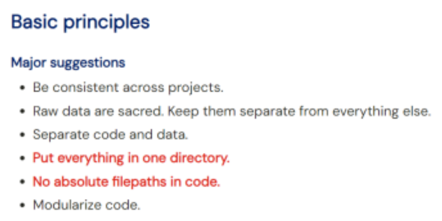
Module 5 Pre-Read - here package

Saturday, February 17, 2024 23:00

Project organization

Learning objectives
1. Define the major principles of project-based organization.
2. Distinguish between absolute and relative paths as it relates to reproducibility.
3. Identify the role the here package plays in file path management.
4. Describe the benefits of modularity in code

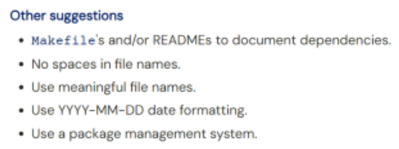
Learning objectives: redux
1. Reorganize projects using the command line.
2. Edit absolute file paths in R scripts and R Markdown documents to be relative using the here package.



Basic principles

Major suggestions

- Be consistent across projects.
- Raw data are sacred. Keep them separate from everything else.
- Separate code and data.
- Put everything in one directory.
- No absolute filepaths in code.
- Modularize code.



Other suggestions

- `Makefile`'s and/or READMEs to document dependencies.
- No spaces in file names.
- Use meaningful file names.
- Use YYYY-MM-DD date formatting.
- Use a package management system.

What to organize?
Depending on the nature of the project, the organization system may look different.

E.g., data analysis vs. methods paper vs. presentation vs. web app vs. etc...
The systems should adhere to the same general principles, but different requirements may necessitate different structures.

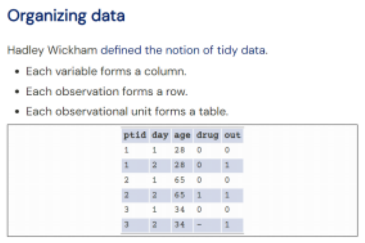
Think about organization of a project from the outset!

Organizing data
Raw data are sacred... but may be a mess.

You'll be surprised (and disheartened) by how many color-coded excel sheets you'll get in your life.
Tempting to edit raw data by hand. Don't!

Everything scripted!
Use meta-data files to describe raw and cleaned data.

structure as data (e.g., .csv) so easy to read

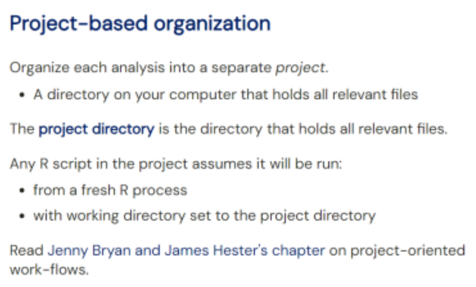


Organizing data

Hadley Wickham defined the notion of tidy data.

- Each variable forms a column.
- Each observation forms a row.
- Each observational unit forms a table.

ptid	day	age	drug	out
1	1	28	0	0
1	2	28	0	1
2	1	65	0	0
2	2	65	1	1
3	1	34	0	0
3	2	34	-	1



Project-based organization

Organize each analysis into a separate *project*.

- A directory on your computer that holds all relevant files

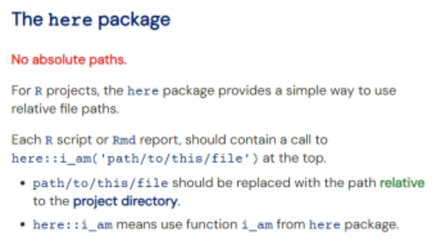
The **project directory** is the directory that holds all relevant files.

Any R script in the project assumes it will be run:

- from a fresh R process
- with working directory set to the project directory

Read Jenny Bryan and James Hester's chapter on project-oriented work-flows.

The here package



The `here` package

No absolute paths.

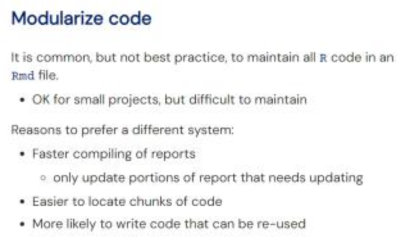
For `R` projects, the `here` package provides a simple way to use relative file paths.

Each `R` script or `Rmd` report, should contain a call to `here::i_am('path/to/this/file')` at the top.

- `path/to/this/file` should be replaced with the path relative to the **project directory**.
- `here::i_am` means use function `i_am` from `here` package.

~~~~
```{r, load-data, echo = FALSE}
library(here)
i_am("hiv_report.Rmd")
full_file_path <- here("data", "vrc01_data.csv")
data <- read.csv(full_file_path, header = TRUE)
```
~~~~

Modularize code



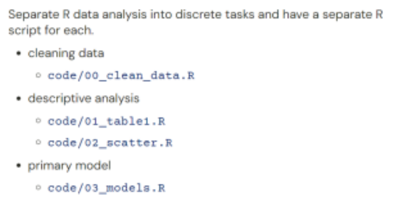
Modularize code

It is common, but not best practice, to maintain all `R` code in an `Rmd` file.

- OK for small projects, but difficult to maintain

Reasons to prefer a different system:

- Faster compiling of reports
  - only update portions of report that needs updating
- Easier to locate chunks of code
- More likely to write code that can be re-used



Separate R data analysis into discrete tasks and have a separate R script for each.

- cleaning data
  - `code/00_clean_data.R`
- descriptive analysis
  - `code/01_table1.R`
  - `code/02_scatter.R`
- primary model
  - `code/03_models.R`

Saving objects from R scripts so it can be read in from other R scripts:
This saves ur dataset as `data_clean` in the folder called `output`

```
saveRDS(
    Object_i_wanna_save,
    File = here::here("output/data_clean.rds")
)
```

For images, use
```
ggsave(
    here::here("output/scatterplot.png"),
    Plot = scatterplot,
    Device = "png"
)
```

Now accessing that data u just saved
```
Data <- readRDS(
    File = here::here("output/data_clean.rds")
)
```

For Figures:
```
Knitr::include_graphics(
    Here::here("output/scatterplot.png")
)
```

In the command line, to run an rscript, do
```
$ Rscript code/03_models.R
```