04 CSI online aphasia: Typing - Descriptives

Kirsten Stark

12 August, 2021

Load packages

```
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
```

```
library(tidyr)

rm(list = ls())
```

Load and preprocess data

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

# input
input <- "data_long_final.csv"

# load data
df <- read.csv(here::here("data", "transient_data_files", input), sep = ",",  na = "")
```

Duration of the experiment

```
print("Outlier-corrected duration (provided by soscisurvey)")
```

```
## [1] "Outlier-corrected duration (provided by soscisurvey)"
```

```r
mean(df$time_wo_outlier)/60 # 21, 37 min = 21 min 22sec
```

```
## [1] 0
```

```r
sd(df$time_wo_outlier)/60
```

```
## [1] 0
```

```r
range(df$time_wo_outlier)/60 # 0.00000 30.21667
```

```
## [1] 0 0
```

```r
# uncomment for more than 10 VPs
# duration <- df %>% dplyr::select(starts_with("TIME")) %>%
#   dplyr::select(!"time_wo_outlier") %>% dplyr::select(!"TIME_RSI")
# print("Outlier-corrected duration (provided by soscisurvey)")
# duration$sum = duration %>% rowSums(na.rm = TRUE)
# mean(duration$sum)/60
# sd(duration$sum)/60
# range(duration$sum)/60
```

Description of participants

Gender:

```r
df <- df %>% mutate(gender_char = case_when(gender == 1 ~ "female",
                                            gender == 2 ~ "male"))
table(df$gender_char)/160 # 1 = female, 2 = male, 3 = diverse
```

```
##
## female
##      1
```

```r
print("percentage female:")
```

```
## [1] "percentage female:"
```

```r
sum(df$gender == 1)/nrow(df)
```

```
## [1] 1
```

Age:

```r
print("mean:"); mean(df$age)
```

```
## [1] "mean:"
```

```
## [1] 28
```

```r
print("sd:"); sd(df$age)
```

```
## [1] "sd:"
```

```
## [1] 0
```

```r
print("range:"); range(df$age)
```

```
## [1] "range:"
```

```
## [1] 28 28
```

Handedness:

```r
# 1 = left handed, 2 = right handed, 3 = ambidexter/both
df <- df %>% mutate(handedness_char = case_when(handedness == 1 ~ "left-handed",
                                                handedness == 2 ~ "right-handed"))
table(df$handedness_char)/160
```

```
##
## right-handed
##            1
```

```r
print("percentage right-handed:")
```

```
## [1] "percentage right-handed:"
```

```r
sum(df$handedness == 2)/nrow(df)
```

```
## [1] 1
```

Fingers used for typing:

```r
# 1 = 1, 2 = 2, 3 = 3, 4 = 4, 5 = 5, 6 = don't know
print("left hand: "); table(df$fingers_l)/160
```

```
## [1] "left hand: "
```

```
##
## 3
## 1
```

```r
print("right hand: "); table(df$fingers_r)/160
```

```
## [1] "right hand: "
```

```
##
## 3
## 1
```

```
df$fingers_l <- ifelse(df$fingers_l == 6, NA, df$fingers_l)
df$fingers_r <- ifelse(df$fingers_r == 6, NA, df$fingers_r)

print("Average number of fingers used (both hands combined):")
```

```
## [1] "Average number of fingers used (both hands combined):"
```

```
mean(df$fingers_l+df$fingers_r, na.rm = T); sd(df$fingers_l+df$fingers_r, na.rm = T)
```

```
## [1] 6
```

```
## [1] 0
```

Mother tongue (experiment was restricted to native German speakers): This seems to have worked

```
table(df$language) # 1 = yes (mother tongue is German), 2 = no
```

```
##
##   1
## 160
```

```
table(df$language.test) # 1 = der, 2 = die, 3 = das (das is correct)
```

```
##
##   1
## 160
```

Average typing speed and accuracy

Subset all typing test columns and select one row per particpant only

```
source("automatic_preprocessing_functions.R")
```

```
##
## Attaching package: 'stringdist'
```

```
## The following object is masked from 'package:tidyr':
##
##     extract
```

```
dat <- df %>% filter(trial == 1) %>%
    dplyr::select(c("subject", starts_with("TT")))
```

```
test1 <- "Der Fuchs ist nah mit Hund und Wolf verwandt. Die Tiere sehen Hunden auch recht ähnlich. Füchs
print("Actual length of characters in test1 (including spaces):")
```

```
## [1] "Actual length of characters in test1 (including spaces):"
```

```
nchar(test1)
```

```
## [1] 155
```

```
test2 <- "Der Schwanz eines Fuchses ist fast halb so lang wie das ganze Tier. Das Fell ist meist rot od
print("Actual length of characters in test2 (including spaces):")
```

```
## [1] "Actual length of characters in test2 (including spaces):"
```

```
nchar(test2)
```

```
## [1] 157
```

```
test3 <- "Typisch für den Fuchs sind zudem seine aufgestellten Ohren, die ihm auf der Jagd behilflich s
print("Actual length of characters in test3 (including spaces):")
```

```
## [1] "Actual length of characters in test3 (including spaces):"
```

```
nchar(test3)
```

```
## [1] 154
```

Accuracy

Accurary similar as in e.g., Crump (2003), Crump & Logan (2010), Pinet, Dubarry, & Alario (2016): Percentage of 5-character words containing no error (neither a backspace nor a typographical error). The data from the three text subsections will be collapsed.

5-character-word based accuracy:

```
## Descriptive statistics:
print("Mean Accuracy (5-char words correct)"); round(mean(accuracy_fiveletterwords),4)*100
```

```
## [1] "Mean Accuracy (5-char words correct)"
```

```
## [1] 83.33
```

```
print("SD Accuracy (5-char words correct)"); round(sd(accuracy_fiveletterwords),4)*100
```

```
## [1] "SD Accuracy (5-char words correct)"
```

```
## [1] NA
```

```
print("Range Accuracy (5-char words correct)"); round(range(accuracy_fiveletterwords),4)*100
```

```
## [1] "Range Accuracy (5-char words correct)"
```

```
## [1] 83.33 83.33
```

Alternative calculation of accuracy: Character-based accuracy on the entire text - Participants typed texts are compared to the entire text and the number of insertions, deletions, substitutions and transpositions are devided by the total amount of characters in the text.

Text-based accuracy:

```
accuracy_raw <- (accuracy$test1_raw + accuracy$test2_raw + accuracy$test3_raw)/3
print("Mean Accuracy (entire text)");round(mean(accuracy_raw),4)*100
```

```
## [1] "Mean Accuracy (entire text)"
```

```
## [1] 69.26
```

```
print("Sd Accuracy (entire text)");round(sd(accuracy_raw),4)*100
```

```
## [1] "Sd Accuracy (entire text)"
```

```
## [1] NA
```

```
print("Range Accuracy (entire text)");round(range(accuracy_raw),4)*100
```

```
## [1] "Range Accuracy (entire text)"
```

```
## [1] 69.26 69.26
```

```
cor(accuracy_raw, accuracy_fiveletterwords)
```

```
## [1] NA
```

```
#cor.test(accuracy_raw, accuracy_fiveletterwords)
```

Speed

Words per minute: Defined as the number of words correctly typed divided by the total time of all words (similar to Pinet et al., 2016; Crump & Logan 2010)

```
wpm <- NA
for (i in 1:nsubject) {
  eval(parse(text=paste0(
  "x<- fivecharwords %>% filter(!is.na(accuracy_subject",i,"))")))
  # calculate no of words correctly typed
  eval(parse(text=paste0(
  "number_of_words <- sum(x$accuracy_subject", i,"!=", 0, ", na.rm=T)")))
  # calculate total time need to type all fiveletter words
  eval(parse(text=paste0(
  "total_time_fiveletter <- sum(x$time_subject", i,", na.rm=T)")))
  # convert time to seconds
  total_time_fiveletter <- total_time_fiveletter/1000/60
  #calculate word per minutes
  wpm[i] <-number_of_words/total_time_fiveletter
}
```

Speed in 5-character words per minute:

```
## Descriptive statistics:
print("Mean Speed (5-char words correct)"); round(mean(wpm),2)
```

```
## [1] "Mean Speed (5-char words correct)"
```

```
## [1] 12.4
```

```
print("SD Speed (5-char words correct)"); round(sd(wpm),2)
```

```
## [1] "SD Speed (5-char words correct)"
```

```
## [1] NA
```

```
print("Range Speed (5-char words correct)"); round(range(wpm),2)
```

```
## [1] "Range Speed (5-char words correct)"
```

```
## [1] 12.4 12.4
```

Alternative speed measurement: Characters per minute, based on the length of the text, not the actual number of characters typed.

```
speed1 <- nchar(test1)/(dat$TT02_02/100/60)
speed2 <- nchar(test2)/(dat$TT05_02/100/60)
speed3 <- nchar(test3)/(dat$TT08_02/100/60)
```

```
cor(speed1, speed2)
```

```
## [1] NA
```

```
cor(speed1, speed3)
```

```
## [1] NA
```

```
cor(speed2, speed3)
```

```
## [1] NA
```

```
#cor.test(speed1, speed2)
#cor.test(speed1, speed3)
#cor.test(speed2, speed3)
speedperppt <- (speed1+speed2+speed3)/3
cor(speedperppt, wpm)
```

```
## [1] NA
```

```r
#cor.test(speedperppt, wpm)
print("mean characters per minute:"); round(mean(speedperppt),2);
```

```
## [1] "mean characters per minute:"
```

```
## [1] 29.29
```

```r
print("SD characters per minute:"); round(sd(speedperppt),2);
```

```
## [1] "SD characters per minute:"
```

```
## [1] NA
```

```r
print("range characters per minute:"); round(range(speedperppt),2);
```

```
## [1] "range characters per minute:"
```

```
## [1] 29.29 29.29
```

System info (indicated by participants)

Browser:

```r
df <- df %>% mutate(browser_char = case_when(browser == 1 ~ "Chrome",
                                             browser == 2 ~ "Coast",
                                             browser == 3 ~ "Firefox",
                                             browser == 4 ~ "Internet Explorer",
                                             browser == 5 ~ "Opera",
                                             browser == 6 ~ "Safari",
                                             browser == 8 ~ "Microsoft Edge"))#,
                                             #browser == 7 ~ browser_other))

table(df$browser_char)
```

```
##
## Safari
##    160
```

Operating system:

```r
df <- df %>% mutate(operator_system_char = case_when(operator_system == 1 ~ "MacOSX",
                                             operator_system == 2 ~ "Linux",
                                             operator_system == 3 ~ "Windows10",
                                             operator_system == 4 ~ "Windows8",
                                             operator_system == 5 ~ "Windows7",
                                             operator_system == 6 ~ "WindowsVista",
                                             operator_system == 8 ~ "WindowsNT",
                                             operator_system == -1 ~ "don't know",
                                             operator_system == -9 ~ "NA"))#,
                                             #operator_system == 7 ~ operator_system_other))
table(df$operator_system_char)
```

```
##
## MacOSX
##    160
```

System:

```
df <- df %>% mutate(system_char = case_when(system == 1 ~ "Computer(PC)",
                                            system == 2 ~ "Laptop",
                                            system == 3 ~ "TV",
                                            system == 4 ~ "Tablet",
                                            system == 5 ~ "Phone",
                                            system == -1 ~ "don't know",
                                            system == -9 ~ "NA"))#,
                                            #system == 7 ~ "other"))
table(df$system_char)
```

```
##
## Laptop
##    160
```

Attention checks

1) Item vs. non-item

```
## Item vs. non-item
# CH01_01 (Taube), CH01_02 (Apfel), CH02_01 (Luftballon) and CH02_02 (Biene) are items and 2 should be
# CH02_03 (Radio), CH02_04 (Sparschwein), CH02_03 (Laptop) and CH02_04 (Wattestäbchen) are non-items an
## Did participants cheat
# CH03 = 1 - yes, I worked through it till the end,
# CH03 = 2 - no, I stopped or cheated midway
# CH03 = -9 - no answer

attcheck <- data.frame(subject = unique(df$subject))

df <- df %>% mutate(itemvsnonitem1 =
                      case_when(CH01_01==2 & CH01_02==2 & CH01_03==1 & CH01_04==1 ~2,
                                CH01_01==2 || CH01_02==2 ~1,
                                CH01_01!=2 & CH01_02!=2 ~0)) %>%
  mutate(itemvsnonitem2 =
                      case_when(CH02_01==2 & CH02_02==2 & CH02_03==1 & CH02_04==1 ~2,
                                CH02_01==2 || CH02_02==2 ~1,
                                CH02_01!=2 & CH02_02!=2 ~0))
table(df$itemvsnonitem1)/160
```

```
##
## 2
## 1
```

```
table(df$itemvsnonitem2)/160
```

```
##
## 2
## 1
```

```
# attcheck <- data.frame(subject = unique(df$subject))
#
#
# data <- data %>% mutate(attcheck =
#                   ifelse(CH01_01 == 2 & CH01_02 == 2 & CH02_01 == 2 & CH02_02 == 2 &
#                   CH01_03 == 1 & CH01_04 == 1 & CH02_03 == 1 & CH02_04 == 1, 1, 0)) %>%
#                    mutate(cheat = ifelse(CH03 == 1,1,ifelse(CH03 == 2,2,0)))
# data.frame(data$subject, )
# table(data$attcheck)
# table(data$cheat)
#
# # get prolific IDs of participants who failed the attention check
# #pretest %>% subset(attcheck == 0 & cheat == 2 ) %>%
# #  pull(SD24_01) # SD24_01 is prolific ID
#
# # subset to participants who passed only
# valid <- pretest  %>% filter(attcheck == 1 & cheat != 2)
#
# inspect <- data.frame(df$subject, df$word, df$fam_typed)
```

2) Cheating

```
df <- df %>% mutate(CH03 = case_when(CH03 == 1 ~
                                       " Ja, ich habe alles bis zum Ende bearbeitet.",
                                           CH03 == 2 ~
                                       "Nein, ich habe zwischendurch aufgehoert oder geschummelt."))
table(df$CH03)/160
```

```
##
##  Ja, ich habe alles bis zum Ende bearbeitet.
##                                            1
```

3) Keyboard Check

```
# self-indicated keyboard type
df <- df %>% mutate(keyboard_type = case_when(keyboard_type == 1 ~ "QWERTY",
                                              keyboard_type == 2 ~ "QWERTZ",
                                              keyboard_type == 3 ~ "QÜERTY",
                                              keyboard_type == 4 ~ "ÄWERTY",
                                              keyboard_type == 5 ~ "AZERTY",
                                              keyboard_type == 6 ~ "QZERTY",
                                              keyboard_type == 7 ~ "other"))

table(df$keyboard_type)/160
```

```
##
## QWERTZ
##      1
```

```
# keyboard test
table(df$KB02_01)/160
```

```
##
## ägyptisch
##         1
```

```
table(df$KB03_01)/160 # code ä is Quote
```

```
##
## Quote
##     1
```

```
table(df$KB03_02)/160 # code y is KeyZ
```

```
##
## KeyZ
##    1
```

```
table(df$KB03_03)/160 # code p is KeyP
```

```
##
## KeyP
##    1
```

(The keyboard screening worked well for all participants)

Comments

Comments don't indicate any problems that should lead to participant exclusion:

```
#table(df$comments)/160
```

Fully anonymize data and reduce data frame

```
df_a <- df %>% dplyr::select(!"gender" & !"age" & !starts_with("language") &
                               !starts_with("TIME") & !starts_with("handedness") &
                               !starts_with("fingers") & !starts_with("browser") &
                               !starts_with("operator_system") & !starts_with("system") &
                               !"keyboard_type" & !starts_with("KB") &
                               !starts_with("CHO") & !starts_with("X.") & !"X" &
                               !starts_with("screen") & !starts_with("os") &
                               !starts_with("questionnaire") & !"ORO1_01" &
                               !starts_with("TTO") & #!"comments" &
                               !"FINISHED" & !"Q_VIEWER" &
                               !"LASTPAGE" & !"MAXPAGE" & !"DEG_TIME"& !"type" & !"fam_typed" &
                               !"gender_char" & !"itemvsnonitem1" & !"itemvsnonitem2")
```

Reduce data frame to columns needed for data analyses or useful to understand the data (this df will be shared online). Interkeystroke intervals could be shared as well, but might only lead to confusion because the data frame will still be very wide.

```r
df_a <- df_a %>% dplyr::select(c("subject", "trial", "item",
                                 "category", "supercategory", "PosOr",
                           "answer_auto_jaro", "correct_auto_jaro",
                           "answer_auto_lv", "correct_auto_lv",
                           "word", "word.c", "letters.01", "timing.01"))
```

```r
write.csv(df_a, here::here("data","data_long_anonymous.csv"))
```