Final Workflow Assignment

Mohammad Dastgheib

3/12/2022

Contents

This project is based on my master's dissertation script. The original work is in `.Rmd` format. Here, as a follow-up to my previous *self-critique* assignment, I will try to improve my code, as well as illustrating some parts of my code that already follows the goals of this assignment (efficiency, fidelity, sharing/reproducibility)

Before starting the assignment, I need to briefly describe the "original files": `thesisaltogether.Rmd` is the main file in my original work. `references.bib` and `Rreferences.bib` are bibtex formats for managing my references. `preamble-latex.tex` is an addendum to my YAML section to address some technical issues with the formatting of figures, etc. `Appendix`, `Figures`, and `Raw data` contain several required files to execute the `thesisaltogether.Rmd`. However, the original Raw Data of my work is already posted in my GitHub repo.

I. Efficiency

Automation

a function to optimize the package installation

In the original work, I inserted this function to reduce the amount of time re-installing/loading the packages.

```r
required_packages <- function(pkg){
  new.pkg <- pkg[!(pkg %in% installed.packages()[, "Package"])]
  if (length(new.pkg))
    install.packages(new.pkg, dependencies = TRUE)
  sapply(pkg, require, character.only = TRUE)
}

packages <- c("rio","readxl", "tidyverse","devtools","dplyr","ggplot2","magrittr","Hmisc","psycho","lme
required_packages(packages)
```

```
##       rio    readxl tidyverse  devtools     dplyr   ggplot2  magrittr     Hmisc
##      TRUE      TRUE      TRUE      TRUE      TRUE      TRUE      TRUE      TRUE
##    psycho  lmerTest  rstanarm    jtools bayesplot  corrplot
##      TRUE      TRUE      TRUE      TRUE      TRUE      TRUE
```

After examining my original work, I noticed that I had uploaded my “Raw Data” as an Excel spreadsheet. This means that unless the project is finished, every time I wanted to add an observation, I needed to replace the existing Excel file with a new one. This process can be improved if I insert the original works into a Google sheets file and R can have access to that file, which almost always contains the latest version of my raw data.

original piece from my work

```
# importing the main excel file from Github repository (mohdasti)
RAW_data <- rio::import('https://github.com/mohdasti/Queens-Thesis/blob/master/Raw%20data/RAW_data.xlsx
#View(RAW_data)
```

The improved version with access to the Google Sheets

I have been working on this piece of code for a while for another similar project. Here, I try to modify the code to show how it works. The code chunk is annotated.

```
#creating a function that gets the location of the file (e.g., link) and makes sure that the link is pu
create_Raw_object <-  function(data_location,
                               pdf_mode = FALSE,
                               sheet_is_publicly_readable = TRUE) {

  raw <- list(
    pdf_mode = pdf_mode,
    links = c()
  )

#using stringr function, I look for specific patters of the google docs link. If I found it, then store

  is_google_sheets_location <- stringr::str_detect(data_location, "docs\\.google\\.com")


  if(is_google_sheets_location){
    if(sheet_is_publicly_readable){

# if the original google docs's sharing settings is set to "anyone with link can view", then, it should

      googlesheets4::gs4_deauth()
    }

# if everything is fine, the the following function will run to read the google sheet. For the sake of

# Here, I imported four columns from that 'raw google sheets' and renamed them in my raw spreadsheet in

    read_gsheet <- function(sheet_id){
      googlesheets4::read_sheet(data_location, sheet = sheet_id, skip = 1, col_types = "c")
    }
    raw$Date  <- read_gsheet(sheet_id = "data")
    raw$Participant_ID        <- read_gsheet(sheet_id = "id")
    raw$GMean   <- read_gsheet(sheet_id = "gmean")
    raw$Age  <- read_gsheet(sheet_id = "age")
  }
knitr::opts_chunk$set(
  results='asis',
  echo = FALSE
)

# now, that I have assigned how I can detect the link and specified which columns to be selected and im
### the link is fake ###

library(magrittr) # For the pipe
Data <- create_data_object(
  data_location = "https://docs.google.com/spreadsheets/d/....",
  pdf_mode = params$pdf_mode
)
}
```

II. Fidelity

In terms of fidelity and implementing procedures that could reduce the additional manual copy/paste of different version of the data, the code above (using Google Sheets) can dramatically increase the fidelity of my work.

removing the hardcoding

Moreover, I went over my code and found that I used hard-coding approach to remove some observations.

```
#after importing the data from GitHub and renaming it RAW
RAW <- RAW_data
#creating subsets of data
MED <- RAW[which(RAW$Condition == 'MED'), ]
WAKE <- RAW[which(RAW$Condition == 'WAKE'), ]
NAP <- RAW[which(RAW$Condition == 'NAP'), ]
NAP_SWS <- RAW[which(RAW$percentSWS != 0), ]
NAP_noSWS <- RAW[which(RAW$percentSWS == 0), ]

# Calculating Geometric Mean and Difference of Medians (two main measures of performance for the declar
# True Positive rate
RAW$TPR <-
  RAW$`Hit ratio` / (RAW$`Hit ratio` + RAW$`False Alarm ratio`)
#True Negative rate
RAW$TNR <-
  RAW$`Correct Rejection ratio` / (RAW$`Correct Rejection ratio` + RAW$`Miss ratio`)
#calculating the G-Mean
RAW$GMean <-
  sqrt(RAW$TPR * RAW$TNR)

#Excluding outliers - creating modified datasets
## any data points that is beyond ±2 SD - these exculusion are specific to each task (one participant m
## those who failed to meet the general criteria of their treatment condition (i.e. falling asleep in M
## those who their number of arousals were beyond ±2 SD


#MED$GMean[abs(scale(MED$GMean)) > 2]
```

```
#WAKE$GMean[abs(scale(WAKE$GMean)) > 2]
#NAP_SWS$GMean[abs(scale(NAP_SWS$GMean)) > 2]
#NAP_noSWS$GMean[abs(scale(NAP_noSWS$GMean)) > 2]

#creating two separate datasets based on the memory task
RAW_declr <- RAW[-c(34,39,25,13,46),]
```

So, here I would like to avoid hard-coding:

```
MED_dplyr <- RAW %>% filter(Condition=='MED') %>% filter(abs(scale(GMean)) < 2)
WAKE_dplyr <- RAW %>% filter(Condition=='WAKE') %>% filter(abs(scale(GMean)) < 2)
NAP_SWS_dplyr <- RAW %>% filter(Condition=='NAP') %>% filter(percentSWS != 0) %>% filter(abs(scale(GMean
NAP_noSWS_dplyr <- RAW %>% filter(Condition=='NAP') %>% filter(percentSWS == 0) %>% filter(abs(scale(GM
```

a simple plot of observations

Here, using ggplot package, I will plot the overall distribution

```
#overall pattern of distribution
NAP_dplyr <- rbind(NAP_SWS_dplyr, NAP_noSWS_dplyr)
NAP_GMean <-
  data.frame(Condition = "NAP", GeometricMean = NAP_dplyr$GMean)
MED_GMean <-
  data.frame(Condition = "MED", GeometricMean = MED_dplyr$GMean)
WAKE_GMean <-
  data.frame(Condition = "WAKE", GeometricMean = WAKE_dplyr$GMean)
df_GMean <- rbind(NAP_GMean, MED_GMean, WAKE_GMean)
```

```
ggplot(df_GMean, aes(x = Condition, y = GeometricMean, color = Condition)) +
  geom_point(size = 4,
             alpha = 0.7,
             position = position_jitter(w = 0.1, h = 0)) +
  stat_summary(
    fun.y = mean,
    geom = "point",
    shape = 23,
    color = "black",
    aes(fill = Condition),
    size = 4
  ) +
  stat_summary(
    fun.ymin = function(x)
      (mean(x) - sd(x)
      ),
    fun.ymax=function(x)(mean(x)+sd(x)),
    geom="errorbar", width=0.1) + xlab("Condition") + ylab("Geometric Mean") +
    theme_apa()
```

III. Sharing/Reproduciblity

As described in my last assignment, I had posted the original raw data in the Open Science Framework. Plus, I wrote the whole dissertation in RMarkdown and uploaded that in my GitHub Repository. You can also

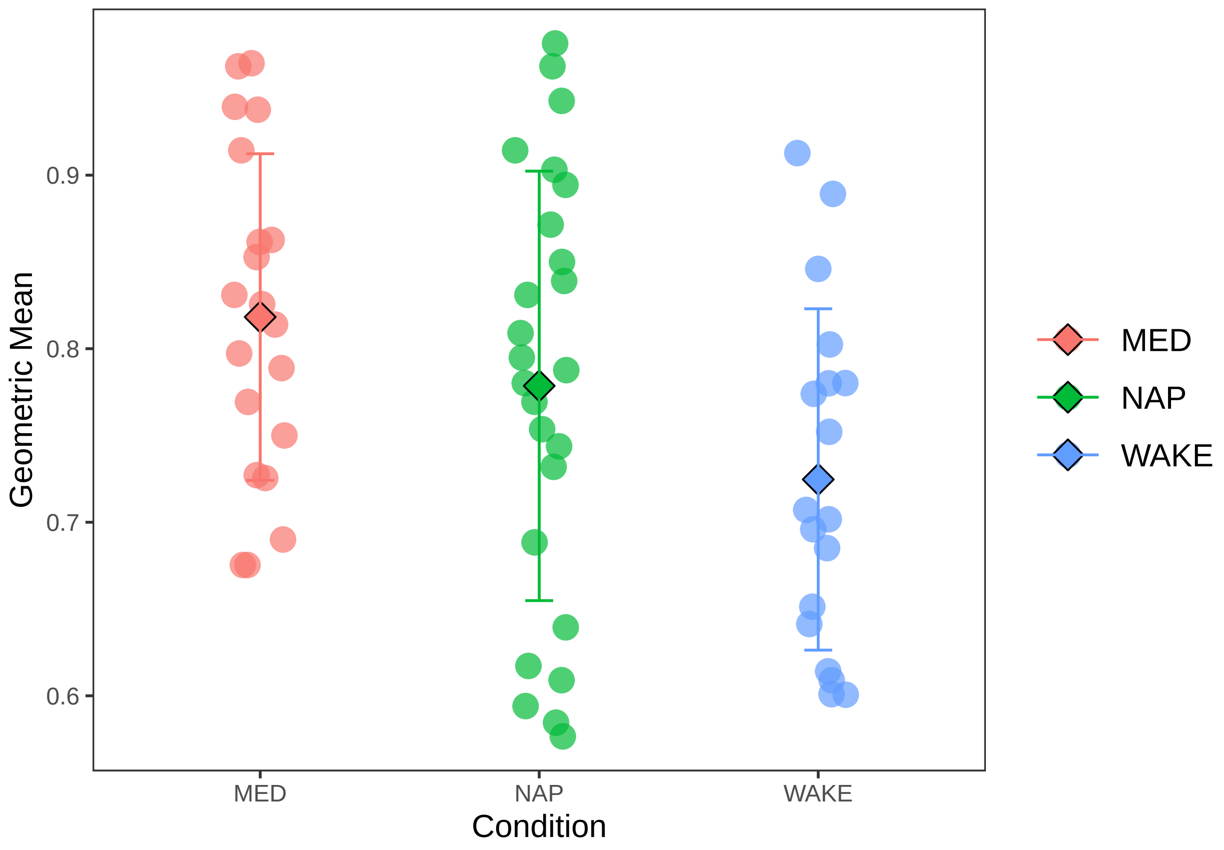

Figure 1: Overall distribution of Geometric Mean scores across the three experimental conditions (NAP; MED, meditation; WAKE). Error bars reflect standard deviation.

access the entire dissertation in this repository from `thesisaltogether.Rmd`.

The other issue that I briefly mentioned in my workflow critique was the usage of `dev` packages and how they might reduce the reproducibility. To avoid that, the `groundhog` package can bring a specific version of a package from a certain date. Below is a sample code chunk of groundhog—in case I needed to retrieve an older version of a package.

```r
# install.packages("groundhog")
# library(groundhog)
# groundhog.library("psycho","2018-12-26")
```