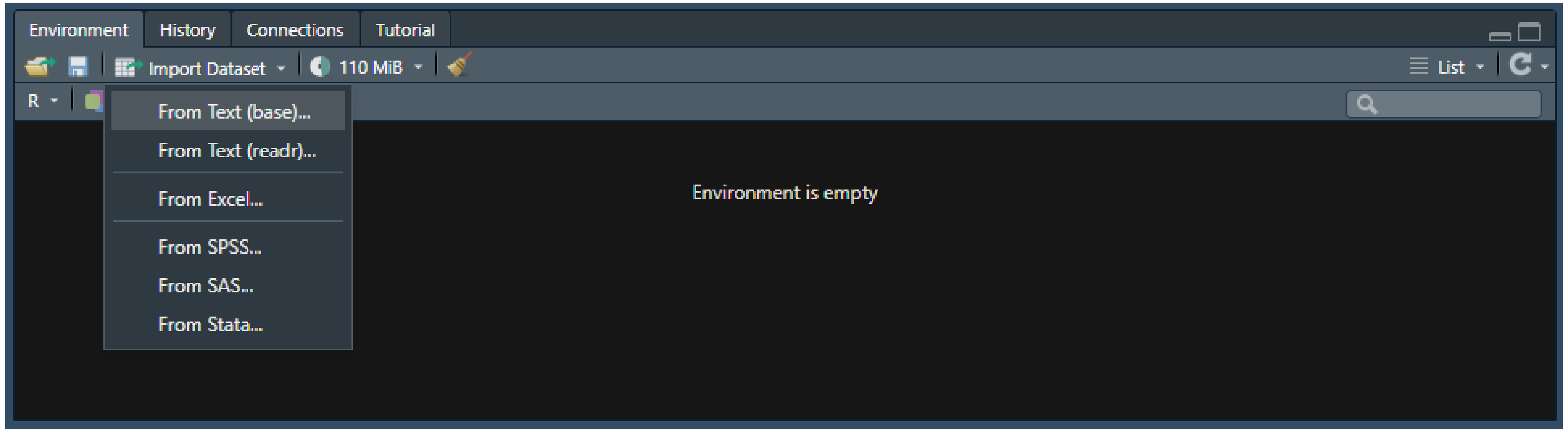
What we will cover

- Different ways how we can potentially move data into R
- How to import the most common data formats into R

Importing Data

Built-in Functionality for Manually Loading Data

The RStudio Graphical User Interface comes with built-in functionalities to manually define how you want to import data.



- define data type
- define object name in R
- define rows to skip, column names etc.

Importing native .RData or .rds files

Sometimes, you will import data that has been generated from within R by another user.

.Rdata files

```
load("your_data.RData")
```

- restores the workspace that has been saved to your_data.RData
- includes all "original" object names
- can load many objects (e.g. data frames) at once

.rds files

```
your_dataNEW <- readRDS("your_data.rds")
```

- you can assign the content of your_data.rds to a new object name
- typically used to load in one individual data frame

Absolute vs. Relative File Paths

When importing data that is located offline on your CPU, we will need to specify a file path.

You can either import a dataset specifying an absolute file path:

- Let's say your data is called your_data.RData
- Data is located on your Desktop
- Specify the absolute file path:

```r
load("C:/User/user/Desktop/your_data.RData")
```

Or you import a dataset using a path relative to your working directory:

```r
setwd("C:/User/user/Desktop")
load("your_data.RData")
```

- path relates to the working directory that you have defined before
- R loads the file your_data.RData that is located in this directory

Note: You can't directly copy-paste file paths from your operating system (e.g. Windows) to R.

- R uses / in file paths
- Windows uses \
- Typing \\ also works in R

Importing External File Formats: tidyverse

Very often, we will import data from an external file that is not a native R object.

- .txt-file?
- comma- or tab-delimited?
- Stata- or SPSS-file?

The tidyverse holds a range of packages available that allow you to import almost any kind of external data file into R.

- readr::read_csv() for comma delimited files
- readr::read_csv2() for semicolon delimited files
- readr::read_tsv() for tab delimited files

- readxl::read_excel() for .xls and xlsx files

- haven::read_dta() for Stata files (.dta)
- haven::read_sav() for SPSS files (.sav)

Importing External File Formats: tidyverse

```r
library(haven)
ess10 <- haven::read_dta("./dat/ESS10.dta")
print(ess10[1:15, 1:10])
```

```
## # A tibble: 15 × 10
##    name        essro…¹ edition prodd…²  idno cntry dweight pweight nwspol netus…³
##    <chr>         <dbl> <chr>   <chr>   <dbl> <chr>   <dbl>   <dbl> <dbl+> <dbl+l>
##  1 ESS10e01_2       10 1.2     28.06.… 10002 BG      1.03    0.218  80     1 [Nev…
##  2 ESS10e01_2       10 1.2     28.06.… 10006 BG      0.879   0.218  63     5 [Eve…
##  3 ESS10e01_2       10 1.2     28.06.… 10009 BG      1.01    0.218 390     5 [Eve…
##  4 ESS10e01_2       10 1.2     28.06.… 10024 BG      0.955   0.218  60     5 [Eve…
##  5 ESS10e01_2       10 1.2     28.06.… 10027 BG      0.841   0.218 120     5 [Eve…
##  6 ESS10e01_2       10 1.2     28.06.… 10048 BG      0.946   0.218  60     5 [Eve…
##  7 ESS10e01_2       10 1.2     28.06.… 10053 BG      1.01    0.218  30     5 [Eve…
##  8 ESS10e01_2       10 1.2     28.06.… 10055 BG      1.03    0.218  70     5 [Eve…
##  9 ESS10e01_2       10 1.2     28.06.… 10059 BG      0.991   0.218  60     1 [Nev…
## 10 ESS10e01_2       10 1.2     28.06.… 10061 BG      1.05    0.218  60     1 [Nev…
## 11 ESS10e01_2       10 1.2     28.06.… 10064 BG      1.00    0.218 300     5 [Eve…
## 12 ESS10e01_2       10 1.2     28.06.… 10068 BG      1.03    0.218   0     1 [Nev…
## 13 ESS10e01_2       10 1.2     28.06.… 10071 BG      0.931   0.218  30     5 [Eve…
## 14 ESS10e01_2       10 1.2     28.06.… 10077 BG      0.991   0.218  30     1 [Nev…
## 15 ESS10e01_2       10 1.2     28.06.… 10078 BG      0.990   0.218  60     1 [Nev…
## # … with abbreviated variable names ¹essround, ²proddate, ³netusoft
```

Importing External File Formats: Other packages

There is a range of other packages that you can use for data import as well as some base R functions.

base R

- read.table()
- read.csv()
- read.delim()
- read.delim2()

Packages

foreign package:

- foreign::read.dta()
- foreign::read.spss()

readstata13 package:

- readstata13::read.dta13()

Note: Type help(read.table) or ?read.table into your R console for more information.

Saving Your Data

Exporting Data: External File Formats

Most read_*-functions come with a sibling that allows you to store your R data frame as an external file format on your CPU. The haven package offers a lot of those, but these functionalities can also be found in base R using the read.* prefix ("." rather than "_").

tidyverse

- write_csv() writes comma delimited files
- write_csv2() writes semicolon delimited files
- write_tsv() writes tab delimited files
- write_dta() writes Stata files (.dta)
- write_sav() writes .sav files

base R

- write.csv() writes comma delimited files
- write.csv2() writes semicolon delimited files

foreign package

- write.dta() writes Stata files (.dta)
- write.foreign(df, package = c("SPSS", "Stata", "SAS")) is a generic function that exports simple data frames to other statistical packages of your choice

Example of Data Export and Import

Write data frame to hard disk

```r
library(readr)

# write to csv file
write_csv(ess10, "./dat/ess10.csv")
```

Import data, inspect

```r
# re-import data from csv file
ess10 <- read_csv("./dat/ess10.csv")

# inspect
print(ess10[1:10, 1:4])
```

```
## # A tibble: 10 × 4
##    name        essround edition proddate
##    <chr>          <dbl>   <dbl> <chr>
##  1 ESS10e01_2        10     1.2 28.06.2022
##  2 ESS10e01_2        10     1.2 28.06.2022
##  3 ESS10e01_2        10     1.2 28.06.2022
##  4 ESS10e01_2        10     1.2 28.06.2022
##  5 ESS10e01_2        10     1.2 28.06.2022
##  6 ESS10e01_2        10     1.2 28.06.2022
##  7 ESS10e01_2        10     1.2 28.06.2022
##  8 ESS10e01_2        10     1.2 28.06.2022
##  9 ESS10e01_2        10     1.2 28.06.2022
## 10 ESS10e01_2        10     1.2 28.06.2022
```

Exporting Data: R's Native Formats

If you wish to stick to R's native file formats, you essentially have two options:

.RData files

```
save(your_data, file = "your_data.RData")
load("your_data.RData")
```

- allows you to store your whole workspace (incl. several objects)

.rds files

```
saveRDS(your_data, file = "your_data.rds")
your_dataNEW <- readRDS("your_data.rds")
```

- only works for individual R objects

Variable and Value Labels

Labelled Data

Especially with pre-compiled data sources that you haven't collected yourself, datasets often come with variable labels. That is, columns are described by their column name and a variable label. Labels are metadata that provide a more detailed description of the column names or the numerical values of a particular variable.

How to read in your data in a way that labels are accessible?

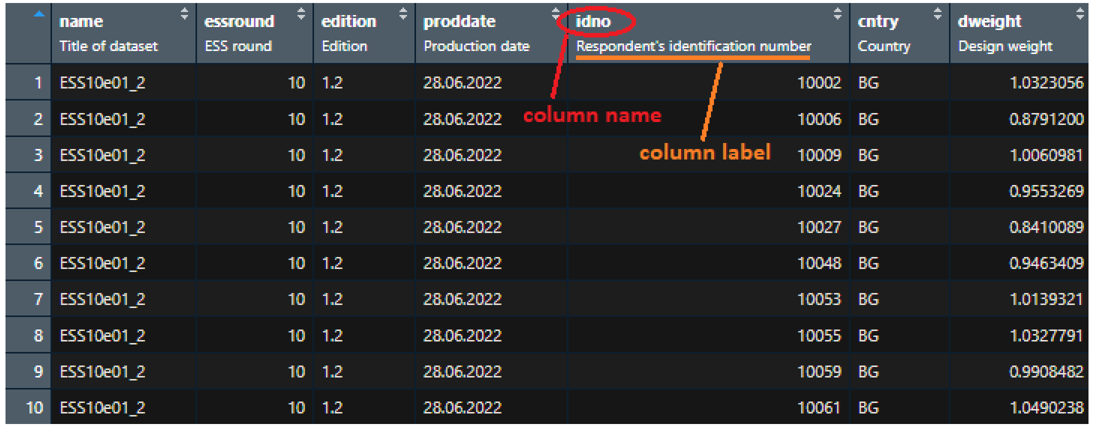
Not all functions for importing data recover variable and value labels that are stored in the metadata.

- The haven package recovers these.
- With the sjlabelled package, you can easily assess them.

```r
library(haven)
ess10 <- haven::read_dta("./dat/ESS10.dta")
```

Column labels

```r
View(ess10)
```



	name Title of dataset	essround ESS round	edition Edition	proddate Production date	idno Respondent's identification number	cntry Country	dweight Design weight
1	ESS10e01_2	10	1.2	28.06.2022	10002	BG	1.0323056
2	ESS10e01_2	10	1.2	28.06.2022	10006	BG	0.8791200
3	ESS10e01_2	10	1.2	28.06.2022	10009	BG	1.0060981
4	ESS10e01_2	10	1.2	28.06.2022	10024	BG	0.9553269
5	ESS10e01_2	10	1.2	28.06.2022	10027	BG	0.8410089
6	ESS10e01_2	10	1.2	28.06.2022	10048	BG	0.9463409
7	ESS10e01_2	10	1.2	28.06.2022	10053	BG	1.0139321
8	ESS10e01_2	10	1.2	28.06.2022	10055	BG	1.0327791
9	ESS10e01_2	10	1.2	28.06.2022	10059	BG	0.9908482
10	ESS10e01_2	10	1.2	28.06.2022	10061	BG	1.0490238

```r
library(sjlabelled)
sjlabelled::get_label(ess10$idno)
```

```
## [1] "Respondent's identification number"
```

Value labels

Start with tabulating the variable happy in an uninformative way:

```
table(ess10$happy)
```

```
##
##    0    1    2    3    4    5    6    7    8    9   10
##  164  122  254  521  627 2158 1808 3262 4483 2585 2030
```

Value labels

Now, get the column label of the variable to understand what is measured.

```
sjlabelled::get_label(ess10$happy)
```

```
## [1] "How happy are you"
```

Finally, get the value labels to understand how it is measured.

```
sjlabelled::get_labels(ess10$happy)
```

```
##  [1] "Extremely unhappy" "1"                 "2"
##  [4] "3"                 "4"                 "5"
##  [7] "6"                 "7"                 "8"
## [10] "9"                 "Extremely happy"   "Refusal"
## [13] "Don't know"        "No answer"
```

Attention: Haven package

One thing to note about the haven package is that although it recovers value labels and variable labels, it stores all variables of a data frame as a special data type: haven_labelled.

```
class(ess10$cntry)
```

```
## [1] "haven_labelled" "vctrs_vctr"     "character"
```

This data type is not always compatible with other procedures in R, but we will see some workarounds!

References

Parts of this course are inspired by the following resources:

- Wickham, Hadley and Garrett Grolemund, 2017. *R for Data Science - Import, Tidy, Transform, Visualize, and Model Data*. O'Reilly.
- Bahnsen, Oke and Guido Ropers, 2022. *Introduction to R for Quantitative Social Science*. Course held as part of the GESIS Workshop Series.
- Breuer, Johannes and Stefan Jünger, 2021. *Introduction to R for Data Analysis*. Course held as part of the GESIS Summer School in Survey Methodology.
- Teaching material developed by Verena Kunz, David Weyrauch, Oliver Rittmann and Viktoriia Semenova.