

02 CSI online typing: Preprocessing

Kirsten Stark

01 November, 2021

Load packages

```
rm(list = ls())

library(tidyr)
library(dplyr)

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##   filter, lag

## The following objects are masked from 'package:base':
##
##   intersect, setdiff, setequal, union

options( "encoding" = "UTF-8" )
```

Load and preprocess data

This input file needs to be entered by hand:

```
# input output main data
type <- c("main", "replacement")
input <- c("data_csi_online_2021_2021-03-05_17-33.csv", "data_csi_online_20212_2021-03-15_14-23.csv")

options( "encoding" = "UTF-8" )

# input output: pretest
# input <- "pretest_complete.csv"
#output <- "pretest_long.csv"

# output
output <- "data_long.csv"
```

```

# arrays
arrays <- "arrays_umlaut.csv"

# prolific IDs to approve
approve <- "prolificid_approve"

# load data
datafiles <- list()
for(i in 1:length(input)) {
  #dataname = paste("data",type[i],sep = "_")
  #assign(dataname,
  #      read.csv(here::here("data", input[i]), sep = ";", na = ""))
  datafiles[[i]] <- read.csv(here::here("data", "raw", input[i]), sep = ";",
                             na = "")
}

# perform some transformations on each dataframe
for(i in 1:length(input)) {
  # add type column
  datafiles[[i]]$type <- type[i]
  # delete description column and experimenter's tryout data
  datafiles[[i]] <- datafiles[[i]][-c(1:2),]
  # add subject id
  datafiles[[i]] <- datafiles[[i]] %>% dplyr::mutate(subject = row_number())
}

```

Convert to long format, prepare wide dataframe, and bind long and wide dataframe together

First convert all variables that have values for each trial, then bind them together. In a next step bind them to the variables that only have one value per participant.

```

for(i in 1:length(input)){

#-----
# Prepare long data frame

  ### MAIN TASK - LETTERS
  # letters of first 1-80 trials of the main experiment
  df1 <- datafiles[[i]] %>% select('subject', starts_with(c("XT")) %>%
    pivot_longer(
      cols = -subject,
      names_to = c("trial", ".value"),
      names_pattern = "([~_]+)_(.*)",
      values_to = "timing") %>%
    group_by(subject) %>%
    dplyr::mutate(trial = seq(1,80, by = 1)) %>%
    setNames(paste0('letters.', names(.)))
  # letters of last 81-160 trials of the main experiment
  df2 <- datafiles[[i]] %>% select('subject', starts_with(c("XU"))) %>%
    pivot_longer(

```

```

      cols = -subject,
      names_to = c("trial", ".value"),
      names_pattern = "([_]+)(.*)",
      values_to = "timing") %>%
      group_by(subject) %>%
      dplyr::mutate(trial = seq(81,160, by = 1)) %>%
      setNames(paste0('letters.', names(.)))
# bind first 80 and last 80 trials together
df_letters <- bind_rows(df1, df2) %>%
  dplyr::rename(subject = letters.subject,
    trial = letters.trial) %>%
  arrange(subject, trial)
# delete letter columns from wide data frame
datafiles[[i]] <- datafiles[[i]] %>% select(!starts_with(c("XT", "XU")))

#### MAIN TASK - LETTER TIMING
# letter timing of first 1-80 trials of the main experiment
df1 <- datafiles[[i]] %>%
  select('subject', starts_with("TI") &
    !starts_with("TIME")) %>%
  pivot_longer(
    cols = -subject,
    names_to = c("trial", ".value"),
    names_pattern = "([_]+)(.*)",
    values_to = "timing") %>%
    group_by(subject) %>%
    dplyr::mutate(trial = seq(1,80, by = 1)) %>%
    setNames(paste0('timing.', names(.)))
# letter timing of last 81-160 trials of the main experiment
df2 <- datafiles[[i]] %>% select('subject', starts_with(c("TJ"))) %>%
  pivot_longer(
    cols = -subject,
    names_to = c("trial", ".value"),
    names_pattern = "([_]+)(.*)",
    values_to = "timing") %>%
    group_by(subject) %>%
    dplyr::mutate(trial = seq(81,160, by = 1)) %>%
    setNames(paste0('timing.', names(.)))
# bind first 80 and last 80 trials together
df_timing <- bind_rows(df1, df2) %>%
  dplyr::rename(subject = timing.subject,
    trial = timing.trial) %>%
  arrange(subject, trial)
# delete timing columns from wide data frame
datafiles[[i]] <- datafiles[[i]] %>%
  select(!(starts_with(c("TI", "TJ")) & !starts_with("TIME")))

### MAIN TASK - ENTIRE WORDS
# typed words of first 1-80 trials of the main experiment
df1 <- datafiles[[i]] %>%
  select('subject', starts_with("AW")) %>%
  pivot_longer(
    cols = -subject,

```

```

        names_to = c("trial"),
        values_to = "word") %>%
        group_by(subject) %>%
        dplyr::mutate(trial = seq(1,80, by = 1))
# typed words of last 1-80 trials of the main experiment
df2 <- datafiles[[i]] %>%
        select('subject', starts_with("AV")) %>%
        pivot_longer(
            cols = -subject,
            names_to = c("trial"),
            values_to = "word") %>%
        group_by(subject) %>%
        dplyr::mutate(trial = seq(81,160, by = 1))
# bind first 80 and last 80 trials together
df_words <- bind_rows(df1, df2) %>%
        arrange(subject, trial)
# delete word columns from wide data frame
datafiles[[i]] <- datafiles[[i]] %>%
        select(!starts_with(c("AW", "AV")))

### FAMILIARIZATION WORDS
### (only to control that ppt payed attention to the task)
# typed words of first 1-80 trials of the familiarization
df1 <- datafiles[[i]] %>%
        select('subject', starts_with("FA03")) %>%
        pivot_longer(
            cols = -subject,
            names_to = c("trial"),
            values_to = "fam_typed") %>%
        group_by(subject) %>%
        dplyr::mutate(trial = seq(1,80, by = 1))
# typed words of last 1-80 trials of the main experiment
df2 <- datafiles[[i]] %>%
        select('subject', starts_with("FA04")) %>%
        pivot_longer(
            cols = -subject,
            names_to = c("trial"),
            values_to = "fam_typed") %>%
        group_by(subject) %>%
        dplyr::mutate(trial = seq(81,160, by = 1))
# bind first 80 and last 80 trials together
df_fam <- bind_rows(df1, df2) %>%
        arrange(subject, trial)
# delete word columns from wide data frame
datafiles[[i]] <- datafiles[[i]] %>%
        select(!starts_with(c("FA03", "FA04")))

### Bind the four trial-wise dataframes together
df_main <- merge(df_fam, df_words, by = c("subject", "trial"))
df_main <- merge(df_main, df_letters, by= c("subject", "trial") )
df_main <- merge(df_main, df_timing, by= c("subject", "trial") ) %>%
        arrange(subject, trial)

```

```

#-----
# Adapt wide data frame with info that is assessed only once

# for control reasons: calculate time sum by hand:
# sum dwell times for each page
datafiles[[i]] <- datafiles[[i]] %>%
  mutate_at(vars(contains("TIME0")), as.numeric)
datafiles[[i]] <- datafiles[[i]] %>% rowwise() %>%
  dplyr::mutate(timetotal = rowSums(across(starts_with("TIME0")))/60)

# delete columns with info we don't need
datafiles[[i]] <- datafiles[[i]] %>%
  select(-c(CASE, QUESTNNR, MODE, STARTED, SD22_PRV,
            SD22_BID, SD22_BVS,
            SD28_01, SD29_01,
            FA02_01, PT01, PT02, PT03, PT04,
            LASTDATA,
            MISSING, MISSREL
            ))

# delete columns that contain only NAs
datafiles[[i]] <- datafiles[[i]] %>% select_if(~sum(!is.na(.)) > 0)

# give columns more recognizable names
if (type[i] == "main") {
  datafiles[[i]] <- datafiles[[i]] %>%
    dplyr::rename(array_no = AY01_01, gender = SD01, age = SD02_01,
                  language.test = SD20, language = SD21,
                  os_system = SD22_OS, browser_automatic = SD22_BNM,
                  system_format = SD22_FmF, prolificid = SD24_01,
                  fingers_l = SD25, fingers_r = SD26,
                  handedness = SD27, operator_system = SD30,
                  system = SD32, browser = SD31,
                  browser_other = SD31_07,
                  keyboard_type = KB01,
                  comments = IM01_01, time_wo_outlier = TIME_SUM,
                  screen_width = SD22_ScW, screen_height = SD22_ScH,
                  questionnaire_width = SD22_QnW)
} else if (type[i] == "replacement") {
  datafiles[[i]] <- datafiles[[i]] %>%
    dplyr::rename(array_no = AY01_01, gender = SD01, age = SD02_01,
                  language.test = SD20, language = SD21,
                  os_system = SD22_OS, browser_automatic = SD22_BNM,
                  system_format = SD22_FmF, prolificid = SD24_01,
                  fingers_l = SD25, fingers_r = SD26,
                  handedness = SD27, operator_system = SD30,
                  system = SD32, browser = SD31,
                  operator_system_other = SD30_07,
                  keyboard_type = KB01,
                  comments = IM01_01, time_wo_outlier = TIME_SUM,
                  screen_width = SD22_ScW, screen_height = SD22_ScH,
                  questionnaire_width = SD22_QnW)
}

```

```

}

#-----
# Bind long and wide data frame together
# Repeat each subjects' rows 160 times (no of trials)
datafiles[[i]] <- datafiles[[i]] %>% slice(rep(seq_len(n()), 160))

# Add trial number to wide data frame
datafiles[[i]]$trial <- rep(1:160, times = max(datafiles[[i]]$subject))

# bind wide and long info together
datafiles[[i]] <- datafiles[[i]] %>%
  left_join(df_main, by = c("subject", "trial")) %>%
  relocate(subject, trial)

#-----
# Convert numeric variables from string to integer:
#str(df)
if(type[i] == "main") {
  datafiles[[i]] <- datafiles[[i]] %>%
    mutate_at(vars(!c("prolificid", "browser_other",
                      "word", "comments", "type",
                      "array_no", "TIME_RSI", contains("KBO"),
                      contains("TT0"), contains("fam_typed"),
                      contains("letters"))), as.numeric)
} else if(type[i] == "replacement") {
  datafiles[[i]] <- datafiles[[i]] %>%
    mutate_at(vars(!c("prolificid", "operator_system_other",
                      "word", "comments", "type",
                      "array_no", contains("KBO"),
                      contains("TT0"), contains("fam_typed"),
                      contains("letters"))), as.numeric)
}
}

```

Roughly check participant for Prolific approval

Did all participants finish the experiment?

```

# did all participants finish the experiment?
for(i in 1:length(input)) {
  print(paste(type[i], ":"))
  print("Experiment completed?")
  print(table(datafiles[[i]]$FINISHED)/160)
  print("What was the last experimental page reached?")
  print(table(datafiles[[i]]$LASTPAGE)/160)
}

```

```

## [1] "main :"
## [1] "Experiment completed?"
##
## 0 1

```

```
## 2 30
## [1] "What was the last experimental page reached?"
##
## 4 12 33
## 1 1 30
## [1] "replacement :"
## [1] "Experiment completed?"
##
## 0 1
## 2 3
## [1] "What was the last experimental page reached?"
##
## 3 20 33
## 1 1 3
```

Two participants didn't finish the experiment and dropped out on page 4 (keyboard test) and 12 (instruction turn on caps lock) of the experiment, respectively.

Replacement data: Here too, two participants entered the experiment, but didn't complete it. One dropped out on page 3 (welcome page) and one on page 20 (main experiment).

Exclude these participants and export prolific IDs for approval:

```
for (i in 1:length(input)) {
  exclude <- datafiles[[i]]$prolificid[datafiles[[i]]$FINISHED == "0" &
    datafiles[[i]]$LASTPAGE != "30"]
  exclude_id <- datafiles[[i]]$subject[datafiles[[i]]$FINISHED == "0" &
    datafiles[[i]]$LASTPAGE != "30"]

  # Exclude from data frame
  datafiles[[i]] <- datafiles[[i]][!datafiles[[i]]$subject %in% exclude_id,]

  # update subject IDs
  datafiles[[i]]$subject <-
    rep(1:nlevels(as.factor(datafiles[[i]]$subject)), each = 160)

  # All the other data look fine, so we can include all others:
  include <- as.data.frame(
    datafiles[[i]]$prolificid[datafiles[[i]]$FINISHED == "1" &
      datafiles[[i]]$LASTPAGE == "33"])

  # export prolific IDs
  filename <- paste(approve, "_", type[i], ".csv", sep = "")
  write.csv(include, here::here("data", "transient_data_files", filename), row.names = FALSE)

  # delete prolific ids to anonymize data frame
  datafiles[[i]] <- datafiles[[i]] %>%
    dplyr::select(-prolificid)
}
```

Add array (actual stimuli) for each participant

```

# load arrays
arrays <- read.csv(here::here("data", "supplementary_info", arrays),
  sep = ",", na = "NA")

for (i in 1:length(input)) {
  # convert array column in datafiles
  datafiles[[i]]$array_no <- as.numeric(stringr::str_remove(
    datafiles[[i]]$array_no, "Array"))
  # bind arrays to data frame
  datafiles[[i]]$item <- NA
  for(j in 1:nlevels(as.factor(datafiles[[i]]$subject))){
    array <- arrays[, unique(datafiles[[i]]$array_no[
      datafiles[[i]]$subject == j])]
    datafiles[[i]]$item[datafiles[[i]]$subject == j] <- array
  }

  # add category columns
  datafiles[[i]] <- datafiles[[i]] %>% group_by(subject, item) %>%
    dplyr::mutate(category = arrays$categorie[arrays$item == item]) %>%
    dplyr::mutate(supercategory =
      arrays$supercategorie[arrays$item == item] )

  # add position number
  datafiles[[i]] <- datafiles[[i]] %>% group_by(subject, category) %>%
    add_count() %>%
    dplyr::mutate(PosOr = seq(1:n)) %>% select(-n)
  # add ordinal position for fillers
  table(datafiles[[i]]$PosOr)
  count <- 1
  for (j in 1:nrow(datafiles[[i]])) {
    if(datafiles[[i]]$category [j] == "Filler") {
      datafiles[[i]]$PosOr[j] <- count
      count <- count+1
    }
    if(count == 6) { count <- 1}
  }
  table(datafiles[[i]]$PosOr)
}

```

Bind the dataframes together

```
df <- bind_rows(datafiles)
```

Export prepared data frame

```
write.csv(df, here::here("data", "transient_data_files", output), row.names = FALSE)
```