Response Rescoring

SGC3A-2-Response Rescoring

Amy Rae Fox

4/29/2022

Table of contents

Status

- 4/29/22 | ported existing .Rmd analysis files to Quarto (.qmd) for sharing status w/ JMH CMW via web

Part I

SGC3A

1 Harmonization

The purpose of this notebook is to harmonize data files for study SGC_3A.

Pre-Requisite	Followed By
spring17_clean_data.Rmd spring18_clean_data.Rmd fall21_clean_data.Rmd winter2022_clean_sgc3a.Rmd	2_sgc3A_rescoring.qmd

2 INTRODUCTION

Data for study SGC_3A were collected across four time periods, interrupted by the Covid-19 pandemic.

Period	Modality
Fall 2017	in person, SONA groups in computer lab
Spring 2018	in person, SONA groups in computer lab
Fall 2021	asynchronous, online, SONA
Winter 2022	asynchronous, online, SONA

Data collected in Fall 2017, Spring 2018 constitute the original SGC_3A study, conducted in person. Data collected in Fall 2021, Winter 2022 constitute the web-based replication, conducted online (asynchronously). In all cases, the experiment was administered via a web application.

3 HARMONIZATION

The underlying data structure of the stimulus web application changed across the data collection period, resulting in slightly different data files (i.e. columns are not named consistently). In this section, we combine the files from each data collection period into a single *harmonized* data file for analysis (one for participants, one for items).

3.1 Participants

First we import participant-level data from each data collection period, selecting only the columns relevant for analysis, and renaming columns to be consistent across each file. The result is a single data frame `df_subjects` containing one row for each subject (across all periods). Note that we *are not* discarding any *response* data. Rather, we discard columns that are automatically recorded by the stimulus web application and help the application run.

Note that we discard some columns representing scores calculated in the stimulus engine. These scores were calculated differently across collection periods, and so we discard them and recalculate scores in the next analysis notebook.

```
#IMPORT PARTICIPANT DATA

#set datafiles
fall17 <- "data/session-level/fall17_sgc3a_participants.csv"
spring18 <- "data/session-level/spring18_sgc3a_participants.csv"
fall21 <- "data/session-level/fall21_sgc3a_participants.csv"
winter22 <- "data/session-level/winter22_sgc3a_participants.rds"

#read datafiles, set mode and term
df_subjects_fall17 <- read.csv(fall17) %>% mutate(mode = "lab-synch", term = "fall17")
df_subjects_spring18 <- read.csv(spring18) %>% mutate(mode = "lab-synch", term = "spring18
df_subjects_fall21 <- read.csv(fall21) %>% mutate(mode = "online-asynch", term = "fall21")
df_subjects_winter22 <- read_rds(winter22) #use RDS file as it contains metadata

#SAVE METADATA FROM WINTER, but no rows
df_subjects <- df_subjects_winter22 %>% filter(condition=='X') %>% select(
  subject,condition,term,mode,
  gender,age,language, schoolyear, country,
  effort,difficulty,confidence,enjoyment,other,
  totaltime_m,absolute_score
)

#reduce data collected using OLD webapp to useful columns
df_subjects_before <- rbind(df_subjects_fall17, df_subjects_spring18, df_subjects_fall21)
  #rename and summarize some columns
  mutate(
    totaltime_m = totalTime / 1000 / 60,
    absolute_score = triangular_score,
    language = native_language,
    gender = sex,
    schoolyear = year) %>%
  #create placeholders for cols not collected until NEW webapp [for later rbind]
  mutate(
    effort = "NULL",
    difficulty = "NULL",
    confidence = "NULL",
    enjoyment = "NULL",
    other = "NULL",
    disability = "NULL"
  ) %>%
  #select only columns we'll be analyzing, discard others
  dplyr::select(subject, condition, term, mode,
                #demographics
                gender, age, language, schoolyear, country,
                #placeholder effort survey
                effort, difficulty, confidence, enjoyment,
                #placeholder misc
                other, disability,
                #response characteristics
                totaltime_m, absolute_score)

#save 'explanation' columns from winter22, which is actually a response to a free response
df_winter22_q16 <- df_subjects_winter22 %>%
  select(subject, condition, term , mode, explanation) %>%
  mutate(
    q = 16,
    response = explanation
  ) %>% select(-explanation)

#reduce data collected using NEW webapp to useful columns
df_subjects_winter22 <- df_subjects_winter22 %>%
  mutate(score = absolute_score) %>%
  #select only columns we'll be analyzing, discard others
  dplyr::select( subject, condition, term, mode,
                 #demographics
                 gender, age, language, schoolyear, country,
                 #effort survey
                 effort, difficulty, confidence, enjoyment,
                 #explanations
                 other,disability,
                 #response characteristics
                 totaltime_m, absolute_score)


effort_labels <- c("I tried my best on each question", "I tried my best on most questions"

#combine dataframes from old and new webapps
df_subjects <- rbind(df_subjects, df_subjects_winter22, df_subjects_before) %>%
  #refactor factors
  mutate (
    subject = factor(subject),
    condition = factor(condition),
    term = factor(term),
    mode = factor(mode),
    gender = factor(gender),
    schoolyear = as.factor(schoolyear)
  )

#FIX METADATA
#Add metadata for columns that lost it [factors, for some reason!]
var_label(df_subjects$subject) <- "ID of subject (randomly assigned in stimulus app)."
var_label(df_subjects$condition) <- "ID indicates randomly assigned condition (111 -> cont
var_label(df_subjects$term) <- "indicates if session was run with experimenter present or
var_label(df_subjects$mode) <- "indicates mode in which the participant completed the stud
var_label(df_subjects$gender) <- "What is your gender identity?"
var_label(df_subjects$schoolyear) <- "What is your year in school?"

#CLEANUP
```

```
rm(df_subjects_fall17,df_subjects_fall21, df_subjects_spring18, df_subjects_winter22,df_su
rm(fall17,fall21,spring18,winter22)
```

3.2 Items

Next we import item-level data from each data collection period, selecting only the columns relevant for analysis, and renaming columns to be consistent across each file. The result is a single data frame `df_items` containing one row for each *graph comprehension task question* (qs=15) (across all periods). A second data frame `df_freeresponse` contains one row for each free response strategy question (last question posed to participants in Winter2022) Note that we *do not* discard any *response* data. Rather, we *do* discard several columns representing accuracy scores for responses that were calculated in the stimulus engine. These scores were calculated differently across collection periods, and so we discard them and recalculate scores in the next analysis notebook. Original response data are always preserved.

```
#set datafiles
fall17 <- "data/session-level/fall17_sgc3a_blocks.csv"
spring18 <- "data/session-level/spring18_sgc3a_blocks.csv"
fall21 <- "data/session-level/fall21_sgc3a_blocks.csv"
winter22 <- "data/session-level/winter22_sgc3a_items.rds"

#read datafiles, set mode and term
df_items_fall17 <- read.csv(fall17) %>% mutate(mode = "lab-synch", term = "fall17")
df_items_spring18 <- read.csv(spring18) %>% mutate(mode = "lab-synch", term = "spring18")
df_items_fall21 <- read.csv(fall21) %>% mutate(mode = "online-asynch", term = "fall21")
df_items_winter22 <- read_rds(winter22) #use RDS file as it contains metadata

#get mapping being question # and interval relation the question tests, that is encoded on
map_relations <- df_items_winter22 %>% group_by(q) %>% select(q,relation) %>% unique()


#SAVE METADATA FROM WINTER, but no rows
df_items <- df_items_winter22 %>% filter(condition=='X') %>% select(
  subject,condition,term,mode,
  question, q, answer, correct, rt_s
)

#reduce data collected using old webapp
df_items_before <- rbind(df_items_fall17, df_items_spring18, df_items_fall21) %>%
  mutate(rt_s = rt / 1000, correct = as.logical(correct)) %>%
  select(subject, condition, term, mode, question, q, answer, correct, rt_s)

#reduce data collected using new webapp
df_items_winter22 <- df_items_winter22 %>%
  select(subject, condition, term, mode, question, q, answer, correct, rt_s) %>% #unfactor
  mutate(
    subject = as.character(subject),
    condition = as.character(condition),
    term = as.character(term),
    mode = as.character(mode),
    q = as.integer(q),
    correct = as.logical(correct)
  )

#combine dataframes from old and new webapps
df_items <- rbind(df_items, df_items_winter22,df_items_before) %>%
  #refactorize columns
  mutate(
    subject = factor(subject),
    condition = factor(condition),
    term = factor(term),
    mode = factor(mode),
    q = as.integer(q)) %>%
  #rename answer column to RESPONSE
  rename(response = answer) %>%
  #remove all commas and make as character string
  mutate(
    response = str_remove_all(as.character(response), ","),
    num_o = str_length(response)
  )


#FIX METADATA
#Add metadata for columns that lost it [factors, for some reason!]
var_label(df_items$subject) <- "ID of subject (randomly assigned in stimulus app)."
var_label(df_items$condition) <- "ID indicates randomly assigned condition (111 -> control
var_label(df_items$term) <- "indicates if session was run with experimenter present or asy
var_label(df_items$mode) <- "indicates mode in which the participant completed the study"
var_label(df_items$q) <- "Question Number (in order)"
var_label(df_items$correct) <- "Is the response (strictly) correct? [dichotomous scoring]"
var_label(df_items$response) <- "options (datapoints) selected by the subject"
var_label(df_items$num_o) <- "number of options selected by the subject"

#HANDLE FREE RESPONSE QUESTION #16
#save `free response` Q#16 in its own dataframe
df_freeresponse <- df_items %>% filter(q == 16) %>% select(-question,-correct,-rt_s,-num_o
#add data from wi22 [stored on subject data]
df_freeresponse <- rbind(df_freeresponse, df_winter22_q16)
#add question description
df_freeresponse <- df_freeresponse %>% mutate(
    question = "Please describe how to determine what event(s) start at 12pm?",
    response = as.character(response) #doesn't need to be factor
  )
#remove 'free response' Q#16 from df_items
df_items <- df_items %>% filter (q != 16)

#CLEANUP
rm(df_items_fall17,df_items_fall21, df_items_spring18, df_items_winter22, df_items_before,
rm(fall17,fall21,spring18,winter22, map_relations)
```

3.2.1 Validation

Next, we validate that we have the complete number of item-level records based on the number of subject-level records

```
#the number of items should be equal to 15 x the number of subjects
nrow(df_items) == 15* nrow(df_subjects) #TRUE
```

```
[1] TRUE
```

```
#each subject should have 15 items
df_items %>% group_by(subject) %>% summarise(n = n()) %>% filter(n != 15) %>% nrow() == 0
```

```
[1] TRUE
```

4 EXPORT

Finally, we export the (session-harmonized) data for analysis, as CSVs, and .RDS (includes metadata)

```
#SAVE FILES
write.csv(df_subjects,"data/sgc3a_participants.csv", row.names = FALSE)
write.csv(df_items,"data/sgc3a_items.csv", row.names = FALSE)
write.csv(df_freeresponse,"data/sgc3a_items.csv", row.names = FALSE)

#SAVE R Data Structures
#export R DATA STRUCTURES (include codebook metadata)
rio::export(df_subjects, "data/sgc3a_participants.rds") # to R data structure file
rio::export(df_items, "data/sgc3a_items.rds") # to R data structure file
```

5 RESOURCES

```r
sessionInfo()
```

```
R version 4.0.2 (2020-06-22)
Platform: x86_64-apple-darwin17.0 (64-bit)
Running under: macOS  10.16

Matrix products: default
BLAS:   /Library/Frameworks/R.framework/Versions/4.0/Resources/lib/libRblas.dylib
LAPACK: /Library/Frameworks/R.framework/Versions/4.0/Resources/lib/libRlapack.dylib

locale:
[1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
 [1] codebook_0.9.2  forcats_0.5.0   stringr_1.4.0   dplyr_1.0.2
 [5] purrr_0.3.4     readr_1.4.0     tidyr_1.1.2     tibble_3.1.2
 [9] ggplot2_3.3.5   tidyverse_1.3.0

loaded via a namespace (and not attached):
 [1] Rcpp_1.0.5        lubridate_1.7.9   assertthat_0.2.1  digest_0.6.27
 [5] utf8_1.2.1        R6_2.5.0          cellranger_1.1.0  backports_1.2.1
 [9] reprex_0.3.0      labelled_2.8.0    evaluate_0.14     httr_1.4.2
[13] pillar_1.6.1      rlang_0.4.11      curl_4.3          readxl_1.3.1
[17] rstudioapi_0.13   data.table_1.13.2 blob_1.2.1        rmarkdown_2.11
[21] foreign_0.8-80    munsell_0.5.0     broom_0.7.12      compiler_4.0.2
[25] modelr_0.1.8      xfun_0.29         pkgconfig_2.0.3   htmltools_0.5.2
[29] tidyselect_1.1.0  rio_0.5.16        fansi_0.5.0       crayon_1.4.1
[33] dbplyr_1.4.4      withr_2.4.2       grid_4.0.2        jsonlite_1.7.1
[37] gtable_0.3.0      lifecycle_1.0.0   DBI_1.1.0         magrittr_2.0.1
[41] scales_1.1.1      zip_2.1.1         cli_3.3.0         stringi_1.7.3
```

```
[45] fs_1.5.0          xml2_1.3.2        ellipsis_0.3.2    generics_0.0.2
[49] vctrs_0.3.8       openxlsx_4.2.3    tools_4.0.2       glue_1.6.2
[53] hms_0.5.3         fastmap_1.1.0     yaml_2.2.1        colorspace_2.0-2
[57] rvest_0.3.6       knitr_1.37        haven_2.3.1
```

6 Response Rescoring

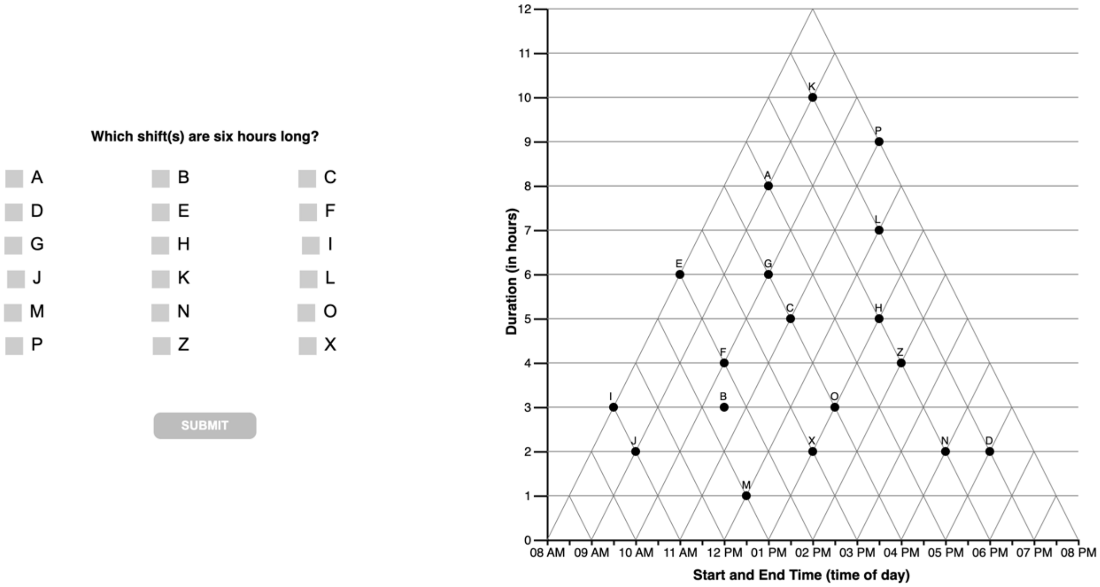
The purpose of this notebook is to re-score the response accuracy data for the SGC_3A study. This is required because the question type on the graph comprehension task used a 'Multiple Answer Multiple Choice' design (MCMA). Warning: this notebook takes several minutes to execute.

Pre-Requisite	Followed By
1_sgc3a_harmonize.qmd	3_sgc3A_descriptives.qmd

```r
#read datafiles, set mode and term
df_items <- read_rds('data/sgc3a_items.rds')
```

7 INTRODUCTION

The *graph comprehension task* of study SGC 3A presents readers with a graph, a question, and a series of checkboxes. Participants are instructed to use the graph to answer the question, and respond by selecting all the checkboxes that apply, where each checkbox corresponds to a datapoint in the graph.



Which shift(s) are six hours long?

Figure 7.1: **Figure 1. Sample Graph Comprehension (Question # 6)**

In the psychological and education literatures on Tests & Measures, the format of this type of question is referred to as *Multiple Choice Multiple Answer* (MCMA) or *Multiple Answer Multiple Choice* (MAMC).
It has a number of properties that make it different from traditional *Single Answer Multiple Choice* (SAMC) questions, where the respondent marks a single response from a number of options In particular, there are a number of very different ways that MAMC questions can be *scored*.

Traditionally in SAMC questions, one point is given for selecting the option designated as correct, and zero points given for marking any of the alternative (i.e. distractor) options.

Responses on MAMC questions, however might be partially correct (i), while responses on other answer options within the same item might be incorrect (n–i). In MAMC, it is not obvious how to allocate points when the respondent marks a true-correct option (i.e. options that *should* be selected), as well as one or more false-correct options (i.e. options that *should not* be selected). Should partial credit be awarded? If so, are options that respondents false-selected and false-unselected items equally penalized?

Schmidt et. al (2021) performed a systematic literature review of publications proposing MAMC (or equivalent) scoring schemes, ultimately synthesizing over 80sources into 27 distinct scoring approaches. Upon reviewing the benefits of tradeoffs of each approach, for this study we choose utilize two of the schemes: **dichotomous scoring** (Schmidt. et. al scheme #1), and **partial scoring** $[-1/q, 0, +1/p]$ (Schmidt et. al. scheme #26), as well as a scaled **discriminant score** that leverages partial scoring to discriminate between strategy-specific patterns of response.

7.1 Response Encoding

First, we note that the question type evaluated by Schmidt et. al. (2021) is referred to as *Multiple True-False* (MTF), a variant of MAMC where respondents are presented with a question (stem) and series of response options with True/False (e.g. radio buttons) for each. Depending on the implementation of the underlying instrument, it may or may not be possible for respondents to *not respond* to a particular option (i.e. leave the item 'blank'). Although MTF questions have a different underlying implementation (and potentially different psychometric properties) they are identical in their mathematical properties; that is, responses to a MAMC question of 'select all that apply' can be coded as a series of T/F responses to each response option

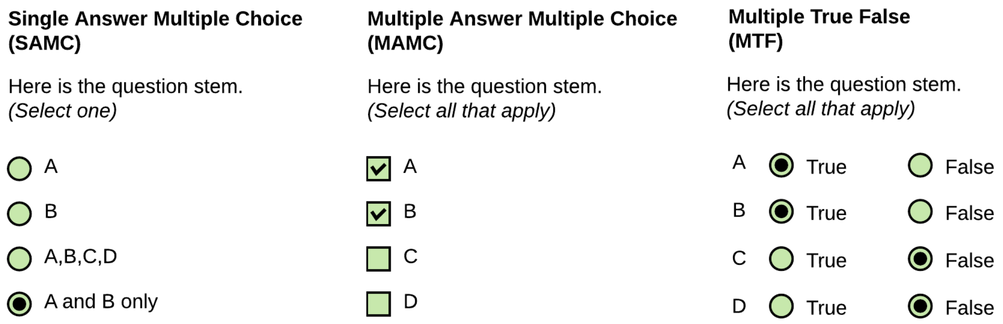


Figure 7.2: **Figure 2. SAMC (vs) MAMC (vs) MTF**