Spring 2018 Data Cleaning

Amy Rae Fox

11/2/2021

The purpose of this file is processing the combined data files for Spring 2018 into study-level files that contain only valid data for analysis, excluding invalid sessions and conditions.

Data is imported from 2 files, indicating two levels of analysis: participants and blocks (item-level). **Note: mouse-cursor data contained in final_mouse_blocks.json file is not handled here.**

```
#IMPORT DATA
df_participants <- fromJSON("combined_files/final_participants.json")
df_blocks <- fromJSON('combined_files/final_blocks.json')

#add term indicator
df_participants$term <- "spring18"
df_blocks$term <- "spring18"
```

```
#create factors in PARTICIPANTS
df_participants <- df_participants %>%
  select(subject,session,term,condition,  #re-arrange columns
         ts_n, tt_n,triangular_score,
         os_n, ot_n,orthogonal_score,
         explicit,impasse,axis,
         triangular_time, totalTime, ts_t, tt_t,
         attn_check,
         native_language, year, major, country, sex, age
         ) %>% #reorder columns
  mutate( #create factors and remove extraneous ""
    subject=factor(subject),
    condition=factor(condition),
    session=factor(session),
    term=factor(term),
    explicit=factor(explicit),
    axis=factor(axis),
    impasse=factor(impasse),
    sex = as.factor(gsub('"',"",sex)),
    age = as.double(gsub('"',"",age)),
    country = gsub('"',"",country),
    major = gsub('"',"",major),
    year = gsub('"',"",year),
    native_language = gsub('"',"",native_language),
)
```

```
df_blocks <- df_blocks %>%
  select( #reorder columns
    subject, session, term, condition,
    q,question,answer,rt,
    correct, orth_correct,
    explicit, impasse, axis) %>%
  mutate(
    subject=factor(subject),
    condition=factor(condition),
    session=factor(session),
    term=factor(term),
    explicit=factor(explicit),
    axis=factor(axis),
    impasse=factor(impasse),
    q=factor(q),
    question=factor(question)
  )
```

Sessions

The (string) session code is entered by the participant based on instructions given by the experimenter, and documents the data-collection session (eg. in-person at a particular time). This code is also used by the experimenter to differentiate test or expert data collection runs.

```
#MANUALLY INSPECT sessions
df_participants %>% group_by(session) %>%
  summarize(n=n())
```

```
## # A tibble: 41 x 2
##    session     n
##    <fct>   <int>
##  1 alpha       4
##  2 bravo      23
##  3 charlie    13
##  4 delta       3
##  5 echo        7
##  6 fire        5
##  7 flower      1
##  8 foxtrot     1
##  9 go1f        1
## 10 golf        6
## # ... with 31 more rows
```

In Spring 2018, 36 (regular)data collection sessions were used, from ALFA -> ZULU.

```
#manually recode sessions in participants
df_participants$session <- recode(df_participants$session,
                                  'go1f'='golf',
                                  't00'="too",
                                  'taine'='thine',
                                  'mouse'='MOUSE')
```

```
#manually recode sessions in blocks
df_blocks$session <- recode(df_blocks$session,
                                  'go1f'='golf',
                                  't00'="too",
                                  'taine'='thine',
                                  'mouse'='MOUSE')

df_participants %>% group_by(session) %>%
  arrange(desc(session)) %>%
  summarize(n=n())
```

```
## # A tibble: 38 x 2
##    session     n
##    <fct>   <int>
##  1 alpha       4
##  2 bravo      23
##  3 charlie    13
##  4 delta       3
##  5 echo        7
##  6 fire        5
##  7 flower      1
##  8 foxtrot     1
##  9 golf        7
## 10 heaven      3
## # ... with 28 more rows
```

Participants who mispelled their sessions have been manually recoded, and one participant erroneously entered their condition code as their session code, and this entry is corrected to 'XTRA'.

Conditions

The three digit condition code is entered by the participant based on instructions given by the experimenter, and determines the stimulus that the participant experiences during the study.

```
df_participants %>% group_by(condition) %>%
  summarize(n=n())
```

```
## # A tibble: 11 x 2
##    condition      n
##    <fct>      <int>
##  1 "111"         39
##  2 "112"          6
##  3 "113"         52
##  4 "114"         38
##  5 "114\n114"     1
##  6 "115"         41
##  7 "115-late"     2
##  8 "121"         45
##  9 "211"         15
## 10 "311"         28
## 11 "311\n311"     1
```

```
#SET CONDITION FACTORS FOR EACH STUDY
#SGC3A is the simple insight study, control (111) vs impasse (121)
f_sgc3a <- c(111,121)

#SGC3B is the factorial insight study (111 control, 121 insight, 211 static, 221 static-impasse, 311 ix

f_sgc3b <- c(111,121,211,221,311,321)

#SGC4 is the gridlines study 111, 112, 113
f_sgc4 <- c(111,112,113)
```

In Spring 2018, data were gathered for three study designs: SGC3B (continued data collection: full factorial insight vs. explicit) and SGC4(partial data collected:gridlines).

A few students duplicate-entered their condition codes. I verified that these codes still yield valid experimental stimuli. These codes are now manually recoded.

```
#manually recode sessions in participants
df_participants$condition <- recode(df_participants$condition,
                                    '114\n114'='114',
                                    '115-late'="115",
                                    '311\n311'='311')

#manually recode sessions in blocks
df_blocks$condition <- recode(df_blocks$condition,
                                   '114\n114'='114',
                                    '115-late'="115",
                                    '311\n311'='311')

df_participants %>% group_by(condition) %>%
  arrange(desc(condition)) %>%
  summarize(n=n())
```

```
## # A tibble: 8 x 2
##   condition     n
##   <fct>     <int>
## 1 111          39
## 2 112           6
## 3 113          52
## 4 114          39
## 5 115          43
## 6 121          45
## 7 211          15
## 8 311          29
```

Finally, data from the master participants and blocks files are segregated into separate files for each individual study, separated by condition.

The session 'MOUSE' was used for participants in the retrospective-narrative SGC3N study. The sessions 'strategy-connecting' and 'strategy-satisficing' were used to demonstrate strategies arising from the SGC3N study.

```
#CREATE SGC3N dataframes
df_sgc3N_participants <- df_participants %>%
  filter(session %in% c("MOUSE","strategy-satisficing","strategy-connecting"))

df_sgc3N_blocks <- df_blocks %>%
  filter(session %in% c("MOUSE","strategy-satisficing","strategy-connecting"))

df_sgc3N_participants %>% group_by(condition) %>%
  arrange(desc(condition)) %>%
  summarize(n=n())
```

```
## # A tibble: 2 x 2
##   condition     n
##   <fct>     <int>
## 1 111           4
## 2 121           8
```

```
#WRITE SGC3N files
write.csv(df_sgc3N_participants,"study_files/spring18_sgc3N_participants.csv", row.names = FALSE)
write.csv(df_sgc3N_blocks,"study_files/spring18_sgc3N_blocks.csv", row.names = FALSE)

#REMOVE SGC3N participants from main dataframes
df_participants <- df_participants %>%
  filter(!session %in% c("MOUSE","strategy-satisficing","strategy-connecting"))

df_blocks <- df_blocks %>%
  filter(!session %in% c("MOUSE","strategy-satisficing","strategy-connecting"))
```

These sessions are removed from the main dataframe and stored as separate files prefixed SGC3N_.

```
#SEPARATE PARTICIPANTS FILES
df_sgc3a <- df_participants %>% filter (condition %in% f_sgc3a)
df_sgc3a %>% group_by(condition) %>%
  summarize(n=n())
```

```
## # A tibble: 2 x 2
##   condition     n
##   <fct>     <int>
## 1 111          35
## 2 121          37
```

```
write.csv(df_sgc3a,"study_files/spring18_sgc3a_participants.csv", row.names = FALSE)

df_sgc3b <- df_participants %>% filter (condition %in% f_sgc3b)
df_sgc3b %>% group_by(condition) %>%
  summarize(n=n())
```

```
## # A tibble: 4 x 2
##   condition     n
##   <fct>     <int>
## 1 111          35
## 2 121          37
```

```
## 3 211          15
## 4 311          29
```

```
write.csv(df_sgc3b,"study_files/spring18_sgc3b_participants.csv", row.names = FALSE)

df_sgc4 <- df_participants %>% filter (condition %in% f_sgc4)
df_sgc4 %>% group_by(condition) %>%
  summarize(n=n())
```

```
## # A tibble: 3 x 2
##   condition     n
##   <fct>     <int>
## 1 111          35
## 2 112           6
## 3 113          52
```

```
write.csv(df_sgc4,"study_files/spring18_sgc4a_participants.csv", row.names = FALSE)

#SEPARATE BLOCKS FILES
df_sgc3a <- df_blocks %>% filter (condition %in% f_sgc3a)
df_sgc3a %>% group_by(condition) %>%
  summarize(n=n())
```

```
## # A tibble: 2 x 2
##   condition     n
##   <fct>     <int>
## 1 111         543
## 2 121         581
```

```
write.csv(df_sgc3a,"study_files/spring18_sgc3a_blocks.csv", row.names = FALSE)

df_sgc3b <- df_blocks %>% filter (condition %in% f_sgc3b)
df_sgc3b %>% group_by(condition) %>%
  summarize(n=n())
```

```
## # A tibble: 4 x 2
##   condition     n
##   <fct>     <int>
## 1 111         543
## 2 121         581
## 3 211         240
## 4 311         464
```

```
write.csv(df_sgc3b,"study_files/spring18_sgc3b_blocks.csv", row.names = FALSE)

df_sgc4 <- df_blocks %>% filter (condition %in% f_sgc4)
df_sgc4 %>% group_by(condition) %>%
  summarize(n=n())
```

```
## # A tibble: 3 x 2
##   condition     n
```

```
##   <fct>     <int>
## 1 111         543
## 2 112          90
## 3 113         803
```

```
write.csv(df_sgc4,"study_files/spring18_sgc4a_blocks.csv", row.names = FALSE)
```