Project Organization

A guide on how to organize a reproducible research project

Florian Hartig, University of Regensburg

This documents gives advice to ensure good project organization and reproducibility in a research project.

Table of contents

Introduction

The purpose of this project template is to suggest a project structure that

1. Keeps your research project ordered
2. Makes sure you have everything that is part of a research project covered and in one place
3. Ultimately makes sure that your research is technically valid and reproducible

Although this sounds easy, experience shows that it is a pervasive problems in science, and most real projects are not fully documented and reproducible. To get an idea about the typical problems and solutions, read:

- Cooper, N. (2017). A guide to reproducible code in ecology and evolution. link

- Alston, J. M., & Rick, J. A. (2021). A beginner's guide to conducting reproducible research. Bulletin of the Ecological Society of America, 102(2), 1-14. link

Further important global steps:

- Decide on a sensible folder structure (suggestion below)
- Put your entire folder structure under version control (usually GIT, coupled with a repository such as GitHub or GitLab).
- If you work with an IDE such as RStudio, create a project. The best idea is typically to create a project at the top level of your folder structure. Alternatively, some times it can make sense to have projects in subfolders (e.g. when you work with different programming languages / IDEs).

Folder Structure

In the project template, I suggest a particular folder structure. It can in some cases make sense to deviate from this structure, but if you do so, please consider why and if this really makes your project more orderly / reproducible.

Literature

Put your reference manager database with all papers that you read for your research project here!

- Use the reference manager and pdfs to make notes about what you read
- Make sure directly that your reference managers links correctly to your writing program

1) For LaTeX, make sure you export (ideally automatically) to BibTex and link to that
2) For Word, make sure the link works

Recommendations:

- For most people, use Zotero
- Open source people that work heavily with LaTeX: use JabRef

Try out that everything works by using your first reference in your thesis / publication

Lab Notebook

You should keep a journal / record of what you do daily.

1. For laboratory projects, this is typically organized in a lab notebook, and most labs have specific standards for the notebook. Here a typical recommendation but please follow the standards in your lab
2. For software projects, including all work done in the TE lab, a written lab notebook is less helpful because it does not integrate with the committ / issue tracker on code. However, it is still useful to have a record of daily work / interim results / ideas. What I recommend is to have a text or notebook file, where you daily track progress. Make sure this integrates with your issue tracker / version control! You can use any format, including OneNote, but make sure the notebook is kept locally and not on the cloud, and is included in the version control software.

Data

1. Place your data in this folder, in the most "raw" format possible. raw = if you received paper, scan the paper and place it here
2. Note the exact origin of your data, e.g. when it was downloaded, how it was obtained.
3. If you pre-process your data (e.g.) digitizing something, keep an exact record of how this was done
4. If you type in data, keep in mind: ideal format for stats programs is each variable a new column, each observation a new row.
5. Assign sensible/ recognizable variable and file names, and use a logical folder structure, and document this!

Analysis

This folder should contain the complete code to create the results of your project here. Can have subfolders

- Read lab / lecture about reproducibility
- Read lab / lecture about good coding practice

Special Comments for R projects or other script-based analysis:

- Use numbers in file names in the order in which they should be executed
- Use separate scripts for cleaning, data preparation, and different analyses
- Files should save all outputs automatically

Example:

1. 01-readingInData.R -> collects data and saves output
2. 02-dataCleaning.R -> saves cleaned data for analysis
3. 03-Analysis1.R -> produces Fig 1
4. 03-Analysis2.R -> produces Fig 2
5. 03-Analysis3.R -> produces Fig 3

Note that the last 3 files get the same training number, because they don't depend on each other.

Coding style:

- Script everything. No point and click, ever. If you absolutely have to, add protocols for all mouse clicks you do
- Scripts should be human-readable. Use comments and good variable names, follow a consistent coding style, have your scripts logically organized, use functions when appropriate
- Never repeat yourself (NRY) - use functions instead of doing the same thing with code in two different places of the script
- Expect that you have made an error. Build in sanity checks / asserts, either in the code, or as unit tests. Put in frequent plots / summaries, but this only helps if you also look at them!

Read also:

Filazzola, A., & Lortie, C. J. (2022). A call for clean code to effectively communicate science.
Methods in Ecology and Evolution, 13(10), 2119-2128.[link]{https://besjournals.onlinelibrary.wiley.com/doi/full
210X.13961}

Paper

Text of your paper / thesis / project here

- Don't use version1.docx, version2.dox - if you work with version control, you have a record of all your versions!