Writing good code

A short primer

Luce Skrabanek

3 October 2023
Consistency is key

Many style guides that will help guide you:

- Rstudio / Hadley Wickham – http://adv-r.had.co.nz/Style.html
Naming things

- Good names are the first step to good code.
  - `snake_case`
  - `lowerCamelCase`
  - `UpperCamelCase`
  - `leopard.case`

- Be consistent with the use of plurals

- Aim to be concise, but meaningful

- Use nouns to name variables

- Use verbs to name functions

- Avoid using names of existing functions or objects
Good indentation

- Essential for readable code
- RStudio will do it for you!
- Be consistent
- Code within curly braces `{}` should always be indented (and opening curly braces should not be on a line on their own)
- Split up very long lines onto multiple lines
Whitespace is your friend

- Place spaces around all infix operators (=, +, -, <-, etc)
- Put spaces after commas; don’t put spaces before commas, except when selecting whole columns: `data.frame[, column2]`
- Use empty lines to group chunks of code that are logically related
Comments

- At a minimum, there should be an explanatory comment before each function; detail the expected inputs and outputs, especially if your argument names are not obvious.
- Anything unclear should also get a clarifying comment.
- Cleverer / more concise code often requires better comments.
Functions and breaking code up

- Good code is broken up into functions
- Each function should do one well-defined thing
- Names of functions should tell you what they do (but use comments anyway)
- *Don’t copy code* (DRY: Don’t repeat yourself)
Performance

- Especially at first, worry about your code being clear and working first.
- For loops aren’t the fastest, but are easy to read and understand as you are learning.
  - Explicitly set the size of the final vector.
Performance: for-loop example

```r
runSimulation <- function(num_iterations = 100) {
  d <- data.frame(lower = numeric(num_iterations),
                  mean = numeric(num_iterations),
                  upper = numeric(num_iterations))
  for (i in 1:num_iterations) {
    x <- rnorm(10000)
    d[i, ] <- mean(x) + c(-1.96, 0, +1.96) * sd(x) / sqrt(length(x))
  }
  length(which(d$lower > 0 | d$upper < 0))
}
runSimulation0 <- function(num_iterations = 100) {
  d <- data.frame(lower = numeric(0),
                  mean = numeric(0),
                  upper = numeric(0))
  for (i in 1:num_iterations) {
    x <- rnorm(10000)
    d[i, ] = mean(x) + c(-1.96, 0, +1.96) * sd(x) / sqrt(length(x))
  }
  length(which(d$lower > 0 | d$upper < 0))
}
```

1000 iterations

10000 iterations