Reproducible Workflows (+ Data Cleaning)

January 22, 2026

Plan for Today

1 Reproducible Workflows

+ Setting up an organized project file structure
+ Creating a reproducible environment using renv + conda
+ Creating a reproducible report using quarto

2 Data Cleaning

Reproducible Workflows

Step 1: Organizing your Project File Structure

```
dsip/
|_ project_name/
  |_ data/               # store all raw and processed data
  |_ notebooks/          # store all notebooks (.qmd, .Rmd, .ipynb, ...)
  |_ other/              # miscellaneous documents
  |_ R/                  # store R functions
  |_ python/             # store python functions
  |_ scripts/            # store R/python scripts
  |_ results/            # store all results
  |_ renv/               # do not edit; created automatically by renv (R only)
  .Rprofile              # R project (R only)
  renv.lock              # do not edit; created automatically by renv (R only)
  environment.yml        # yml file to reproduce conda environment (python only)
  conda-lock.yml         # conda lock file to reproduce conda environment (python only)
```

Step 2: Creating a reproducible environment

- **Goal:** to install all necessary packages/dependencies in one command
  - Using **renv** in **R**:

    ```
    renv::restore()
    ```

  - Using **conda** + **conda-lock** in **Python**:

    ```
    conda-lock install --name {env_name}
    ```

- We will also follow best practice and create a different environment for each project/lab

Reproducible environments in python using conda

Conda cheat sheet

- `conda create --name [env_name]`: initialize a conda environment
- `conda env remove --name [env_name]`: remove a conda environment
- `conda activate [env_name]`: activate a conda environment
- `conda deactivate`: deactivate current conda environment
- `conda install [pkg_name]`: install package in active conda environment
- `conda update [pkg_name]`: update package in active conda environment
- `conda remove [pkg_name]`: remove package from active conda environment
- `conda list`: list all packages installed in active conda environment
- `conda env export --from-history > environment.yml`: export conda environment (explicitly installed packages only) to yml
- `conda env create -f environment.yml`: create conda environment from yml

Reproducible environments in R using `renv`

- `renv::init()`: initialize renv in a project
- `renv::status()`: check current status of environment; report inconsistencies between lockfile, library, and dependencies
- `renv::install()`: install packages
- `renv::remove()`: remove packages
- `renv::update()`: update packages
- `renv::snapshot()`: record current state of project library in the lockfile
- `renv::restore()`: restore project library from a lockfile

Full list of available functions here

Overview of renv and conda

After navigating to your project root directory,

	renv	conda
1. **Create environment:**	`renv::init()`	`conda create --name env_name`
2. **Activate:**	`renv::activate()` (not necessary if you're working in your project root directory because of .Rprofile)	`conda activate env_name`
3. **Add packages:**	`renv::install("pkg_name")`	`conda install pkg_name`

4. **Create/update lock file:**	`renv::snapshot()`	`conda env export --from-history > environment.yml`
		`conda lock`

Step 3: Creating a reproducible report with quarto

- A powerful open-source tool for creating **dynamic documents**, reports, presentations, and websites using a simple **markdown-based format**
  - Akin to R Markdown but has multi-language support (R, Python, Julia, JavaScript, …)
- Designed to enhance **reproducible research** and data science workflows by allowing users to **combine code, text, and visualizations** in a single document
- How to install?
  - Download quarto: https://quarto.org/docs/download/index.html
  - To check installation, type in terminal: `quarto --version`

When it comes time to replicate your labs, I will...

1. Change directories to your `lab1/` folder
2. Run:
   - `renv::restore()` [for R users]
   - `conda-lock install --name [env_name]` [for python users]
3. Run `quarto render "notebooks/lab1.qmd"` or follow the steps outlined in your `lab1/readme.md`
   - This could include special instructions on how to download the data files and where to store the data. By default, I will assume the data should be placed in the `dsip/lab1/data/` folder. Note: do NOT push data to GitHub due to strict file size limits (100MB).
   - I will rerun your analysis locally to generate the results myself. If your analysis is too computationally expensive to run locally in a reasonable amount of time, please store and push all results necessary to reproduce your lab report. However, you should also include the code and instructions on how to generate these results in the readme.
   - Note: Be careful with your file names and make sure this is not specific to your computer. Use relative paths or tools like `here::here()` (in R).
4. Output of this process should be your rendered report (html, pdf, ...)

Recap + Next Time

Recap

- Organizing your project file structure
- Managing package dependencies using **renv** and **conda** for reproducible environments
- Reproducible reporting with **quarto**

Next time

- Quickstart on data cleaning on Tuesday
  - **Please carefully read through Tolle et al. before next class**
  - Begin playing around with the data