Best Practice:

This is the summary of the best practice from the Wilson2017.pdf(which is available in the same directory to view and download)

Author: Hesam Sadri

Contact info:

email : hsadri@chapman.edu

github : hesam-sadri

We need to follow some guidelines to achieve efficiency and reliable and reproducible results in the programming and software coding. We need organized data , good Documentation of every step.

Data management:

- Save the original(raw-data) and have a copy of it in some other device.
- Change the format of the file and variable names, and deal with missing values(NA), use a unique identifier for each and every sample to have a much clearer and easy to manipulate data file. Do not forget to record and explain all the steps you took to get the processed data.
- Share your processed data and your approach with others(Github)

Software:

The goal here is to make a readable, reusable, and testable codes:

- Use simple and short functions with precise inputs and outputs.
- Functions should have meaningful names.
- Place a brief explanatory comment at the start (use an example to show what the process is doing).
- Use well-maintained libraries for what you need.
- It would be good practice to add a test example, so people could test the program's functionality and compare their results with the correct one.

- Add your codes to some publicly available repository and make it easy for people to jump in and collaborate on your codes and give you some feedback.
- Adding readme, collaboration, and license file would help new contributors know how they could improve the program.
- Make it clear to people how they are supposed to communicate with you and others in the group.
- For projects, use directories with the same name of the project; if projects have less than 50 percent in common(data and codes), split them into two separate projects.

Keep track of changes and have access to the history of your coding:

- There is countless beneficiary to have access to old versions of your program/data; you can use the previous versions to try out some new approach or correct some mistakes.
- Using version control tools like Github will allow you to have full access to previous versions and compare them, see who made the specific changes, or contribute to which functions. For this purpose, try to keep changes small and make meaningful comments on the changes.
- We can also use version control programs to keep track of the manuscripts.
- It would be useful to add and explain supplementary materials which have been used during the project life cycle.

There are lots of other worthy tools and approaches which could help us like: Branches, unit tests, continuous integration, profiling and performance tuning, and code review and pair programming.

Hopefully, by adopting these approaches, we would able to have more transparent, reliable, and reproducible results.