Simple tips for writing and publishing clear code to ensure reproducible results

Roman Hornung

Institute for Medical Information Processing, Biometry and Epidemiology, LMU Munich

Workshop “Open Replicable Research”, Munich

October 6, 2023
Serving as a Reproducible Research Editor at the Biometrical Journal since May 2022, along with Fabian Scheipl and Lien Le.

Authors are required to ensure the reproducibility of their analyses by submitting code, which is published alongside the paper.

The submitted code undergoes a review process to verify its reproducibility and adherence to certain structural requirements, enhancing its manageability for independent individuals.
Reproducible Research at the Biometrical Journal

- The upcoming tips are based on the Biometrical Journal’s reproducible research guidelines and my experiences in reproducing results from various papers.

- Specifically, my tips are based on common challenges encountered during my work for the Biometrical Journal.

- A fundamental principle in code structuring and commenting is: Keep it simple! 😊
Why Reproducible Code?

- **Validation of Results:**
  - Code serves as *empirical proof*, validating the results and claims made in studies.
  - **Lack** of accessible code equates to *unproven claims* (adapted from Fabian Scheipl).

- **Clarity and Detail:**
  - Code encapsulates every detail of the analysis flow, **improving** the paper’s *readability* by omitting exhaustive descriptions.
  - Ensures **no detail is omitted** or overlooked.

- **Encourages Reuse and Extension:**
  - Others can use, modify, and expand upon the published methods, simulation designs, code etc.
  - **Accompanying data availability** enables further research by peers.
Why Reproducible Code?

■ **Enhances Integrity:**
  ■ Publishing code mandates tidiness, aiding in *early error detection*.
  ■ Results are *more trustworthy* and less susceptible to manipulation.

■ **Promotes Open Science:**
  ■ Sharing code aligns with Open Science principles, *potentially improving funding prospects*. 
Effective **commenting** is essential for understanding; focus on commenting **code chunks** rather than every line.

**Incorporate README file:**

- **Describe all contents** of the code supplement.
- Provide clear, concise **instructions for reproducing** all results.
- **Avoid** extensive **references to** settings or **details** of the analysis; those reproducing may not be acquainted with all paper details.
- Instead, clearly delineate which scripts produce which figures and tables, **referencing result names** (e.g., **Table 1, Figure 1**) in both the README and the code.
Organize files using a **clear folder structure** such as Code, Data, Results.

**Save** final results in a **readable format**, e.g. figures as PDFs and tables as Excel files (or data.frames in R).

Ensure **output** figures and tables **match** their **appearance** and structure in the **paper**.

**Limit** the **number of** scripts and output **files** to avoid confusion:

- Combine similar scripts with proper commenting.
- Combine all R functions used in the analysis into a single script.
- Save all iterations of a simulation in a single (.Rda) file.

For clarity, **store functions** used in the analysis in **separate scripts**, sourced in the analysis script.
Code Formatting & Naming Conventions

- **Avoid absolute paths; use relative paths** such as “code/results/figure1.pdf” instead of “C:/Users/Sepp/Arbeit/code/results/figure1.pdf”.

- Choose **descriptive file names** like “simulation.R” over ambiguous ones like “rs_rfg_3.R”.

- Use **underscores instead of spaces in file and folder names**.

- Ensure **code is well-spaced and properly indented** for ease of reading.
Achieving Reproducibility

- Ensure **every figure and table** presented in the paper is reproducible.

- For analyses involving randomness, **set the seed** of the random number generator to ensure reproducibility.

- Software is constantly updated. ⇒ **Detail** the **versions of the software packages** and the system used, e.g., via the output of `sessionInfo()` in R.

- **Conduct reproducibility checks** by running individual scripts; ideally have a **project partner** reproduce results to uncover potential errors.
Importance of Intermediate Results

- **Save** not only the final but also intermediate results:
  - Use one script for calculations and intermediate results storage.
  - Use another to evaluate the intermediate results, producing final results (figures, tables).
  - Keep the results of each iteration available and reproducible in simulations, e.g. by setting separate seeds.

- Intermediate results allow for reproducibility spot checks without repeating the entire analysis, crucial for computationally expensive analyses.

- They also allow analysts to make changes or additions at a later time, such as editing or adding figures.
Publication

■ **Preferred:** Ideally, the code should be published as a supplement on the journal’s homepage alongside the paper.

■ **GitHub:** An acceptable alternative. Crucially, specify the commit in the paper since repositories are subject to changes.

■ For supplements with large intermediate results, use platforms like figshare.

■ Strive to make data available. If unfeasible, provide pseudo data.

■ Combine all files into a single, organized folder, regardless of the publication method chosen.
Take Home Messages

- Availability of **code** ...
  - ... serves as **empirical proof of results**.
  - ... facilitates **reuse** and further development.
  - ... potentially improves **funding opportunities**.

- **Clarity and organization**: clear folder structures, README with clear instructions, few files, clear alignment of code with results in the paper.

- **Intermediate results** enable easier reproducibility and later modifications.

- **Final reproducibility spot checks** to catch potential errors **before** publication.

- If possible, **publish code** as a supplement **with the paper** to ensure long-term accessibility.
Thank you for your attention!