Best (Computational) Practices

Modern Techniques in Modelling

LONDON SCHOOL of HYGIENE &TROPICAL MEDICINE

Overview

Overview

- Why
- Where
- How

WHY?

LONDON SCHOOL *of* HYGIENE &TROPICAL MEDICINE

Discussion

LONDON SCHOOL of HYGIENE &TROPICAL MEDICINE

Discussion

- Verifiability
- Repeatability
- Flexability / Portability
- Maintainability

Exercise

Exercise

– What does the code in `bad.R` do?

– What could have been different to make the code easier to understand?

WHERE?

LONDON SCHOOL of HYGIENE &TROPICAL MEDICINE

Discussion

Discussion

Organize your project:

- have separate places (folders) for separate projects
- within a particular project, have separate places for input, code, figures, other outputs, etc.
- separate your code files into steps
- name files so that someone other than you could roughly guess their purpose
- consistency! whatever you decide is best for a particular project, do it that way as much as possible for that project.
- moderation! possible to overdo all the above

Exercise

LONDON SCHOOL of HYGIENE &TROPICAL MEDICINE

Exercise

- As previously discussed: `bad.R` - you're going to keep it working, but re-write it with best practices
- Recall: this code reads in some parameters, simulates a system, analyzes the results of those simulations, and then plots figures based on the analysis & results
- Without worrying about the low level code yet, create a structure to accommodate this process and move the relevant pieces of `bar.R` into the matching files
- Feel free to create folders, rename files, etc.

HOW?

LONDON SCHOOL of HYGIENE &TROPICAL MEDICINE

Discussion

LONDON
SCHOOL of
HYGIENE
&TROPICAL
MEDICINE

Discussion

– Comments

- start by writing out “pseudocode”: plain language description of the process / steps of your work
- given an overall picture (*what* + *where*), you can write down the details of each larger step directly in the files
- with the pseudocode in place, start writing the code by following that description; as you recognize additional detail needed to understand the code (or steps you forgot), expand your comments

Discussion

– Whitespace
  - Code is for *you* as well as the machine
  - using blank space effectively can help you read the code by indicating distinct sections of code (corresponding to distinct sub-steps), making declaration / assignment clearer, and by emphasizing evaluation blocks (versus debugging blocks)

Discussion

- Naming
  - Code is for *you* as well as the machine
  - naming variables, function arguments, etc can ensure you understand what that variable does

Discussion

– Don’ts

- re-use variable names (also applies to re-using the common base R names)
- have one script to rule them all
- copy anything more than once

Exercise

Exercise

– Go into each the files you made in the *Where Practical* and write the pseudocode for what that step is doing

– move the bad code to be along side the various steps

– rename variables, space for readability, etc until the code is highly comprehensible

– run it to verify you didn’t break anything