Practice writing R functions

That are relevant to simulation studies

Ian Hussey

06 Mai, 2024

Contents

General structure of a function

Note that this is pseudo-code only: chunk is set not to run (`eval=FALSE`).

```
#basics of functions

data <- "my string"

whatever_function_one <- function(data) {

#dependencies require(dplyr)

# tests/checks if(is.data.frame(data)){ stop("argument 'data' must be a data frame or tibble") }

# do stuff output <- data |> mutate(var=var + 1) # lets assume var exists

# do stuff

return(output)

}
```

```
# define function
function_name <- function(argument_1, # first arugment is often the data, if the function takes a data
                          argument_2 = "default", # arguments can have defaults
                          argument_3) {
  # required packages
  require(dplyr)

  # checks
  # well written functions contain checks.
  # e.g., if the function assumes that argument_1 is a data frame, check that this is the case.
  # note that it is more useful to write the function first and add checks later.
  if(!is.data.frame(argument_1)){
    stop("argument_1 must be a data frame")
  }

  # code that does things
  object_to_be_returned <- input_data_frame |>
    # do things
    mutate(value = value + 1)

  # object to be returned
  return(object_to_be_returned)
}

# use function
function_name()
```

Ideas for useful functions

Many of these functions have already been implemented by existing R packages (e.g., packages in the {easystats} universe). However, our goal is not merely to find an existing solution, but to write one ourselves for practice.

Relevant to generating data

HOW DOES ANY OF THE FOLLOWING INFLUENCE COHEN'S D IN ANYWAY

- Generate data from a uniform distribution and return a data frame.
- Generate bounded data, e.g., responses that are continuous but must be between 1 and 10. # a good question would be how bounding influences cohen's D
- Generate Likert data, e.g., responses that are whole numbers between 1 and 7, but which follow an underlying normal distribution.
- Simulate outliers, e.g., from careless responding or bots.
- Generate item-level data for cross sectional studies [useful but too complicated]

Relevant to analyzing data

- Convert a *p* value's significance to create APA-format table stars (i.e., “ ***” vs.*** ”” *vs.* ”” vs “ns”)

```
p_con <-function(p_value){

  #check results from data frame and convert

  if(p_value < .001){
    new_p <- "***"
  } else if(p_value < .01){
    new_p <- "**"
  } else if (p_value < .05){
    new_p <- "*"
  } else if(p_value >=.05){
    new_p <- "ns"
  }
```

```
    # return
  return(new_p)
}

p_value <- 0.01

p_con(p_value)
```

```
## [1] "*"
```

- Convert a Cohen's *d* estimate to an interpretation
- Fit a correlation test and extract the p value and correlation
- Fit a regression and extract key results (p values, Beta estimates, etc)
- Fit and extract Cronbach's alpha [requires item level data]
- Choose another kind of test, fit it and extract its key information (eg estimate, p value), such as an assumption test, so that we could simulate the utility of tests of assumptions.
- Simulate publication bias by labeling a given study as "published" or "unpublished" based on a combination of its *p* value and a defined probability of (non)significant studies being published or not.

Relevant to summarizing simation results across iterations

- Summarize a column of data into a string that summarizes its mean and SD, which could be pasted directly into a manuscript. i.e., taking the form "M = XX.X (SD = XX.X)", with rounding and retention of lagging zeros.
- A function that rounds all numeric variables in a data frame by a given number of places.

Session info

```
sessionInfo()
```

```
## R version 4.3.2 (2023-10-31 ucrt)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 11 x64 (build 22631)
##
## Matrix products: default
##
##
## locale:
## [1] LC_COLLATE=German_Switzerland.utf8  LC_CTYPE=German_Switzerland.utf8
## [3] LC_MONETARY=German_Switzerland.utf8 LC_NUMERIC=C
## [5] LC_TIME=German_Switzerland.utf8
##
## time zone: Europe/Zurich
## tzcode source: internal
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## loaded via a namespace (and not attached):
##  [1] compiler_4.3.2    fastmap_1.1.1     cli_3.6.2         tools_4.3.2
##  [5] htmltools_0.5.8.1 rstudioapi_0.16.0 yaml_2.3.8        rmarkdown_2.26
##  [9] knitr_1.46        xfun_0.43         digest_0.6.35     rlang_1.1.3
## [13] evaluate_0.23
```