Data Processing

Milla Pihlajamaki and Caitlin Dawson

05 elokuu, 2022

Notes

- takes data downloaded from Gorilla **blinded, semicolon separated, csv format, short form, versions 22-25**
- cleans and processes data for the study 'Scientific thinking and decision-making in everyday life'

Version history: * version 22: Links fixed in information and consent * version 23: HJ updated the experiment (source deleted from CI task–only 5 sources because of trouble getting a stable URL; Viiskunta source was taken down) * version 24: HJ updated CI task to include all 6 sources again, one of them as image * version 25: HJ. Word adult taken off from Study Information and Content

Load packages

```
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
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(ggplot2)
library(lavaan)
```

```
## This is lavaan 0.6-9
## lavaan is FREE software! Please report any bugs.
```

```
library(naniar)
```

```
## Warning: package 'naniar' was built under R version 4.1.3
```

```
library(psych)
```

```
##
## Attaching package: 'psych'
```

```
## The following object is masked from 'package:lavaan':
##
##     cor2cov
```

```
## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha
```

```
library(psycho)
library(tidyr)
```

Load Data

This chunk loads the data and omits experiment-general columns from all files (except for participant private ID).

```
# Versions 22-25
versions <- paste0("v", 22:25)
questionnaire_types <- c("2wlk", "79hv", "7qsg", "81k4", "c7cw", "cl1u", "eygv", "go4a", "mz16", "o43u"
task_types <- c("7mar", "8mu5", "9nll", "n8ns", "tg12", "zryk")
questionnaire_names <- c("epiQ", "heurQ", "openQ", "validQ", "demoQ", "sciattQ", "needcogQ", "scicurQ",
task_names <- c("uutT", "NavonT", "NeckerT", "nogoT", "matreasT", "citiT")
names_all <- c(questionnaire_names, task_names)

for (j in 1:length(versions)) {
  for (i in 1:length(questionnaire_types)) {
  assign(paste0("data_", questionnaire_names[i], "_", versions[j]),
         read.csv(paste0("data_exp_55551-", versions[j], "_questionnaire-", questionnaire_types[i], ".cs
    }
}

for (j in 1:length(versions)) {
  for (i in 1:length(task_types)) {
    assign(paste0("data_", task_names[i], "_", versions[j]),
           read.csv(paste0("data_exp_55551-", versions[j], "_task-", task_types[i], ".csv"), sep = ";")
  }
}
```

Create Questionnaire Dataframe

This series of chunks computes the necessary data cleaning to merge the questionnaire dataframes and merges them.

Merge the Different Versions of Each Questionnaire

This chunk combines the different versions of each questionnaire (leaving us with one df per questionnaire).

```
# Concatenate the versions of each questionnaire
data_big5Q <- rbind(data_big5Q_v22, data_big5Q_v23,
                    data_big5Q_v24, data_big5Q_v25)
data_citiT <- rbind(data_citiT_v22, data_citiT_v23,
                    data_citiT_v24, data_citiT_v25)
data_demoQ <- rbind(data_demoQ_v22, data_demoQ_v23,
                    data_demoQ_v24, data_demoQ_v25)
data_epiQ <- rbind(data_epiQ_v22, data_epiQ_v23,
                   data_epiQ_v24, data_epiQ_v25)
data_nogoT <- rbind(data_nogoT_v22, data_nogoT_v23,
                    data_nogoT_v24, data_nogoT_v25)
data_heurQ <- rbind(data_heurQ_v22, data_heurQ_v23,
                    data_heurQ_v24, data_heurQ_v25)
data_inthumQ <- rbind(data_inthumQ_v22, data_inthumQ_v23,
                      data_inthumQ_v24, data_inthumQ_v25)
data_matreasT <- rbind(data_matreasT_v22, data_matreasT_v23,
                       data_matreasT_v24, data_matreasT_v25)
data_NavonT <- rbind(data_NavonT_v22, data_NavonT_v23,
                     data_NavonT_v24, data_NavonT_v25)
data_NeckerT <- rbind(data_NeckerT_v22, data_NeckerT_v23,
                      data_NeckerT_v24, data_NeckerT_v25)
data_needcloQ <- rbind(data_needcloQ_v22, data_needcloQ_v23,
                       data_needcloQ_v24, data_needcloQ_v25)
data_needcogQ <- rbind(data_needcogQ_v22, data_needcogQ_v23,
                       data_needcogQ_v24, data_needcogQ_v25)
data_openQ <- rbind(data_openQ_v22, data_openQ_v23,
                    data_openQ_v24, data_openQ_v25)
data_rpQ <- rbind(data_rpQ_v22, data_rpQ_v23,
                  data_rpQ_v24, data_rpQ_v25)
data_sciattQ <- rbind(data_sciattQ_v22, data_sciattQ_v23,
                      data_sciattQ_v24, data_sciattQ_v25)
data_scicurQ <- rbind(data_scicurQ_v22, data_scicurQ_v23,
                      data_scicurQ_v24, data_scicurQ_v25)
data_infoQ <- rbind(data_infoQ_v22, data_infoQ_v23,
                    data_infoQ_v24, data_infoQ_v25)
data_uutT <- rbind(data_uutT_v22, data_uutT_v23,
                   data_uutT_v24, data_uutT_v25)
data_validQ <- rbind(data_validQ_v22, data_validQ_v23,
                     data_validQ_v24, data_validQ_v25)

# Remove the empty last line of each file
for (i in 1:length(names_all)) {
  assign(paste0("data_", names_all[i]),
         get(paste0("data_", names_all[i]))[which(!is.na(get(paste0("data_", names_all[i]))[, 1])), ])
}

# Remove version files to clean up the environment
for (j in 1:length(versions)) {
  for (i in 1:length(names_all)) {
    rm(list = paste0("data_", names_all[i], "_", versions[j]))
```

```
  }
}
```

Merge Questionnaire Dataframes

This chunk merges all questionnaire dataframes, leaving us with one dataframe for all questionnaire data.

```
# Rename two columns before combining all the questionnaires into one df (the column has the same name
    # END.QUESTIONNAIRE column gives you the response time for that questionnaire.
    # Randomise.questionnaire.elements. column gives you a logical value indicating whether the questio

# END.QUESTIONNAIRE
names(data_demoQ)[names(data_demoQ) == "END.QUESTIONNAIRE"] <- "END.QUESTIONNAIRE.DEMOGRAPHICS"
names(data_big5Q)[names(data_big5Q) == "END.QUESTIONNAIRE"] <- "END.QUESTIONNAIRE.BIG5"
names(data_epiQ)[names(data_epiQ) == "END.QUESTIONNAIRE"] <- "END.QUESTIONNAIRE.EPISTEMIC"
names(data_heurQ)[names(data_heurQ) == "END.QUESTIONNAIRE"] <- "END.QUESTIONNAIRE.HEURISTIC"
names(data_inthumQ)[names(data_inthumQ) == "END.QUESTIONNAIRE"] <- "END.QUESTIONNAIRE.INT.HUM"
names(data_needcloQ)[names(data_needcloQ) == "END.QUESTIONNAIRE"] <- "END.QUESTIONNAIRE.NEED.CLO"
names(data_needcogQ)[names(data_needcogQ) == "END.QUESTIONNAIRE"] <- "END.QUESTIONNAIRE.NEED.COG"
names(data_openQ)[names(data_openQ) == "END.QUESTIONNAIRE"] <- "END.QUESTIONNAIRE.OPEN.THINK"
names(data_rpQ)[names(data_rpQ) == "END.QUESTIONNAIRE"] <- "END.QUESTIONNAIRE.RANDOM.PROB"
names(data_sciattQ)[names(data_sciattQ) == "END.QUESTIONNAIRE"] <- "END.QUESTIONNAIRE.SCIENCE.ATT"
names(data_scicurQ)[names(data_scicurQ) == "END.QUESTIONNAIRE"] <- "END.QUESTIONNAIRE.SCIENCE.CUR"
names(data_infoQ)[names(data_infoQ) == "END.QUESTIONNAIRE"] <- "END.QUESTIONNAIRE.INFO"
names(data_validQ)[names(data_validQ) == "END.QUESTIONNAIRE"] <- "END.QUESTIONNAIRE.VALIDITY"

# Randomise.questionnaire.elements.
names(data_demoQ)[names(data_demoQ) == "Randomise.questionnaire.elements."] <- "Randomise.questionnaire
names(data_big5Q)[names(data_big5Q) == "Randomise.questionnaire.elements."] <- "Randomise.questionnaire
names(data_epiQ)[names(data_epiQ) == "Randomise.questionnaire.elements."] <- "Randomise.questionnaire.el
names(data_heurQ)[names(data_heurQ) == "Randomise.questionnaire.elements."] <- "Randomise.questionnaire
names(data_inthumQ)[names(data_inthumQ) == "Randomise.questionnaire.elements."] <- "Randomise.questionna
names(data_needcloQ)[names(data_needcloQ) == "Randomise.questionnaire.elements."] <- "Randomise.questio
names(data_needcogQ)[names(data_needcogQ) == "Randomise.questionnaire.elements."] <- "Randomise.questio
names(data_openQ)[names(data_openQ) == "Randomise.questionnaire.elements."] <- "Randomise.questionnaire
names(data_rpQ)[names(data_rpQ) == "Randomise.questionnaire.elements."] <- "Randomise.questionnaire.ele
names(data_sciattQ)[names(data_sciattQ) == "Randomise.questionnaire.elements."] <- "Randomise.questionna
names(data_scicurQ)[names(data_scicurQ) == "Randomise.questionnaire.elements."] <- "Randomise.questionna
names(data_infoQ)[names(data_infoQ) == "Randomise.questionnaire.elements."] <- "Randomise.questionnaire
names(data_validQ)[names(data_validQ) == "Randomise.questionnaire.elements."] <- "Randomise.questionnai

# Merge by participant ID
# All rows for all dataframes are kept
data1 <- merge(data_demoQ, data_big5Q,
               by = "Participant.Private.ID",
               all = TRUE)
data2 <- merge(data_epiQ, data_heurQ,
               by = "Participant.Private.ID",
               all = TRUE)
data3 <- merge(data_inthumQ, data_needcloQ,
               by = "Participant.Private.ID",
               all = TRUE)
```

```r
data4 <- merge(data_needcogQ, data_openQ,
               by = "Participant.Private.ID",
               all = TRUE)
data5 <- merge(data_rpQ, data_sciattQ,
               by = "Participant.Private.ID",
               all = TRUE)
data6 <- merge(data_scicurQ, data_infoQ,
               by = "Participant.Private.ID",
               all = TRUE)
data7 <- merge(data1, data2,
               by = "Participant.Private.ID",
               all = TRUE)
data8 <- merge(data3, data4,
               by = "Participant.Private.ID",
               all = TRUE)
data9 <- merge(data5, data6,
               by = "Participant.Private.ID",
               all = TRUE)
data10 <- merge(data7, data8,
                by = "Participant.Private.ID",
                all = TRUE)
data11 <- merge(data9, data_validQ,
                by = "Participant.Private.ID",
                all = TRUE)
data_Q_total <- merge(data10, data11,
                      by = "Participant.Private.ID",
                      all = TRUE)

# Clean the environment
rm(data1, data2, data3, data4, data5, data6, data7, data8, data9, data10, data11)
rm(data_big5Q, data_epiQ, data_heurQ, data_infoQ, data_inthumQ, data_needcloQ, data_needcogQ, data_open
```

Questionnaire Data

Exclusion Criteria

ALL QUESTIONNAIRES: This chunk goes through all questionnaire data, removing **all data for a participant** if they met one or more of the following exclusion criteria: * stated their data are not valid * stated their Finnish is not Äidinkieli, Keskusteleva, or Muu, Mikä.

Questionnaire Data Processing

After excluding data in accordance with the exclusion criteria above, the following steps are taken for each questionnaire: * reverse-code items (if necessary) * visualize and describe each item * check the structure of the questionnaire data with Cronbach's alpha (separately for each measured construct) * calculate mean or sum scores according to the questionnaire instructions * replace sum score 0s with NAs (no questionnaires included 0 as a lower bound in the response options, so sum/mean scores of 0 are not valid) * visualize and describe mean or sum scores

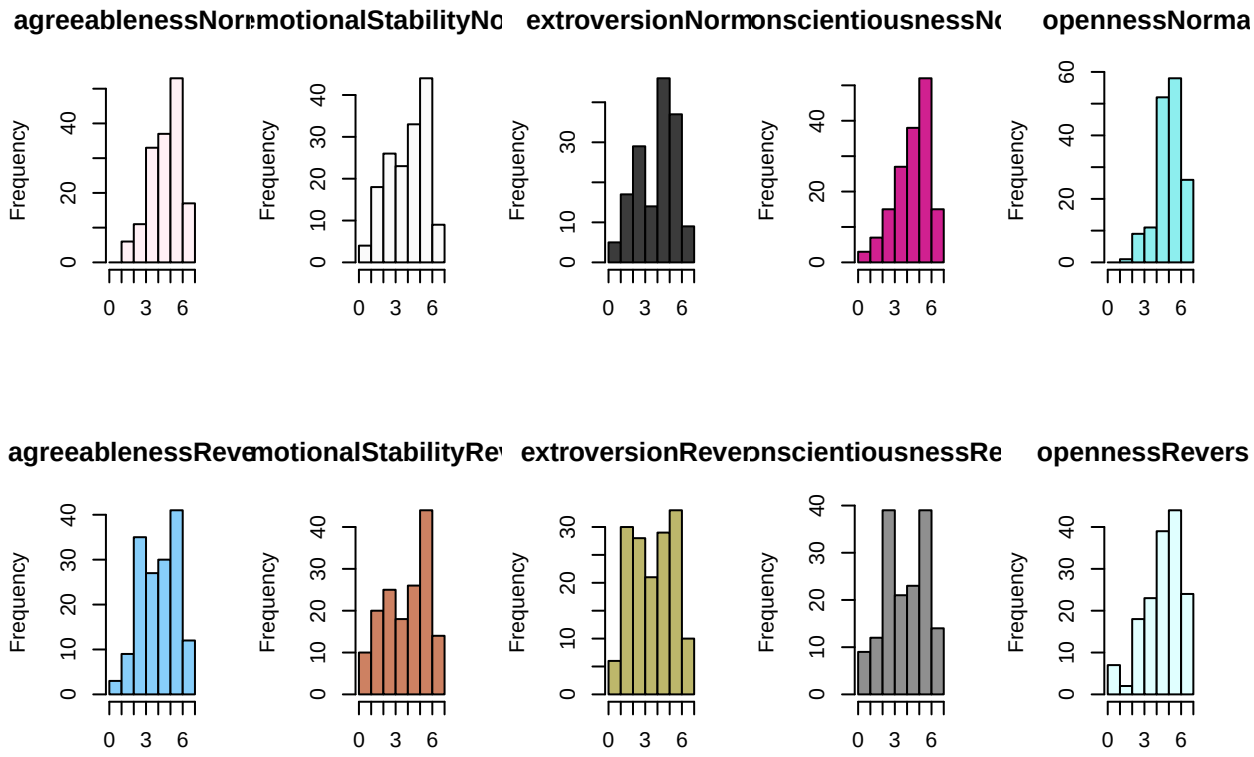
Validity and Language

```
# Save data for nonvalid ppts ("älä huomioi aineistoani") in a separate file prior to removal
data_nonvalid <- data_Q_total %>%
  filter(Validity == "Ã„lÃ¤ huomioi aineistoani. Jokin muu syy esti minua osallistumasta kunnolla." | Va

# Save data for participants with nonvalid language answers in a separate file. One participant who sai
data_Q_lang_omit <- data_Q_total %>%
  filter(Language != "Sujuva / Ã¤idinkieli" & Language != "Keskitaso / keskusteleva" & Language != "Muu

# Remove nonvalid participants
data_Q_total <- data_Q_total %>%
  filter(Validity %in% c("Aineistoani voi kÃ¤yttÃ¤Ã¤.", "Muu, mikÃ¤? ", NA, "") & Language %in% c("Suju
```

TIPI (Big Five)

Reliability are not calculated for TIPI, see Gosling's website for explanation. http://gosling.psy.utexas.edu/scales-weve-developed/ten-item-personality-measure-tipi/

```
# Big Five
b5 <- c("agreeablenessNormal", "emotionalStabilityNormal", "extroversionNormal", "conscientiousnessNorma
b5_rev <- c("agreeablenessReverse", "emotionalStabilityReverse", "extroversionReverse", "conscientiousne

# Reverse-code
for (i in 1:length(b5_rev)) {
  data_Q_total[, b5_rev[i]] <- reverse.code(keys = c(-1),
                                            items = data_Q_total[, b5_rev[i]],
                                            mini = c(1),
                                            maxi = c(7))
}

# Describe: item-level
par(mfrow = c(2, 5))
for (i in 1:length(b5)) {
  hist(as.numeric(data_Q_total[, b5[i]]),
       main = colnames(data_Q_total[b5[i]]),
       col = sample(colors(), 1),
       breaks = seq(0, 7, 1),
       xlab = ""
       )
}
```

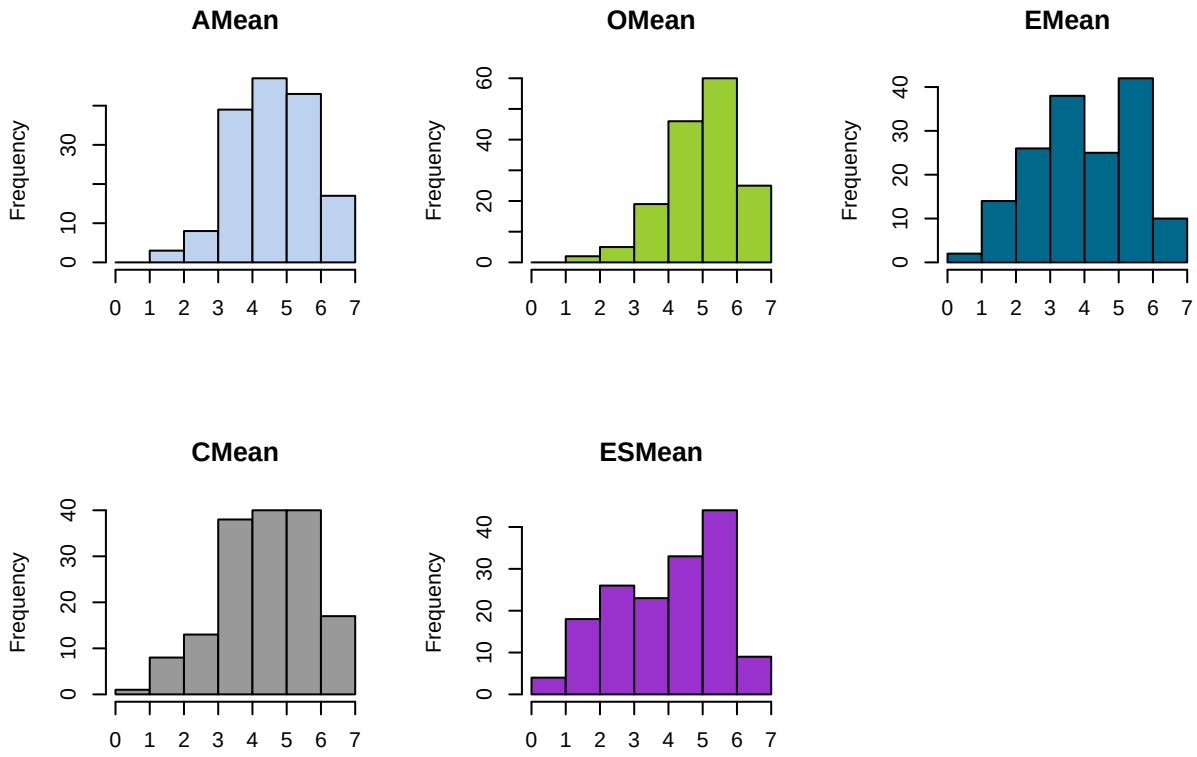



```
describe(data_Q_total[, b5])
```

```
##                          vars   n mean   sd median trimmed  mad min max range
## agreeablenessNormal         1 157 5.09 1.27      5    5.16 1.48   2   7     5
## emotionalStabilityNormal    2 157 4.47 1.58      5    4.54 1.48   1   7     6
## extroversionNormal          3 157 4.44 1.57      5    4.51 1.48   1   7     6
## conscientiousnessNormal     4 157 4.95 1.41      5    5.04 1.48   1   7     6
## opennessNormal              5 157 5.50 1.07      6    5.58 1.48   2   7     5
## agreeablenessReverse        6 157 4.55 1.50      5    4.58 1.48   1   7     6
## emotionalStabilityReverse   7 157 4.39 1.79      5    4.45 1.48   1   7     6
## extroversionReverse         8 157 4.12 1.70      4    4.12 2.97   1   7     6
## conscientiousnessReverse    9 157 4.34 1.71      4    4.38 1.48   1   7     6
## opennessReverse            10 157 4.99 1.53      5    5.12 1.48   1   7     6
##                           skew kurtosis   se
## agreeablenessNormal      -0.50    -0.37 0.10
## emotionalStabilityNormal -0.35    -0.97 0.13
## extroversionNormal       -0.39    -0.89 0.13
## conscientiousnessNormal  -0.70    -0.05 0.11
## opennessNormal           -0.67     0.35 0.09
## agreeablenessReverse     -0.21    -0.91 0.12
## emotionalStabilityReverse -0.32   -1.13 0.14
## extroversionReverse      -0.04    -1.22 0.14
## conscientiousnessReverse -0.17    -1.05 0.14
## opennessReverse          -0.76     0.12 0.12
```

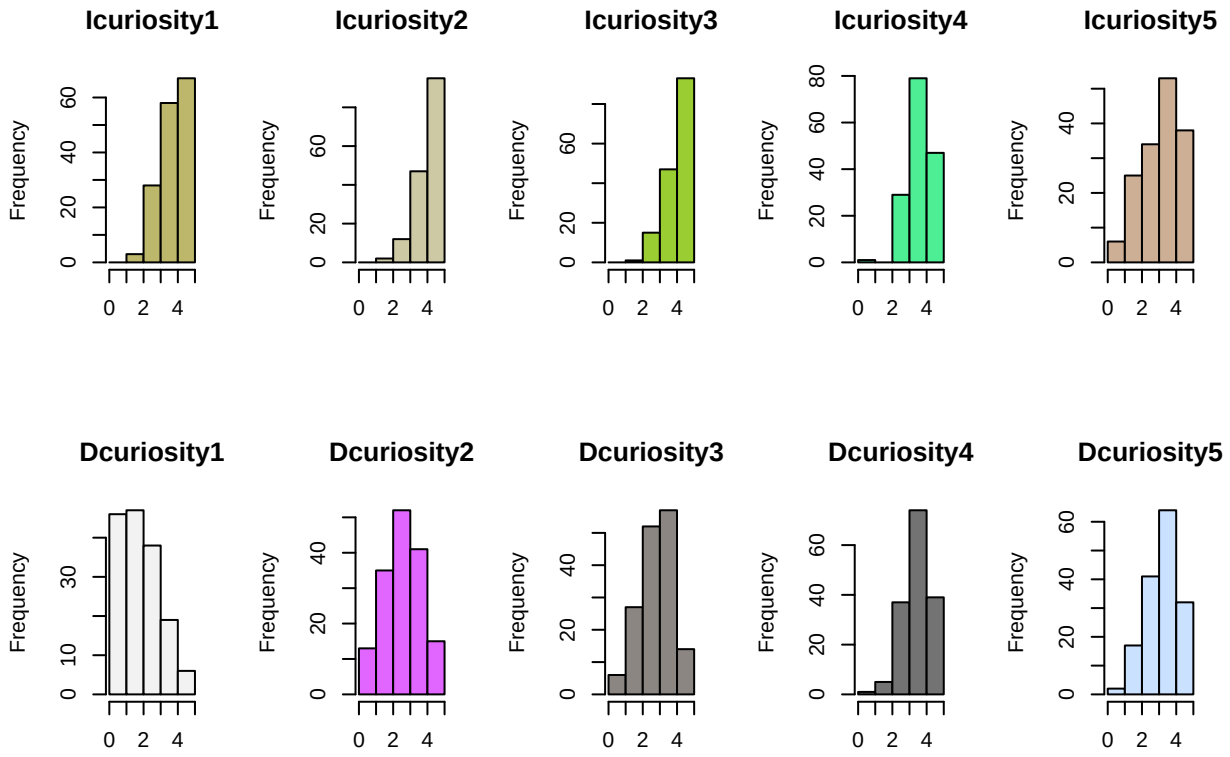
```r
# Calculate means
data_Q_total$AMean <- rowMeans(cbind(data_Q_total$agreeablenessNormal,
                                     data_Q_total$agreeablenessReverse),
                               na.rm = TRUE)
data_Q_total$CMean <- rowMeans(cbind(data_Q_total$conscientiousnessNormal,
                               na.rm = TRUE)
data_Q_total$EMean <- rowMeans(cbind(data_Q_total$extroversionNormal,
                                     data_Q_total$extroversionReverse),
                               na.rm = TRUE)
data_Q_total$ESMean <- rowMeans(cbind(data_Q_total$emotionalStabilityNormal,
                                      data_Q_total$emostabilityReverse),
                                na.rm = TRUE)
data_Q_total$OMean <- rowMeans(cbind(data_Q_total$opennessNormal,
                                     data_Q_total$opennessReverse),
                               na.rm = TRUE)

# Replace mean score 0s with NAs (these participants stopped mid-questionnaire or before questionnaire
data_Q_total$AMean[which(data_Q_total$AMean == 0)] <- NA
data_Q_total$OMean[which(data_Q_total$OMean == 0)] <- NA
data_Q_total$EMean[which(data_Q_total$EMean == 0)] <- NA
data_Q_total$CMean[which(data_Q_total$CMean == 0)] <- NA
data_Q_total$ESMean[which(data_Q_total$ESMean == 0)] <- NA

# Describe: mean score level
b5_mean <- c("AMean", "OMean", "EMean", "CMean", "ESMean")
par(mfrow = c(2, 3))
for (i in 1:length(b5_mean)) {
  hist(data_Q_total[, b5_mean[i]],
       main = colnames(data_Q_total[b5_mean[i]]),
       col = sample(colors(), 1),
       breaks = seq(0, 7, 1),
       xlab = ""
       )
}
describe(data_Q_total[, b5_mean])
```

```
##        vars   n mean   sd median trimmed  mad min max range  skew kurtosis   se
## AMean     1 157 4.82 1.15    5.0    4.86 1.48 1.5   7   5.5 -0.39    -0.14 0.09
## OMean     2 157 5.25 1.07    5.5    5.30 0.74 1.5   7   5.5 -0.60     0.50 0.09
## EMean     3 157 4.28 1.47    4.0    4.32 2.22 1.0   7   6.0 -0.17    -0.97 0.12
## CMean     4 157 4.64 1.31    4.5    4.72 1.48 1.0   7   6.0 -0.44    -0.37 0.10
## ESMean    5 157 4.47 1.58    5.0    4.54 1.48 1.0   7   6.0 -0.35    -0.97 0.13
```



Epistemic Curiosity

```
# Epistemic Curiosity
epicur <- c(paste0("Icuriosity", 1:5), paste0("Dcuriosity", 1:5))

# Describe: item-level
par(mfrow = c(2, 5))
for (i in 1:length(epicur)) {
  hist(data_Q_total[, epicur[i]],
       main = colnames(data_Q_total[epicur[i]]),
       col = sample(colors(), 1),
       breaks = seq(0, 5, 1),
       xlab = ""
       )
}
```




```
describe(data_Q_total[, epicur])
```

```
##             vars   n mean   sd median trimmed  mad min max range  skew kurtosis
## Icuriosity1    1 156 4.21 0.80      4    4.29 1.48   2   5     3 -0.62    -0.57
## Icuriosity2    2 156 4.51 0.70      5    4.63 0.00   2   5     3 -1.28     1.09
## Icuriosity3    3 156 4.49 0.70      5    4.61 0.00   2   5     3 -1.10     0.31
## Icuriosity4    4 156 4.10 0.73      4    4.13 0.00   1   5     4 -0.54     0.64
## Icuriosity5    5 156 3.59 1.14      4    3.66 1.48   1   5     4 -0.44    -0.74
## Dcuriosity1    6 156 2.31 1.13      2    2.21 1.48   1   5     4  0.52    -0.61
## Dcuriosity2    7 156 3.06 1.10      3    3.06 1.48   1   5     4 -0.07    -0.71
## Dcuriosity3    8 156 3.29 0.99      3    3.30 1.48   1   5     4 -0.29    -0.45
## Dcuriosity4    9 156 3.93 0.82      4    3.97 1.48   1   5     4 -0.50     0.14
## Dcuriosity5   10 156 3.69 0.96      4    3.75 1.48   1   5     4 -0.42    -0.41
##                se
## Icuriosity1 0.06
## Icuriosity2 0.06
## Icuriosity3 0.06
## Icuriosity4 0.06
## Icuriosity5 0.09
## Dcuriosity1 0.09
## Dcuriosity2 0.09
## Dcuriosity3 0.08
## Dcuriosity4 0.07
## Dcuriosity5 0.08
```

```r
# Cronbach's alphas
psych::alpha(data_Q_total[, epicur[1:5]])
```

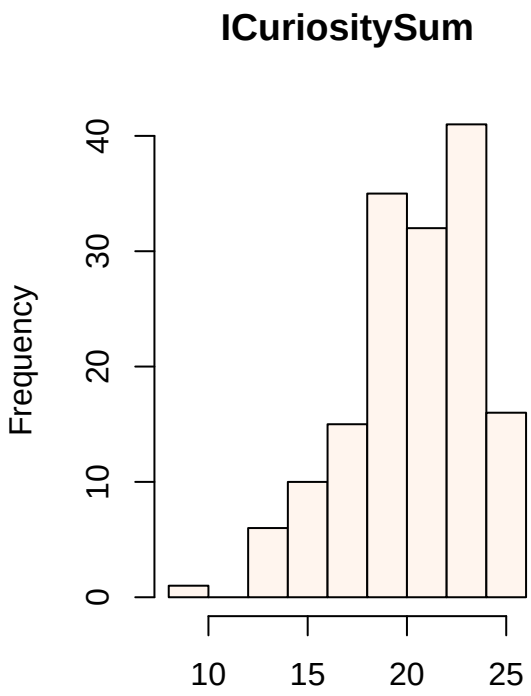
```
##
## Reliability analysis
## Call: psych::alpha(x = data_Q_total[, epicur[1:5]])
##
##   raw_alpha std.alpha G6(smc) average_r S/N   ase mean   sd median_r
##        0.8      0.83    0.82      0.49 4.7 0.023  4.2 0.62     0.48
##
##  lower alpha upper     95% confidence boundaries
## 0.76 0.8 0.85
##
##  Reliability if an item is dropped:
##             raw_alpha std.alpha G6(smc) average_r S/N alpha se  var.r med.r
## Icuriosity1      0.75      0.79    0.78      0.48 3.7    0.031 0.0228  0.43
## Icuriosity2      0.75      0.77    0.72      0.46 3.3    0.030 0.0018  0.46
## Icuriosity3      0.75      0.77    0.72      0.46 3.3    0.030 0.0032  0.48
## Icuriosity4      0.77      0.81    0.79      0.51 4.2    0.028 0.0199  0.49
## Icuriosity5      0.82      0.82    0.80      0.53 4.5    0.023 0.0157  0.50
##
##  Item statistics
##                n raw.r std.r r.cor r.drop mean   sd
## Icuriosity1 156  0.78  0.77  0.69   0.63  4.2 0.80
## Icuriosity2 156  0.78  0.82  0.80   0.66  4.5 0.70
## Icuriosity3 156  0.79  0.82  0.80   0.67  4.5 0.70
## Icuriosity4 156  0.72  0.73  0.62   0.57  4.1 0.73
## Icuriosity5 156  0.76  0.70  0.57   0.52  3.6 1.14
##
## Non missing response frequency for each item
##                1    2    3    4    5 miss
## Icuriosity1 0.00 0.02 0.18 0.37 0.43  0.1
## Icuriosity2 0.00 0.01 0.08 0.30 0.61  0.1
## Icuriosity3 0.00 0.01 0.10 0.30 0.60  0.1
## Icuriosity4 0.01 0.00 0.19 0.51 0.30  0.1
## Icuriosity5 0.04 0.16 0.22 0.34 0.24  0.1
```

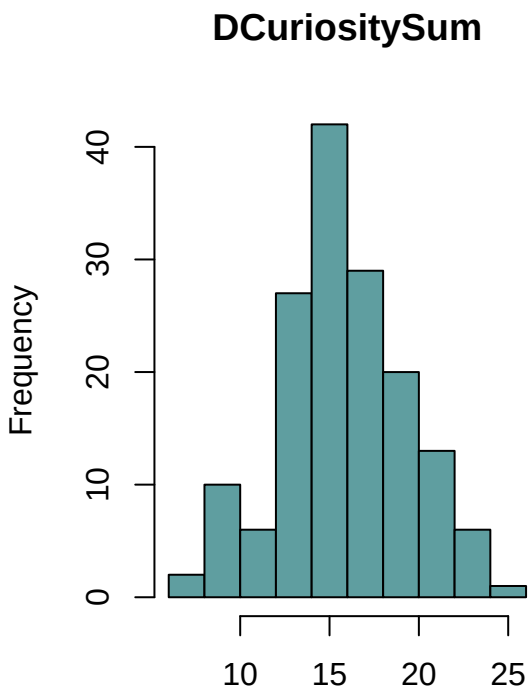
```r
psych::alpha(data_Q_total[, epicur[6:10]])
```

```
##
## Reliability analysis
## Call: psych::alpha(x = data_Q_total[, epicur[6:10]])
##
##   raw_alpha std.alpha G6(smc) average_r S/N   ase mean  sd median_r
##       0.73      0.73    0.72      0.36 2.8 0.031  3.3 0.7     0.36
##
##  lower alpha upper     95% confidence boundaries
## 0.67 0.73 0.8
##
##  Reliability if an item is dropped:
##             raw_alpha std.alpha G6(smc) average_r S/N alpha se  var.r med.r
## Dcuriosity1      0.71      0.71    0.65      0.38 2.4    0.036 0.0027  0.38
```

```
## Dcuriosity2      0.62      0.63    0.57      0.30 1.7    0.048 0.0049  0.28
## Dcuriosity3      0.70      0.70    0.66      0.36 2.3    0.037 0.0150  0.37
## Dcuriosity4      0.70      0.70    0.67      0.37 2.3    0.036 0.0172  0.35
## Dcuriosity5      0.70      0.70    0.67      0.37 2.4    0.036 0.0161  0.36
##
##  Item statistics
##                n raw.r std.r r.cor r.drop mean   sd
## Dcuriosity1 156  0.69  0.66  0.55   0.46  2.3 1.13
## Dcuriosity2 156  0.82  0.79  0.76   0.66  3.1 1.10
## Dcuriosity3 156  0.68  0.68  0.56   0.47  3.3 0.99
## Dcuriosity4 156  0.63  0.68  0.54   0.46  3.9 0.82
## Dcuriosity5 156  0.66  0.67  0.54   0.45  3.7 0.96
##
## Non missing response frequency for each item
##                   1    2    3    4    5 miss
## Dcuriosity1 0.29 0.30 0.24 0.12 0.04  0.1
## Dcuriosity2 0.08 0.22 0.33 0.26 0.10  0.1
## Dcuriosity3 0.04 0.17 0.33 0.37 0.09  0.1
## Dcuriosity4 0.01 0.03 0.24 0.47 0.25  0.1
## Dcuriosity5 0.01 0.11 0.26 0.41 0.21  0.1
```

```r
# Calculate sum scores
data_Q_total$ICuriositySum <- rowSums(cbind(data_Q_total$Icuriosity1, data_Q_total$Icuriosity2, data_Q_
data_Q_total$DCuriositySum <- rowSums(cbind(data_Q_total$Dcuriosity1, data_Q_total$Dcuriosity2, data_Q_

# Replace sum score 0s with NAs (these participants stopped mid-questionnaire or before questionnaire a
data_Q_total$ICuriositySum[which(data_Q_total$ICuriositySum == 0)] <- NA
data_Q_total$DCuriositySum[which(data_Q_total$DCuriositySum == 0)] <- NA

# Describe: sum score level
epicur_sum <- c("ICuriositySum", "DCuriositySum")
par(mfrow = c(1, 2))
for (i in 1:length(epicur_sum)) {
  hist(data_Q_total[, epicur_sum[i]],
       main = colnames(data_Q_total[epicur_sum[i]]),
       col = sample(colors(), 1),
       xlab = ""
       )
}
```





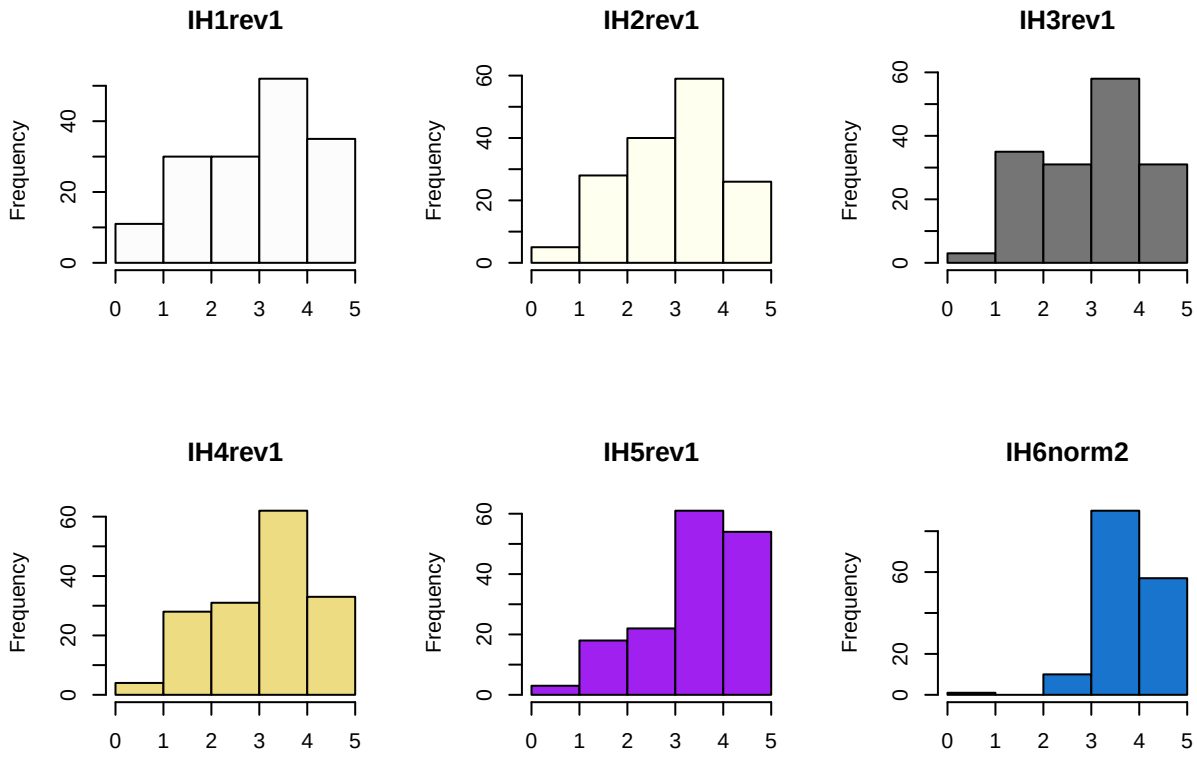

```
describe(data_Q_total[, c(epicur_sum)])
```

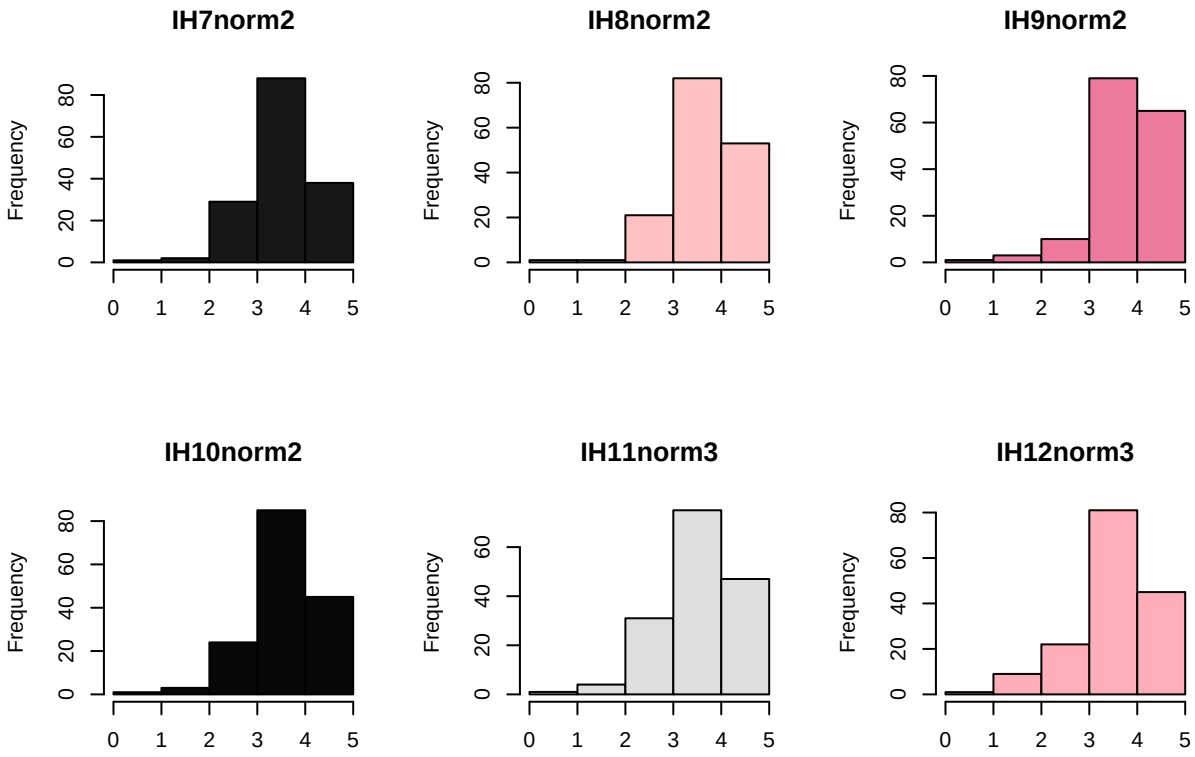
```
##                vars   n  mean  sd median trimmed  mad min max range  skew
## ICuriositySum     1 156 20.89 3.1     21   21.14 2.97   9  25    16 -0.76
## DCuriositySum     2 156 16.28 3.5     16   16.31 2.97   7  25    18 -0.05
##                kurtosis   se
## ICuriositySum      0.39 0.25
## DCuriositySum     -0.16 0.28
```

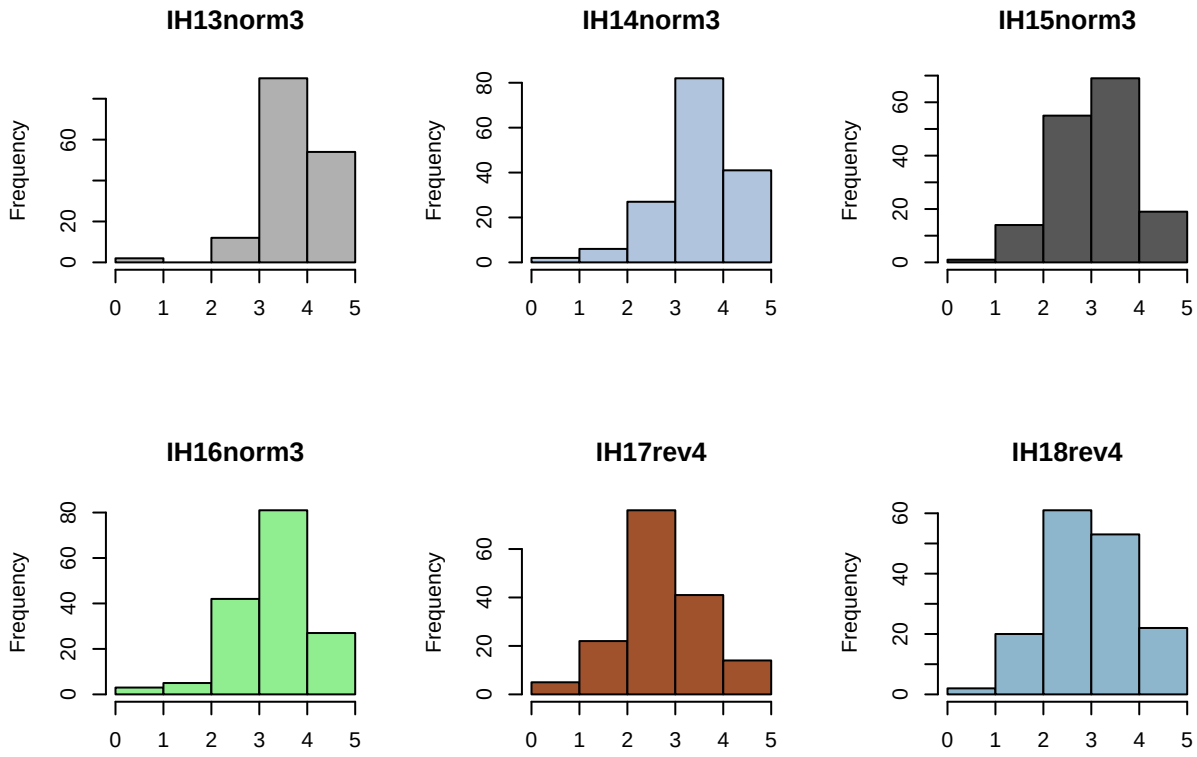
Intellectual Humility

```
# Intellectual Humility
IH <- c(paste0("IH", 1:5, "rev1"), paste0("IH", 6:10, "norm2"),
        paste0("IH", 11:16, "norm3"), paste0("IH", 17:22, "rev4"))
IH_rev <- c(paste0("IH", 1:5, "rev1"), paste0("IH", 17:22, "rev4"))

# Reverse-code
for (i in 1:length(IH_rev)) {
  data_Q_total[, IH_rev[i]] <- reverse.code(keys = c(-1),
                                            items = data_Q_total[, IH_rev[i]],
                                            mini = c(1),
                                            maxi = c(5))
}
```

```r
# Describe: item-level
par(mfrow = c(2, 3))
for (i in 1:length(IH)) {
  hist(data_Q_total[, IH[i]],
       main = colnames(data_Q_total[IH[i]]),
       col = sample(colors(), 1),
       breaks = seq(0, 5, 1),
       xlab = ""
       )
}
```




IH7norm2
IH8norm2
IH9norm2
IH10norm2
IH11norm3
IH12norm3
Frequency
0
20
40
60
80
0
1
2
3
4
5




```
describe(data_Q_total[, IH])
```

```
##           vars   n mean   sd median trimmed  mad min max range  skew kurtosis
## IH1rev1      1 158 3.44 1.22      4    3.52 1.48   1   5     4 -0.39    -0.93
## IH2rev1      2 158 3.46 1.06      4    3.49 1.48   1   5     4 -0.33    -0.70
## IH3rev1      3 158 3.50 1.10      4    3.52 1.48   1   5     4 -0.27    -1.02
## IH4rev1      4 158 3.58 1.08      4    3.63 1.48   1   5     4 -0.43    -0.74
## IH5rev1      5 158 3.92 1.05      4    4.04 1.48   1   5     4 -0.81    -0.16
## IH6norm2     6 158 4.28 0.64      4    4.33 0.00   1   5     4 -0.90     3.07
## IH7norm2     7 158 4.01 0.73      4    4.05 0.00   1   5     4 -0.60     1.02
## IH8norm2     8 158 4.17 0.72      4    4.23 0.00   1   5     4 -0.77     1.32
## IH9norm2     9 158 4.29 0.73      4    4.39 0.74   1   5     4 -1.19     2.60
## IH10norm2   10 158 4.08 0.75      4    4.13 0.00   1   5     4 -0.75     1.13
## IH11norm3   11 158 4.03 0.81      4    4.09 1.48   1   5     4 -0.63     0.38
## IH12norm3   12 158 4.01 0.84      4    4.10 0.00   1   5     4 -0.84     0.68
## IH13norm3   13 158 4.23 0.69      4    4.30 0.00   1   5     4 -1.24     4.16
## IH14norm3   14 158 3.97 0.84      4    4.05 0.00   1   5     4 -0.86     1.10
## IH15norm3   15 158 3.58 0.84      4    3.60 1.48   1   5     4 -0.24    -0.23
## IH16norm3   16 158 3.78 0.83      4    3.82 0.00   1   5     4 -0.71     1.04
## IH17rev4    17 158 3.23 0.91      3    3.22 1.48   1   5     4 -0.02    -0.01
## IH18rev4    18 158 3.46 0.93      3    3.47 1.48   1   5     4 -0.08    -0.48
## IH19rev4    19 158 3.20 1.05      3    3.18 1.48   1   5     4 -0.03    -0.69
## IH20rev4    20 158 3.58 0.92      4    3.60 1.48   1   5     4 -0.39    -0.52
## IH21rev4    21 158 4.25 0.78      4    4.37 1.48   1   5     4 -1.34     2.96
## IH22rev4    22 158 3.42 0.88      3    3.42 1.48   1   5     4 -0.19    -0.29
```

```
##             se
## IH1rev1   0.10
## IH2rev1   0.08
## IH3rev1   0.09
## IH4rev1   0.09
## IH5rev1   0.08
## IH6norm2  0.05
## IH7norm2  0.06
## IH8norm2  0.06
## IH9norm2  0.06
## IH10norm2 0.06
## IH11norm3 0.06
## IH12norm3 0.07
## IH13norm3 0.06
## IH14norm3 0.07
## IH15norm3 0.07
## IH16norm3 0.07
## IH17rev4  0.07
## IH18rev4  0.07
## IH19rev4  0.08
## IH20rev4  0.07
## IH21rev4  0.06
## IH22rev4  0.07
```

```
# Cronbach's alphas
psych::alpha(data_Q_total[, IH[1:5]])
```

```
##
## Reliability analysis
## Call: psych::alpha(x = data_Q_total[, IH[1:5]])
##
##   raw_alpha std.alpha G6(smc) average_r S/N   ase mean  sd median_r
##       0.87      0.87    0.88      0.58 6.9 0.016  3.6 0.9     0.55
##
##  lower alpha upper     95% confidence boundaries
## 0.84 0.87 0.9
##
##  Reliability if an item is dropped:
##         raw_alpha std.alpha G6(smc) average_r S/N alpha se  var.r med.r
## IH1rev1      0.87      0.87    0.86      0.63 6.8    0.016 0.0191  0.63
## IH2rev1      0.82      0.83    0.82      0.54 4.8    0.022 0.0138  0.55
## IH3rev1      0.84      0.84    0.81      0.57 5.2    0.020 0.0067  0.55
## IH4rev1      0.82      0.83    0.84      0.54 4.8    0.023 0.0245  0.49
## IH5rev1      0.86      0.86    0.86      0.61 6.2    0.019 0.0288  0.63
##
##  Item statistics
##           n raw.r std.r r.cor r.drop mean  sd
## IH1rev1 158  0.75  0.74  0.65   0.59  3.4 1.2
## IH2rev1 158  0.86  0.86  0.85   0.77  3.5 1.1
## IH3rev1 158  0.82  0.83  0.81   0.71  3.5 1.1
## IH4rev1 158  0.86  0.86  0.82   0.78  3.6 1.1
## IH5rev1 158  0.77  0.77  0.68   0.64  3.9 1.1
##
## Non missing response frequency for each item
```

```
##            1    2    3    4    5 miss
## IH1rev1 0.07 0.19 0.19 0.33 0.22 0.09
## IH2rev1 0.03 0.18 0.25 0.37 0.16 0.09
## IH3rev1 0.02 0.22 0.20 0.37 0.20 0.09
## IH4rev1 0.03 0.18 0.20 0.39 0.21 0.09
## IH5rev1 0.02 0.11 0.14 0.39 0.34 0.09
```

```
psych::alpha(data_Q_total[, IH[6:10]])
```

```
##
## Reliability analysis
## Call: psych::alpha(x = data_Q_total[, IH[6:10]])
##
##   raw_alpha std.alpha G6(smc) average_r S/N   ase mean   sd median_r
##       0.85      0.85    0.82      0.53 5.5 0.018  4.2 0.56     0.53
##
##  lower alpha upper     95% confidence boundaries
## 0.81 0.85 0.88
##
##  Reliability if an item is dropped:
##           raw_alpha std.alpha G6(smc) average_r S/N alpha se  var.r med.r
## IH6norm2       0.82      0.82    0.78      0.54 4.7    0.022 0.0022  0.53
## IH7norm2       0.81      0.81    0.77      0.52 4.4    0.023 0.0033  0.53
## IH8norm2       0.80      0.80    0.76      0.50 4.1    0.025 0.0021  0.51
## IH9norm2       0.81      0.81    0.77      0.52 4.3    0.024 0.0034  0.52
## IH10norm2      0.83      0.83    0.79      0.55 4.8    0.021 0.0016  0.54
##
##  Item statistics
##              n raw.r std.r r.cor r.drop mean   sd
## IH6norm2   158  0.75  0.77  0.68   0.62  4.3 0.64
## IH7norm2   158  0.80  0.79  0.72   0.66  4.0 0.73
## IH8norm2   158  0.82  0.82  0.77   0.71  4.2 0.72
## IH9norm2   158  0.80  0.80  0.74   0.68  4.3 0.73
## IH10norm2 158  0.76  0.75  0.66   0.61  4.1 0.75
##
## Non missing response frequency for each item
##               1    2    3    4    5 miss
## IH6norm2   0.01 0.00 0.06 0.57 0.36 0.09
## IH7norm2   0.01 0.01 0.18 0.56 0.24 0.09
## IH8norm2   0.01 0.01 0.13 0.52 0.34 0.09
## IH9norm2   0.01 0.02 0.06 0.50 0.41 0.09
## IH10norm2 0.01 0.02 0.15 0.54 0.28 0.09
```

```
psych::alpha(data_Q_total[, IH[11:16]])
```

```
##
## Reliability analysis
## Call: psych::alpha(x = data_Q_total[, IH[11:16]])
##
##   raw_alpha std.alpha G6(smc) average_r S/N   ase mean   sd median_r
##       0.84      0.84    0.84      0.47 5.4 0.018  3.9 0.61     0.47
##
##  lower alpha upper     95% confidence boundaries
```

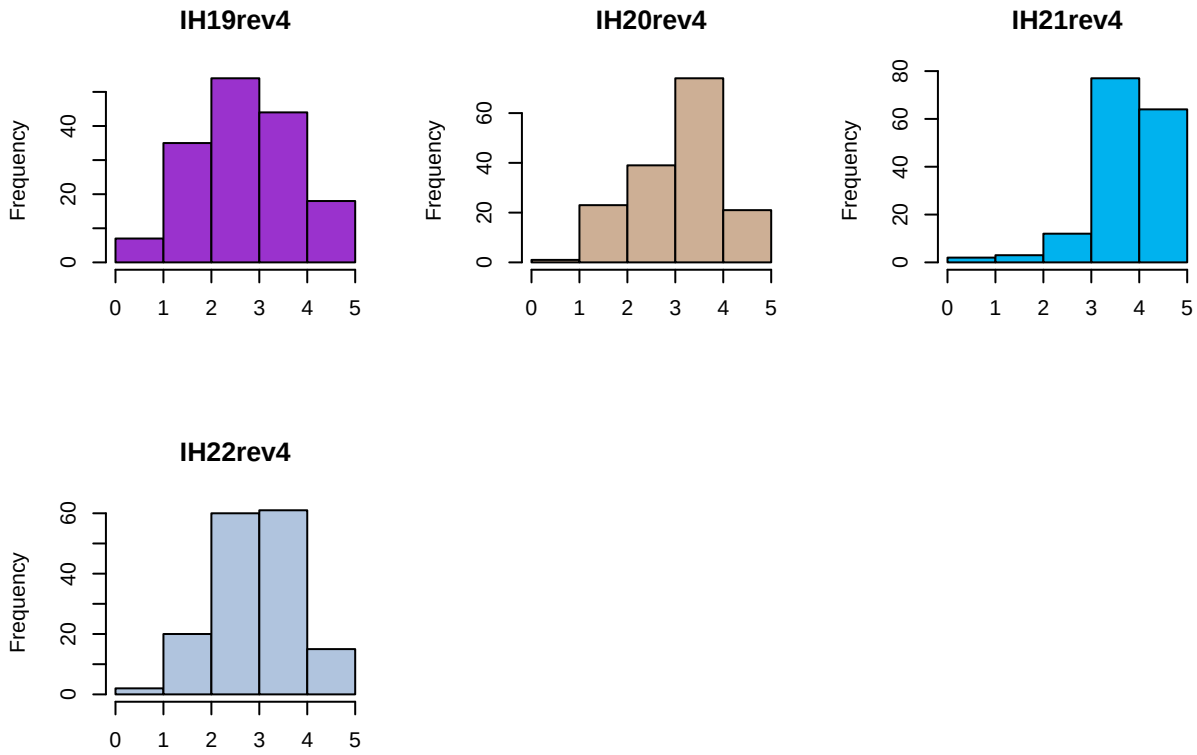
```
## 0.81 0.84 0.88
##
##  Reliability if an item is dropped:
##           raw_alpha std.alpha G6(smc) average_r S/N alpha se  var.r med.r
## IH11norm3      0.80      0.80    0.79      0.45 4.1    0.024 0.0079  0.44
## IH12norm3      0.82      0.82    0.80      0.48 4.6    0.021 0.0077  0.48
## IH13norm3      0.83      0.83    0.81      0.49 4.8    0.021 0.0058  0.48
## IH14norm3      0.81      0.81    0.79      0.46 4.3    0.022 0.0088  0.44
## IH15norm3      0.82      0.82    0.81      0.48 4.6    0.022 0.0094  0.45
## IH16norm3      0.82      0.82    0.81      0.48 4.7    0.021 0.0093  0.45
##
##  Item statistics
##              n raw.r std.r r.cor r.drop mean   sd
## IH11norm3 158  0.81  0.81  0.77   0.71  4.0 0.81
## IH12norm3 158  0.74  0.74  0.67   0.60  4.0 0.84
## IH13norm3 158  0.69  0.71  0.64   0.57  4.2 0.69
## IH14norm3 158  0.77  0.77  0.72   0.65  4.0 0.84
## IH15norm3 158  0.75  0.74  0.67   0.61  3.6 0.84
## IH16norm3 158  0.74  0.73  0.65   0.60  3.8 0.83
##
## Non missing response frequency for each item
##                1    2    3    4    5 miss
## IH11norm3 0.01 0.03 0.20 0.47 0.30 0.09
## IH12norm3 0.01 0.06 0.14 0.51 0.28 0.09
## IH13norm3 0.01 0.00 0.08 0.57 0.34 0.09
## IH14norm3 0.01 0.04 0.17 0.52 0.26 0.09
## IH15norm3 0.01 0.09 0.35 0.44 0.12 0.09
## IH16norm3 0.02 0.03 0.27 0.51 0.17 0.09
```

```
psych::alpha(data_Q_total[, IH[17:22]])
```

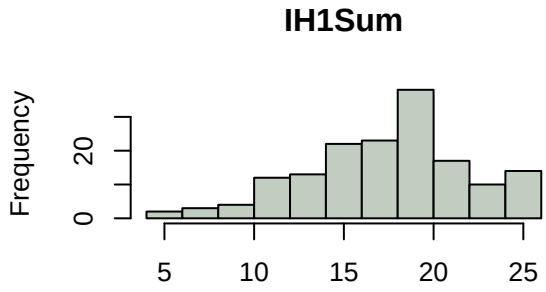
```
##
## Reliability analysis
## Call: psych::alpha(x = data_Q_total[, IH[17:22]])
##
##   raw_alpha std.alpha G6(smc) average_r S/N   ase mean   sd median_r
##       0.76      0.77    0.75      0.35 3.3 0.028  3.5 0.62     0.32
##
##  lower alpha upper     95% confidence boundaries
## 0.71 0.76 0.82
##
##  Reliability if an item is dropped:
##           raw_alpha std.alpha G6(smc) average_r S/N alpha se  var.r med.r
## IH17rev4      0.73      0.73    0.71      0.36 2.8    0.033 0.0085  0.32
## IH18rev4      0.72      0.73    0.71      0.35 2.7    0.033 0.0100  0.32
## IH19rev4      0.72      0.72    0.69      0.34 2.6    0.033 0.0070  0.32
## IH20rev4      0.75      0.75    0.72      0.38 3.0    0.030 0.0067  0.36
## IH21rev4      0.73      0.73    0.71      0.36 2.8    0.032 0.0096  0.33
## IH22rev4      0.72      0.72    0.69      0.34 2.6    0.033 0.0089  0.31
##
##  Item statistics
##              n raw.r std.r r.cor r.drop mean   sd
## IH17rev4 158  0.68  0.68  0.58   0.51  3.2 0.91
## IH18rev4 158  0.70  0.69  0.60   0.53  3.5 0.93
```

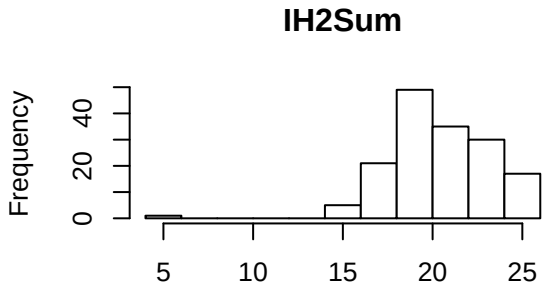
```
## IH19rev4 158  0.73  0.70  0.63   0.54  3.2 1.05
## IH20rev4 158  0.62  0.62  0.51   0.43  3.6 0.92
## IH21rev4 158  0.65  0.67  0.58   0.50  4.3 0.78
## IH22rev4 158  0.70  0.71  0.64   0.55  3.4 0.88
##
## Non missing response frequency for each item
##              1    2    3    4    5 miss
## IH17rev4 0.03 0.14 0.48 0.26 0.09 0.09
## IH18rev4 0.01 0.13 0.39 0.34 0.14 0.09
## IH19rev4 0.04 0.22 0.34 0.28 0.11 0.09
## IH20rev4 0.01 0.15 0.25 0.47 0.13 0.09
## IH21rev4 0.01 0.02 0.08 0.49 0.41 0.09
## IH22rev4 0.01 0.13 0.38 0.39 0.09 0.09
```

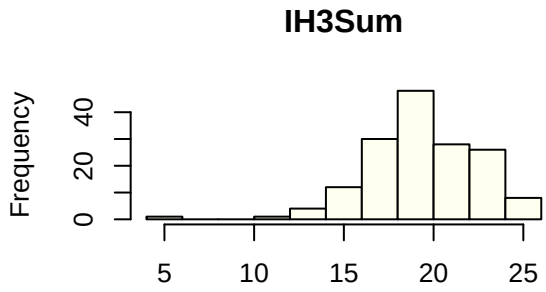
```r
# Intellectual Humility Scoring
# https://seaver.pepperdine.edu/social-science/content/comprehensive-intellectual-humility.pdf -> sum s

# Calculate sum scores
data_Q_total$IH1Sum <- rowSums(cbind(data_Q_total$IH1rev1, data_Q_total$IH2rev1, data_Q_total$IH3rev1, c
data_Q_total$IH2Sum <- rowSums(cbind(data_Q_total$IH6norm2, data_Q_total$IH7norm2, data_Q_total$IH8norm1
data_Q_total$IH3Sum <- rowSums(cbind(data_Q_total$IH11norm3, data_Q_total$IH12norm3, data_Q_total$IH13n
data_Q_total$IH4Sum <- rowSums(cbind(data_Q_total$IH17rev4, data_Q_total$IH18rev4, data_Q_total$IH19rev4

# Replace sum score 0s with NAs
data_Q_total$IH1Sum[which(data_Q_total$IH1Sum == 0)] <- NA
data_Q_total$IH2Sum[which(data_Q_total$IH2Sum == 0)] <- NA
data_Q_total$IH3Sum[which(data_Q_total$IH3Sum == 0)] <- NA
data_Q_total$IH4Sum[which(data_Q_total$IH4Sum == 0)] <- NA

# Describe: sum score level
IH_sum <- c("IH1Sum", "IH2Sum", "IH3Sum", "IH4Sum")
par(mfrow = c(2, 2))
```

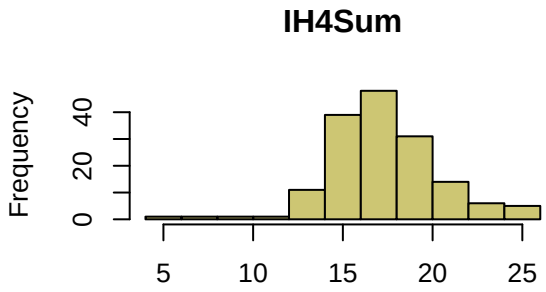



```
for (i in 1:length(IH_sum)) {
  hist(data_Q_total[, IH_sum[i]],
       main = colnames(data_Q_total[IH_sum[i]]),
       col = sample(colors(), 1),
       xlab = ""
       )
}
```









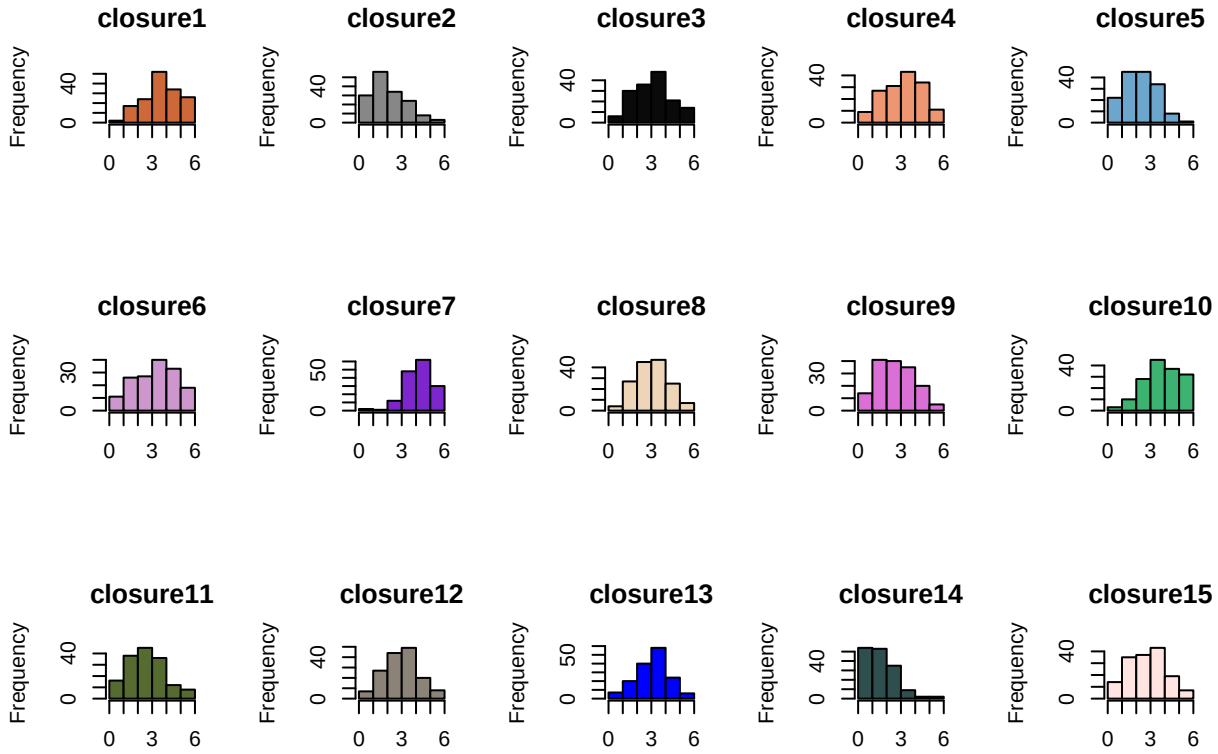

```
describe(data_Q_total[, IH_sum])
```

```
##         vars   n  mean   sd median trimmed  mad min max range  skew kurtosis
## IH1Sum     1 158 17.91 4.49   18.5   18.09 5.19   5  25    20 -0.41    -0.29
## IH2Sum     2 158 20.83 2.82   21.0   20.95 2.97   5  25    20 -1.09     4.77
## IH3Sum     3 158 19.82 3.08   20.0   19.95 2.97   5  25    20 -0.76     2.25
## IH4Sum     4 158 17.72 3.17   17.0   17.69 2.97   5  25    20 -0.25     1.90
##           se
## IH1Sum 0.36
## IH2Sum 0.22
## IH3Sum 0.25
## IH4Sum 0.25
```

Need for closure

```
# Need for Closure
need_clo <- paste0("closure", 1:15)

# Describe: item-level
par(mfrow = c(3, 5))
for (i in 1:length(need_clo)) {
  hist(data_Q_total[, need_clo[i]],
       main = colnames(data_Q_total[need_clo[i]]),
```

```
        breaks = seq(0, 6, 1),
        col = sample(colors(), 1),
        xlab = ""
        )
}
```




```
describe(data_Q_total[, need_clo])
```

```
##           vars   n mean   sd median trimmed  mad min max range  skew kurtosis
## closure1     1 155 4.14 1.26      4    4.19 1.48   1   6     5 -0.25    -0.65
## closure2     2 155 2.57 1.22      2    2.47 1.48   1   6     5  0.65    -0.15
## closure3     3 155 3.58 1.30      4    3.54 1.48   1   6     5  0.10    -0.68
## closure4     4 155 3.64 1.34      4    3.66 1.48   1   6     5 -0.16    -0.83
## closure5     5 155 2.77 1.14      3    2.75 1.48   1   6     5  0.20    -0.63
## closure6     6 155 3.72 1.45      4    3.74 1.48   1   6     5 -0.18    -0.93
## closure7     7 155 4.66 0.98      5    4.74 1.48   1   6     5 -0.75     1.18
## closure8     8 155 3.54 1.16      4    3.52 1.48   1   6     5  0.07    -0.56
## closure9     9 155 3.14 1.28      3    3.12 1.48   1   6     5  0.22    -0.73
## closure10   10 155 4.28 1.27      4    4.36 1.48   1   6     5 -0.35    -0.54
## closure11   11 155 3.09 1.29      3    3.05 1.48   1   6     5  0.34    -0.41
## closure12   12 155 3.46 1.20      3    3.45 1.48   1   6     5  0.06    -0.44
## closure13   13 155 3.58 1.15      4    3.61 1.48   1   6     5 -0.21    -0.26
## closure14   14 155 2.08 1.06      2    1.95 1.48   1   6     5  1.03     1.23
## closure15   15 155 3.25 1.31      3    3.25 1.48   1   6     5  0.10    -0.71
##             se
```

```
## closure1  0.10
## closure2  0.10
## closure3  0.10
## closure4  0.11
## closure5  0.09
## closure6  0.12
## closure7  0.08
## closure8  0.09
## closure9  0.10
## closure10 0.10
## closure11 0.10
## closure12 0.10
## closure13 0.09
## closure14 0.09
## closure15 0.11
```

```
# Cronbach's alphas
psych::alpha(data_Q_total[, need_clo])
```

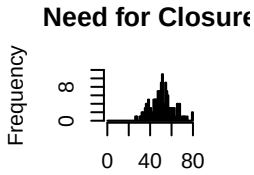
```
##
## Reliability analysis
## Call: psych::alpha(x = data_Q_total[, need_clo])
##
##   raw_alpha std.alpha G6(smc) average_r S/N   ase mean   sd median_r
##       0.84      0.83    0.87      0.25 4.9 0.017  3.4 0.68     0.21
##
##  lower alpha upper     95% confidence boundaries
## 0.8 0.84 0.87
##
##  Reliability if an item is dropped:
##           raw_alpha std.alpha G6(smc) average_r S/N alpha se var.r med.r
## closure1       0.82      0.81    0.85      0.24 4.4    0.019 0.025  0.21
## closure2       0.83      0.82    0.86      0.25 4.6    0.018 0.028  0.21
## closure3       0.82      0.82    0.85      0.24 4.5    0.019 0.025  0.21
## closure4       0.82      0.81    0.85      0.24 4.4    0.019 0.026  0.21
## closure5       0.83      0.82    0.86      0.25 4.7    0.018 0.027  0.21
## closure6       0.82      0.81    0.85      0.23 4.3    0.020 0.023  0.21
## closure7       0.84      0.84    0.87      0.27 5.1    0.017 0.026  0.24
## closure8       0.84      0.83    0.86      0.26 4.9    0.018 0.027  0.24
## closure9       0.84      0.83    0.86      0.26 4.8    0.018 0.027  0.24
## closure10      0.83      0.82    0.86      0.24 4.5    0.019 0.025  0.21
## closure11      0.83      0.82    0.86      0.24 4.5    0.019 0.028  0.21
## closure12      0.83      0.82    0.85      0.24 4.5    0.019 0.025  0.22
## closure13      0.82      0.81    0.85      0.24 4.3    0.019 0.024  0.21
## closure14      0.85      0.84    0.87      0.27 5.3    0.017 0.022  0.24
## closure15      0.82      0.81    0.85      0.24 4.3    0.020 0.025  0.21
##
##  Item statistics
##             n raw.r std.r r.cor r.drop mean   sd
## closure1  155  0.65  0.64  0.62   0.58  4.1 1.26
## closure2  155  0.53  0.53  0.49   0.44  2.6 1.22
## closure3  155  0.62  0.62  0.60   0.53  3.6 1.30
## closure4  155  0.65  0.64  0.62   0.57  3.6 1.34
## closure5  155  0.51  0.51  0.46   0.42  2.8 1.14
```

```
## closure6  155  0.72  0.70  0.70   0.64  3.7 1.45
## closure7  155  0.30  0.32  0.23   0.21  4.7 0.98
## closure8  155  0.40  0.41  0.34   0.30  3.5 1.16
## closure9  155  0.44  0.45  0.39   0.33  3.1 1.28
## closure10 155  0.59  0.59  0.55   0.50  4.3 1.27
## closure11 155  0.59  0.58  0.55   0.50  3.1 1.29
## closure12 155  0.59  0.60  0.58   0.51  3.5 1.20
## closure13 155  0.67  0.68  0.68   0.60  3.6 1.15
## closure14 155  0.23  0.25  0.16   0.13  2.1 1.06
## closure15 155  0.68  0.67  0.66   0.60  3.3 1.31
##
## Non missing response frequency for each item
##              1    2    3    4    5    6 miss
## closure1  0.01 0.11 0.15 0.34 0.22 0.17 0.11
## closure2  0.19 0.36 0.22 0.15 0.05 0.02 0.11
## closure3  0.04 0.19 0.23 0.31 0.14 0.09 0.11
## closure4  0.06 0.17 0.20 0.28 0.22 0.07 0.11
## closure5  0.14 0.29 0.29 0.22 0.05 0.01 0.11
## closure6  0.07 0.17 0.17 0.26 0.21 0.12 0.11
## closure7  0.01 0.01 0.08 0.31 0.40 0.19 0.11
## closure8  0.03 0.17 0.29 0.30 0.16 0.05 0.11
## closure9  0.09 0.26 0.26 0.23 0.13 0.03 0.11
## closure10 0.02 0.06 0.18 0.29 0.24 0.21 0.11
## closure11 0.10 0.25 0.29 0.23 0.08 0.05 0.11
## closure12 0.05 0.17 0.28 0.32 0.13 0.05 0.11
## closure13 0.05 0.13 0.26 0.37 0.15 0.04 0.11
## closure14 0.35 0.34 0.23 0.06 0.01 0.01 0.11
## closure15 0.09 0.23 0.24 0.28 0.12 0.05 0.11
```

```r
# Need for Closure Scoring
# https://www.midss.org/sites/default/files/need_for_closure_scale.pdf
# Not entirely sure if this is the correct file but it says to sum so I'll do that here

# Calculate sum score
data_Q_total$CloSum <- rowSums(cbind(data_Q_total$closure1, data_Q_total$closure2, data_Q_total$closure3

# Replace sum score 0s with NAs
data_Q_total$CloSum[which(data_Q_total$CloSum == 0)] <- NA

# Describe: sum score level
describe(data_Q_total$CloSum)
```

```
##    vars   n mean    sd median trimmed mad min max range skew kurtosis   se
## X1    1 155 51.5 10.23     51   51.33 8.9  27  80    53 0.19     0.03 0.82
```

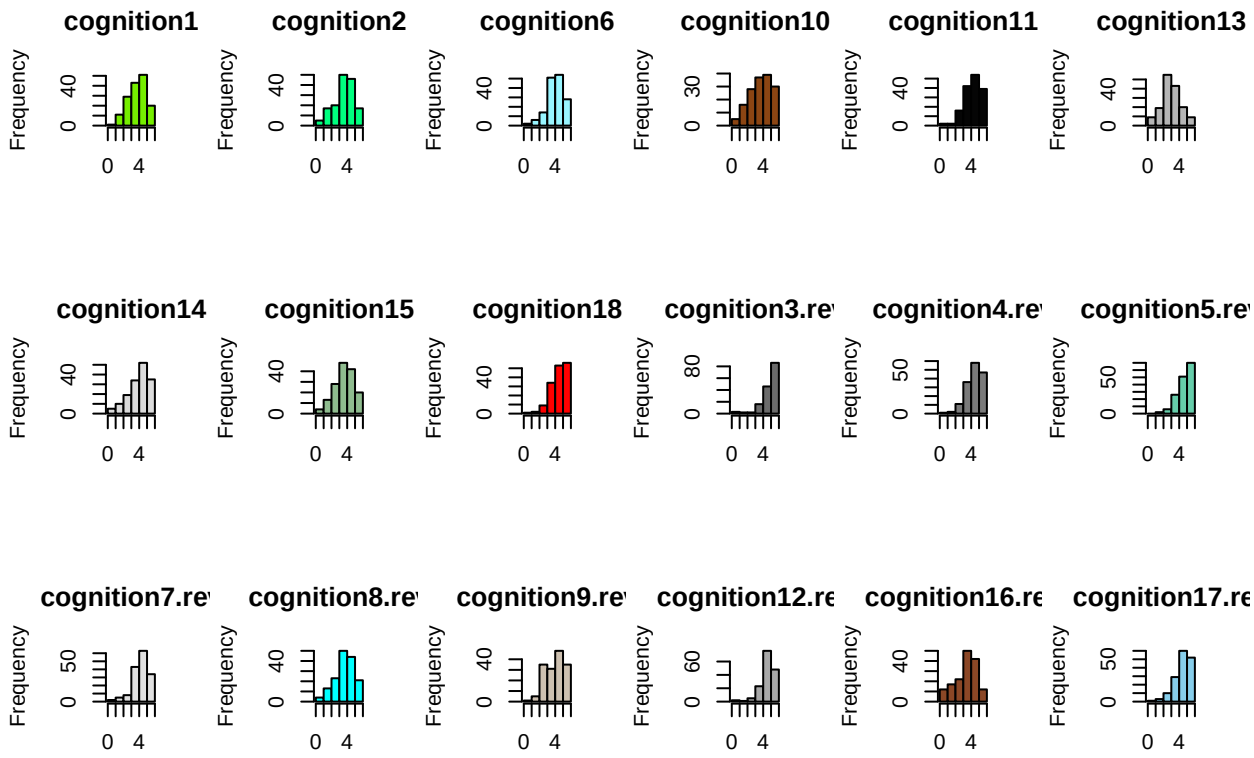
```r
hist(data_Q_total$CloSum,
     main = "Need for Closure",
     col = rainbow(14),
     ylim = c(0, 12),
     breaks = seq(0, 80, 1),
     xlab = "")
```



Need for cognition

```
# Need for Cognition
need_cog <- c(paste0("cognition", c(1:2, 6, 10:11, 13:15, 18)),
              paste0("cognition", c(3:5, 7:9, 12, 16:17), ".rev"))
need_cog_rev <- paste0("cognition", c(3:5, 7:9, 12, 16:17), ".rev")

# Reverse-code
for (i in 1:length(need_cog_rev)) {
  data_Q_total[, need_cog_rev[i]] <- reverse.code(keys = c(-1),
                                                  items = data_Q_total[, need_cog_rev[i]],
                                                  mini = c(1),
                                                  maxi = c(6))
}

# Describe: item-level
par(mfrow = c(3, 6))
for (i in 1:length(need_cog)) {
  hist(data_Q_total[, need_cog[i]],
       main = colnames(data_Q_total[need_cog[i]]),
       breaks = seq(0, 6, 1),
       col = sample(colors(), 1),
       xlab = ""
       )
```

```
}
```




```
describe(data_Q_total[, need_cog])
```

```
##                 vars   n mean   sd median trimmed  mad min max range  skew
## cognition1         1 155 4.24 1.15      4    4.28 1.48   1   6     5 -0.34
## cognition2         2 155 4.07 1.26      4    4.13 1.48   1   6     5 -0.52
## cognition6         3 155 4.50 1.09      5    4.58 1.48   1   6     5 -0.67
## cognition10        4 155 4.15 1.38      4    4.23 1.48   1   6     5 -0.37
## cognition11        5 155 4.68 1.08      5    4.78 1.48   1   6     5 -0.71
## cognition13        6 155 3.47 1.21      3    3.46 1.48   1   6     5  0.08
## cognition14        7 155 4.44 1.31      5    4.58 1.48   1   6     5 -0.75
## cognition15        8 155 4.10 1.24      4    4.16 1.48   1   6     5 -0.38
## cognition18        9 155 4.96 1.02      5    5.08 1.48   1   6     5 -0.91
## cognition3.rev    10 155 5.31 1.02      6    5.50 0.00   1   6     5 -2.09
## cognition4.rev    11 155 4.86 1.01      5    4.98 1.48   1   6     5 -0.81
## cognition5.rev    12 155 5.17 0.93      5    5.29 1.48   2   6     4 -1.01
## cognition7.rev    13 155 4.69 1.05      5    4.81 1.48   1   6     5 -0.95
## cognition8.rev    14 155 4.16 1.24      4    4.23 1.48   1   6     5 -0.47
## cognition9.rev    15 155 4.45 1.20      5    4.50 1.48   1   6     5 -0.33
## cognition12.rev   16 155 5.03 0.93      5    5.14 1.48   1   6     5 -1.47
## cognition16.rev   17 155 3.83 1.36      4    3.91 1.48   1   6     5 -0.52
## cognition17.rev   18 155 4.94 1.03      5    5.07 1.48   1   6     5 -1.01
##                 kurtosis   se
## cognition1         -0.56 0.09
```

```
## cognition2         -0.34 0.10
## cognition6          0.46 0.09
## cognition10        -0.75 0.11
## cognition11         0.45 0.09
## cognition13        -0.25 0.10
## cognition14        -0.09 0.11
## cognition15        -0.39 0.10
## cognition18         0.76 0.08
## cognition3.rev      5.24 0.08
## cognition4.rev      0.60 0.08
## cognition5.rev      0.56 0.07
## cognition7.rev      1.22 0.08
## cognition8.rev     -0.27 0.10
## cognition9.rev     -0.84 0.10
## cognition12.rev     3.70 0.07
## cognition16.rev    -0.49 0.11
## cognition17.rev     0.97 0.08
```

```
# Cronbach's alphas
psych::alpha(data_Q_total[, need_cog])
```

```
##
## Reliability analysis
## Call: psych::alpha(x = data_Q_total[, need_cog])
##
##   raw_alpha std.alpha G6(smc) average_r S/N   ase mean   sd median_r
##       0.88      0.89    0.91       0.3 7.9 0.013  4.5 0.66      0.3
##
##  lower alpha upper     95% confidence boundaries
## 0.86 0.88 0.91
##
##  Reliability if an item is dropped:
##                 raw_alpha std.alpha G6(smc) average_r S/N alpha se var.r med.r
## cognition1           0.87      0.88    0.90      0.30 7.2    0.014 0.017  0.30
## cognition2           0.87      0.87    0.90      0.29 7.0    0.014 0.015  0.30
## cognition6           0.87      0.88    0.90      0.30 7.3    0.014 0.018  0.30
## cognition10          0.89      0.89    0.91      0.32 8.0    0.012 0.016  0.32
## cognition11          0.87      0.88    0.90      0.30 7.2    0.014 0.015  0.30
## cognition13          0.87      0.88    0.90      0.30 7.2    0.014 0.017  0.30
## cognition14          0.87      0.88    0.90      0.30 7.2    0.014 0.017  0.30
## cognition15          0.87      0.88    0.90      0.29 7.1    0.014 0.016  0.29
## cognition18          0.88      0.88    0.91      0.31 7.6    0.013 0.018  0.30
## cognition3.rev       0.88      0.88    0.90      0.31 7.5    0.013 0.018  0.30
## cognition4.rev       0.87      0.88    0.90      0.30 7.2    0.014 0.017  0.29
## cognition5.rev       0.87      0.88    0.90      0.30 7.2    0.014 0.017  0.29
## cognition7.rev       0.88      0.89    0.91      0.32 7.9    0.013 0.016  0.31
## cognition8.rev       0.88      0.88    0.91      0.31 7.6    0.013 0.018  0.30
## cognition9.rev       0.88      0.88    0.91      0.31 7.6    0.013 0.018  0.30
## cognition12.rev      0.88      0.88    0.90      0.30 7.3    0.014 0.017  0.30
## cognition16.rev      0.89      0.89    0.91      0.32 8.0    0.012 0.016  0.32
## cognition17.rev      0.88      0.88    0.91      0.31 7.7    0.013 0.018  0.31
##
##  Item statistics
##                    n raw.r std.r r.cor r.drop mean   sd
```

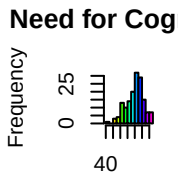
```
## cognition1      155  0.66  0.66  0.65   0.60  4.2 1.15
## cognition2      155  0.76  0.75  0.75   0.71  4.1 1.26
## cognition6      155  0.63  0.64  0.63   0.57  4.5 1.09
## cognition10     155  0.39  0.37  0.31   0.29  4.2 1.38
## cognition11     155  0.66  0.66  0.65   0.61  4.7 1.08
## cognition13     155  0.66  0.65  0.63   0.59  3.5 1.21
## cognition14     155  0.69  0.68  0.66   0.62  4.4 1.31
## cognition15     155  0.71  0.70  0.69   0.66  4.1 1.24
## cognition18     155  0.51  0.53  0.49   0.45  5.0 1.02
## cognition3.rev  155  0.55  0.56  0.53   0.49  5.3 1.02
## cognition4.rev  155  0.65  0.66  0.64   0.60  4.9 1.01
## cognition5.rev  155  0.66  0.68  0.66   0.61  5.2 0.93
## cognition7.rev  155  0.41  0.42  0.39   0.34  4.7 1.05
## cognition8.rev  155  0.53  0.53  0.49   0.45  4.2 1.24
## cognition9.rev  155  0.54  0.54  0.50   0.47  4.5 1.20
## cognition12.rev 155  0.62  0.64  0.61   0.57  5.0 0.93
## cognition16.rev 155  0.40  0.38  0.32   0.29  3.8 1.36
## cognition17.rev 155  0.48  0.49  0.44   0.41  4.9 1.03
##
## Non missing response frequency for each item
##                    1    2    3    4    5    6 miss
## cognition1      0.01 0.07 0.19 0.28 0.33 0.13 0.11
## cognition2      0.03 0.11 0.13 0.32 0.30 0.11 0.11
## cognition6      0.01 0.04 0.09 0.33 0.35 0.18 0.11
## cognition10     0.03 0.10 0.18 0.24 0.25 0.19 0.11
## cognition11     0.01 0.01 0.10 0.27 0.35 0.25 0.11
## cognition13     0.06 0.12 0.35 0.28 0.13 0.06 0.11
## cognition14     0.03 0.06 0.12 0.22 0.34 0.23 0.11
## cognition15     0.03 0.08 0.18 0.31 0.27 0.13 0.11
## cognition18     0.01 0.01 0.06 0.22 0.34 0.36 0.11
## cognition3.rev  0.02 0.01 0.01 0.10 0.30 0.55 0.11
## cognition4.rev  0.01 0.01 0.07 0.23 0.37 0.30 0.11
## cognition5.rev  0.00 0.01 0.04 0.17 0.33 0.45 0.11
## cognition7.rev  0.01 0.03 0.05 0.28 0.41 0.22 0.11
## cognition8.rev  0.03 0.08 0.15 0.32 0.28 0.14 0.11
## cognition9.rev  0.01 0.03 0.23 0.20 0.31 0.23 0.11
## cognition12.rev 0.01 0.01 0.03 0.15 0.49 0.31 0.11
## cognition16.rev 0.08 0.11 0.14 0.32 0.27 0.08 0.11
## cognition17.rev 0.01 0.02 0.06 0.19 0.39 0.34 0.11
```

```r
# Need for Cognition Scoring
# https://centerofinquiry.org/uncategorized/need-for-cognition-scale-wabash-national-study/
# According to this source you should calculate the sum score for this scale

# Calculate sum score
data_Q_total$CogSum <- rowSums(cbind(data_Q_total$cognition1, data_Q_total$cognition2, data_Q_total$cogn

# Replace sum score 0s with NAs
data_Q_total$CogSum[which(data_Q_total$CogSum == 0)] <- NA

# Describe: sum-score level
describe(data_Q_total$CogSum)
```

```
##    vars   n  mean    sd median trimmed   mad min max range  skew kurtosis   se
```

```
## X1    1 155 81.06 11.95     84   81.48 10.38  44 105    61 -0.43    -0.15 0.96
```

```
hist(data_Q_total$CogSum,
     main = "Need for Cognition",
     col = rainbow(14),
     xlab = "")
```



Actively openminded thinking

```
# Actively Openminded Thinking
aot <- c(paste0("openReverse", 1:4), paste0("openNormal", 1:3))
aot_rev <- paste0("openReverse", 1:4)

# Reverse-code
for (i in 1:length(aot_rev)) {
  data_Q_total[, aot_rev[i]] <- reverse.code(keys = c(-1),
                                             items = data_Q_total[, aot_rev[i]],
                                             mini = c(1),
                                             maxi = c(7))
}

# Describe: item-level
par(mfrow = c(2, 4))
for (i in 1:length(aot)) {
```

```
  hist(data_Q_total[, aot[i]],
       main = colnames(data_Q_total[aot[i]]),
       col = sample(colors(), 1),
       breaks = seq(0, 7, 1),
       xlab = "")
}
describe(data_Q_total[, aot])
```

```
##              vars   n mean   sd median trimmed  mad min max range  skew
## openReverse1    1 149 6.74 0.48      7    6.82 0.00   5   7     2 -1.59
## openReverse2    2 149 5.23 1.24      6    5.30 1.48   2   7     5 -0.52
## openReverse3    3 149 5.85 1.13      6    5.98 1.48   3   7     4 -0.72
## openReverse4    4 149 6.45 0.95      7    6.65 0.00   1   7     6 -2.51
## openNormal1     5 149 5.98 1.06      6    6.13 1.48   2   7     5 -1.14
## openNormal2     6 149 6.34 0.79      7    6.44 0.00   4   7     3 -0.91
## openNormal3     7 149 5.58 1.26      6    5.71 1.48   1   7     6 -0.91
##              kurtosis   se
## openReverse1     1.58 0.04
## openReverse2    -0.60 0.10
## openReverse3    -0.41 0.09
## openReverse4     8.32 0.08
## openNormal1      1.40 0.09
## openNormal2     -0.03 0.06
## openNormal3      0.75 0.10
```

```
# Cronbach's alpha
psych::alpha(data_Q_total[, aot])
```

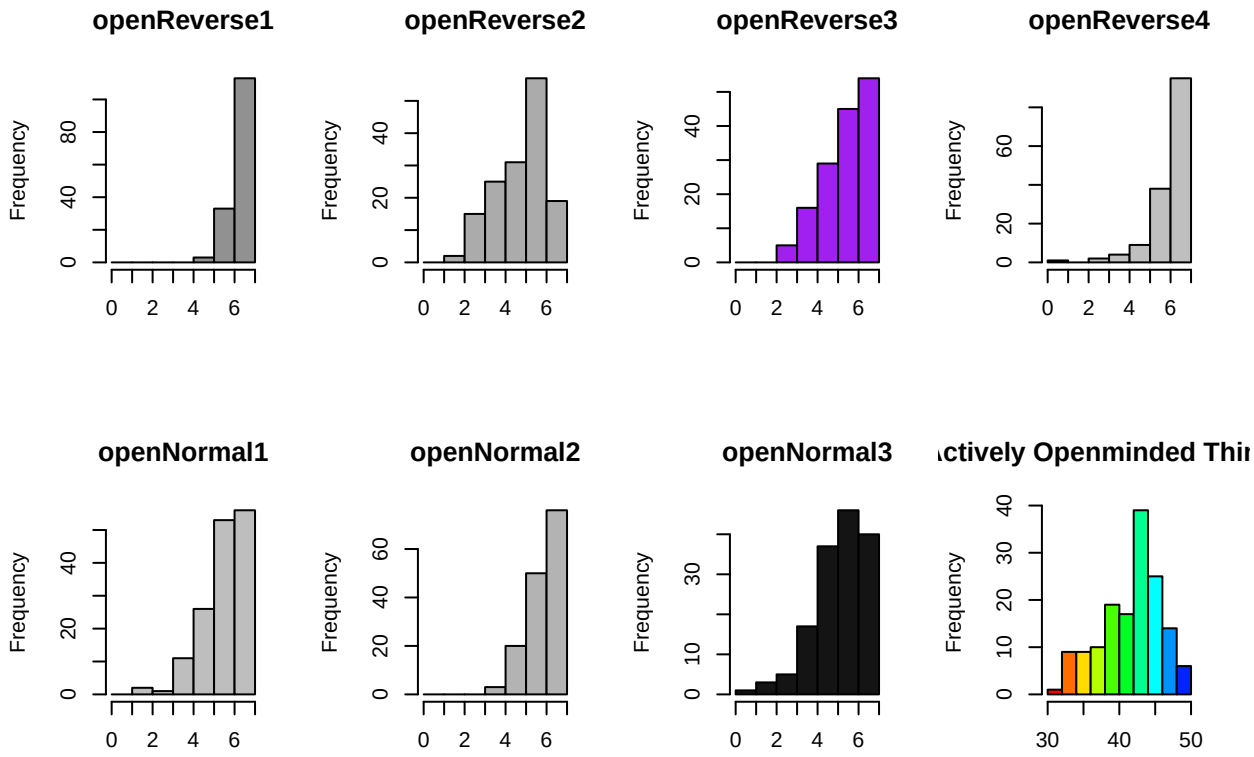
```
##
## Reliability analysis
## Call: psych::alpha(x = data_Q_total[, aot])
##
##   raw_alpha std.alpha G6(smc) average_r S/N   ase mean  sd median_r
##       0.69       0.7     0.7      0.25 2.4 0.035    6 0.6     0.29
##
##  lower alpha upper     95% confidence boundaries
## 0.62 0.69 0.76
##
##  Reliability if an item is dropped:
##              raw_alpha std.alpha G6(smc) average_r S/N alpha se var.r med.r
## openReverse1      0.69      0.71    0.69      0.29 2.4    0.037 0.014  0.30
## openReverse2      0.70      0.71    0.70      0.29 2.5    0.033 0.012  0.30
## openReverse3      0.62      0.65    0.64      0.23 1.8    0.044 0.018  0.27
## openReverse4      0.63      0.65    0.64      0.23 1.8    0.042 0.019  0.25
## openNormal1       0.63      0.66    0.64      0.24 1.9    0.042 0.012  0.27
## openNormal2       0.63      0.64    0.63      0.23 1.8    0.041 0.017  0.25
## openNormal3       0.66      0.68    0.67      0.26 2.1    0.038 0.014  0.29
##
##  Item statistics
##                 n raw.r std.r r.cor r.drop mean   sd
## openReverse1 149  0.35  0.48  0.33   0.25  6.7 0.48
## openReverse2 149  0.52  0.47  0.31   0.25  5.2 1.24
## openReverse3 149  0.70  0.67  0.60   0.51  5.9 1.13
```

```
## openReverse4 149  0.66  0.67  0.60   0.50  6.4 0.95
## openNormal1  149  0.66  0.64  0.58   0.48  6.0 1.06
## openNormal2  149  0.64  0.69  0.63   0.50  6.3 0.79
## openNormal3  149  0.62  0.57  0.47   0.38  5.6 1.26
##
## Non missing response frequency for each item
##                 1    2    3    4    5    6    7 miss
## openReverse1 0.00 0.00 0.00 0.00 0.02 0.22 0.76 0.14
## openReverse2 0.00 0.01 0.10 0.17 0.21 0.38 0.13 0.14
## openReverse3 0.00 0.00 0.03 0.11 0.19 0.30 0.36 0.14
## openReverse4 0.01 0.00 0.01 0.03 0.06 0.26 0.64 0.14
## openNormal1  0.00 0.01 0.01 0.07 0.17 0.36 0.38 0.14
## openNormal2  0.00 0.00 0.00 0.02 0.13 0.34 0.51 0.14
## openNormal3  0.01 0.02 0.03 0.11 0.25 0.31 0.27 0.14
```

```
# Actively Openminded Thinking Scoring
# https://www.sciencedirect.com/science/article/pii/S1871187119303700
# Again, this article says to do a sum score

# Calculate sum score
data_Q_total$AOTSum <- rowSums(cbind(data_Q_total$openNormal1, data_Q_total$openNormal2, data_Q_total$o

# Replace sum score 0s with NAs
data_Q_total$AOTSum[which(data_Q_total$AOTSum == 0)] <- NA

# Describe: sum-score level
describe(data_Q_total$AOTSum)
```

```
##    vars   n  mean   sd median trimmed  mad min max range  skew kurtosis   se
## X1    1 149 42.16 4.21     43   42.43 4.45  31  49    18 -0.57    -0.37 0.35
```

```
hist(data_Q_total$AOTSum,
     main = "Actively Openminded Thinking",
     col = rainbow(14),
     xlab = "")
```

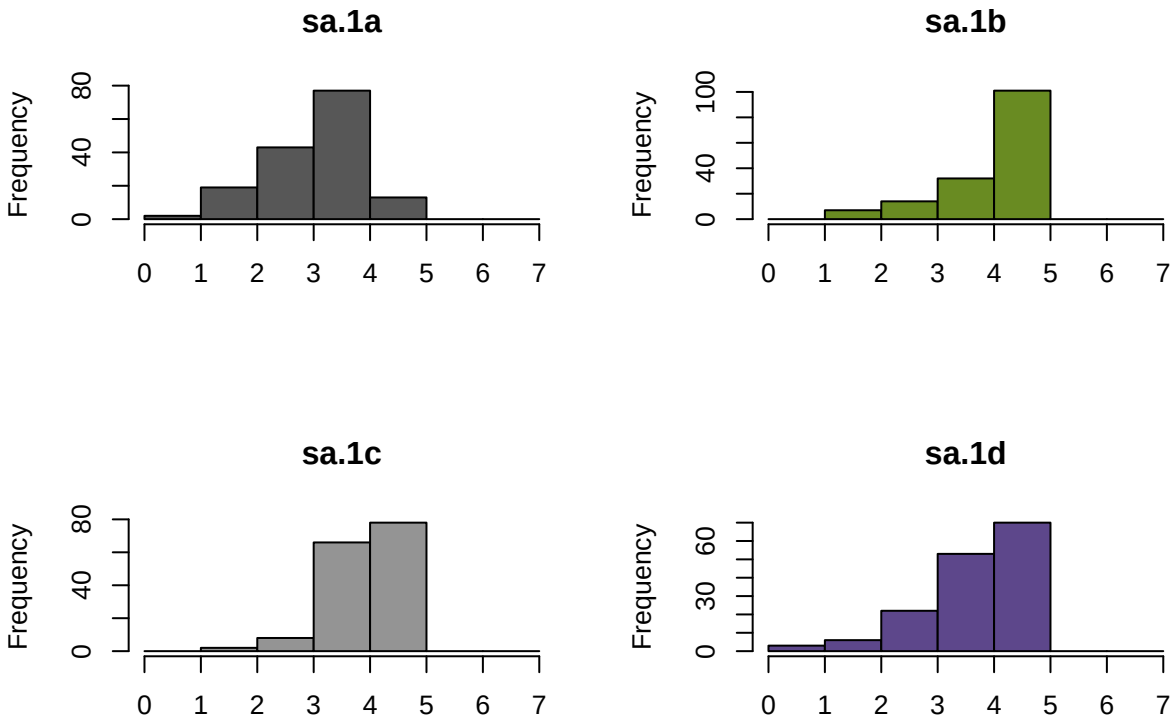


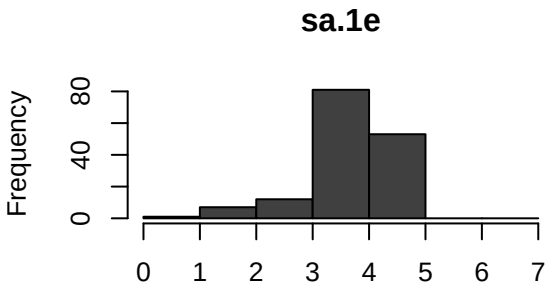
Science Attitudes Questionnaire

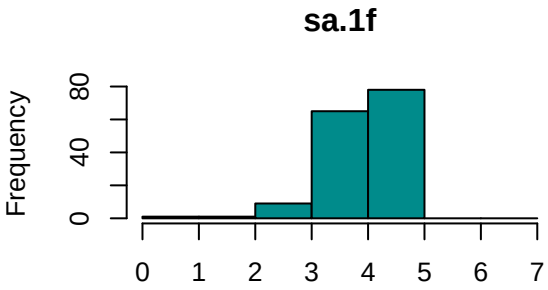
This section has 12 questions with Likert responses that represent a (possible) 3 factor model based on the sources of the questions, from an early version of the Science Capital Scale from the FINSCI population survey study. The first 8 questions originally come from Archer, 2015, and the last 4 are modified from the Trust in Science and Scientists Scale (Nadelson et al., 2014).

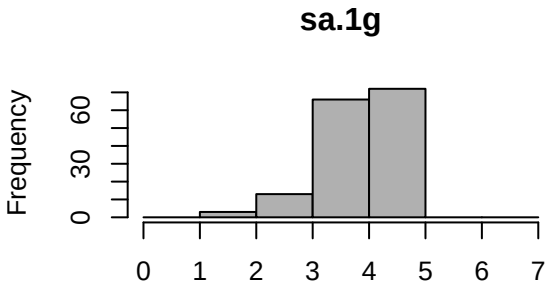
```
# Science Attitudes
sci_att <- c(paste0("sa.1", letters[1:8]), paste0("sa.2", letters[1:4]))
sci_att_rev <- c("sa.1b", "sa.1d", "sa.2a", "sa.2b", "sa.2d")

# Reverse-code
for (i in 1:length(sci_att_rev)) {
  data_Q_total[, sci_att_rev[i]] <- reverse.code(keys = c(-1),
                                                 items = data_Q_total[, sci_att_rev[i]],
                                                 mini = c(1),
                                                 maxi = c(5))
}

# Describe: item-level
par(mfrow = c(2, 2))
for (i in 1:length(sci_att)) {
  hist(data_Q_total[, sci_att[i]],
       main = colnames(data_Q_total[sci_att[i]]),
       col = sample(colors(), 1),
```

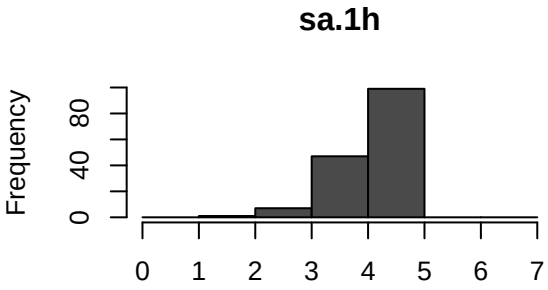
```
        breaks = seq(0, 7, 1),
        xlab = "")
}
```

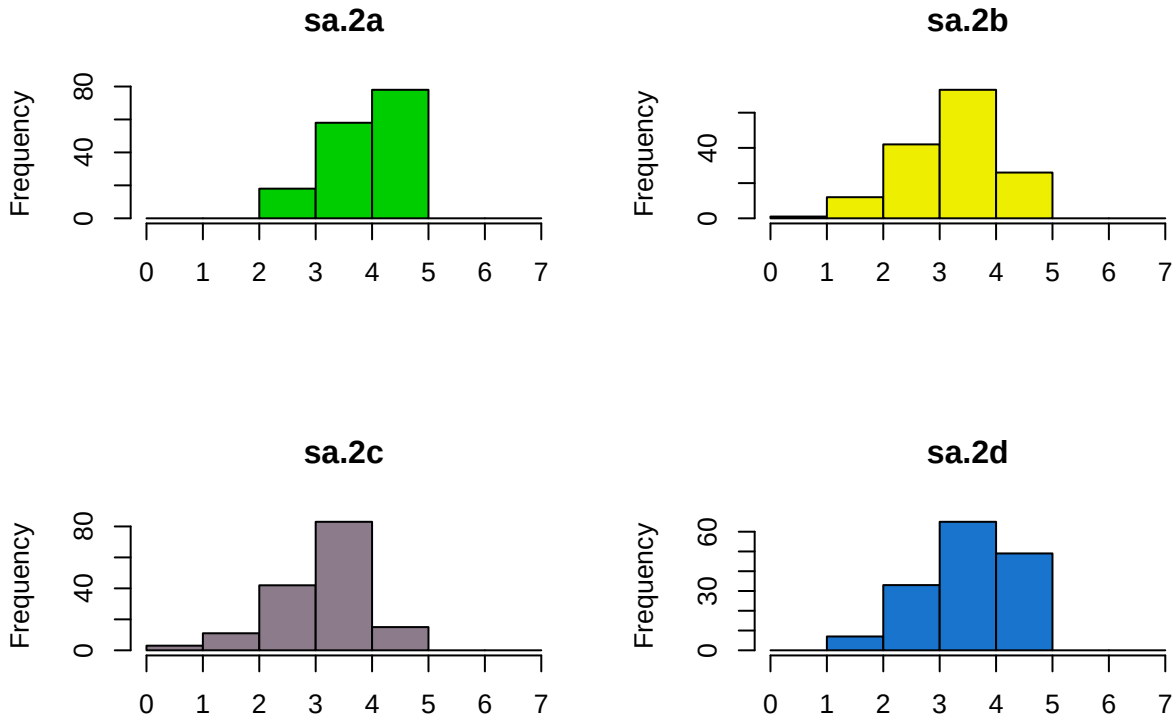



sa.1e
Frequency
0
40
80
0 1 2 3 4 5 6 7


sa.1f
Frequency
0
40
80
0 1 2 3 4 5 6 7


sa.1g
Frequency
0
30
60
0 1 2 3 4 5 6 7


sa.1h
Frequency
0
40
80
0 1 2 3 4 5 6 7




```
describe(data_Q_total[, sci_att])
```

```
##       vars   n mean   sd median trimmed  mad min max range  skew kurtosis   se
## sa.1a    1 154 3.52 0.86      4    3.56 0.74   1   5     4 -0.54    -0.08 0.07
## sa.1b    2 154 4.47 0.84      5    4.65 0.00   2   5     3 -1.52     1.37 0.07
## sa.1c    3 154 4.43 0.66      5    4.51 0.00   2   5     3 -0.98     0.99 0.05
## sa.1d    4 154 4.18 0.95      4    4.31 1.48   1   5     4 -1.17     1.08 0.08
## sa.1e    5 154 4.16 0.80      4    4.27 0.00   1   5     4 -1.12     1.70 0.06
## sa.1f    6 154 4.42 0.69      5    4.51 0.00   1   5     4 -1.34     3.13 0.06
## sa.1g    7 154 4.34 0.72      4    4.45 1.48   2   5     3 -0.92     0.61 0.06
## sa.1h    8 154 4.58 0.61      5    4.68 0.00   2   5     3 -1.34     1.48 0.05
## sa.2a    9 154 4.39 0.69      5    4.48 0.00   3   5     2 -0.67    -0.71 0.06
## sa.2b   10 154 3.72 0.86      4    3.77 1.48   1   5     4 -0.42    -0.13 0.07
## sa.2c   11 154 3.62 0.83      4    3.67 0.00   1   5     4 -0.76     0.78 0.07
## sa.2d   12 154 4.01 0.85      4    4.07 1.48   2   5     3 -0.47    -0.53 0.07
```

```
# Cronbach's alphas
psych::alpha(data_Q_total[, sci_att])
```

```
##
## Reliability analysis
## Call: psych::alpha(x = data_Q_total[, sci_att])
##
##   raw_alpha std.alpha G6(smc) average_r S/N   ase mean   sd median_r
##       0.76      0.77    0.81      0.22 3.4 0.026  4.2 0.41     0.22
```

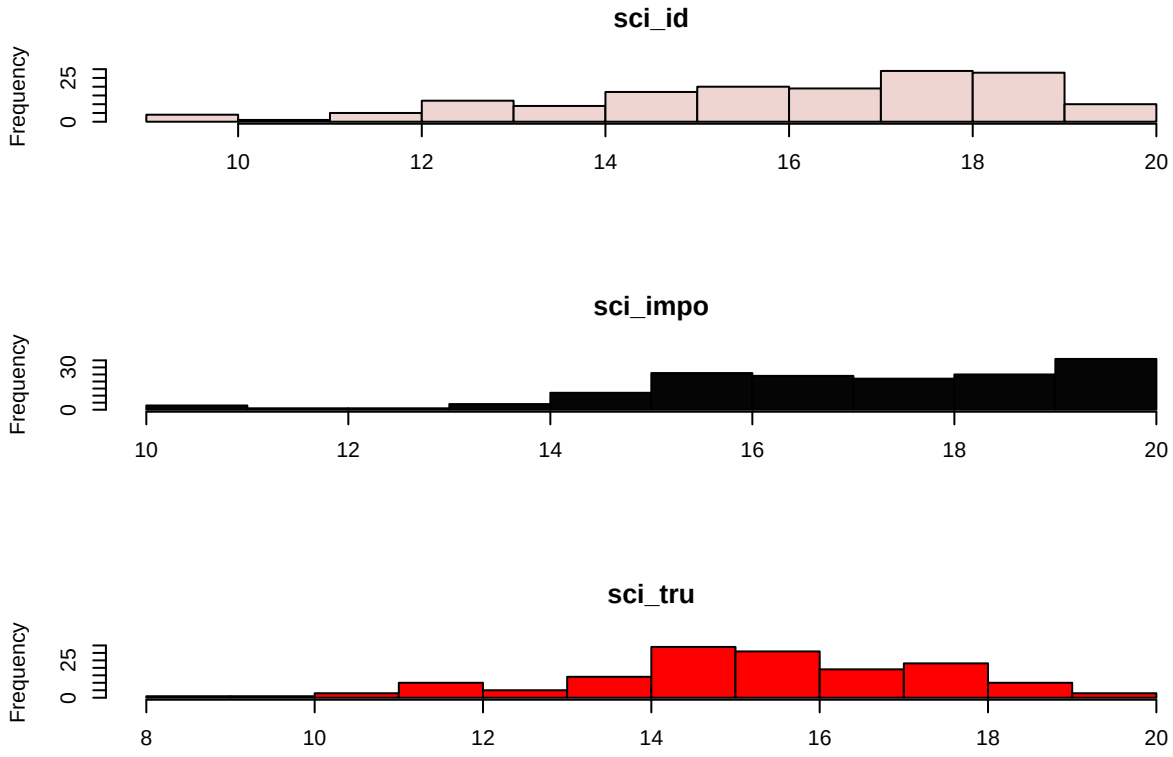
```
##
##  lower alpha upper     95% confidence boundaries
## 0.71 0.76 0.82
##
##  Reliability if an item is dropped:
##       raw_alpha std.alpha G6(smc) average_r S/N alpha se var.r med.r
## sa.1a      0.74      0.75    0.78      0.22 3.0    0.029 0.020  0.22
## sa.1b      0.73      0.74    0.77      0.21 2.9    0.031 0.018  0.18
## sa.1c      0.75      0.76    0.79      0.23 3.2    0.028 0.021  0.22
## sa.1d      0.76      0.77    0.80      0.23 3.4    0.027 0.019  0.23
## sa.1e      0.74      0.75    0.78      0.22 3.1    0.029 0.019  0.22
## sa.1f      0.72      0.73    0.77      0.20 2.7    0.031 0.018  0.17
## sa.1g      0.75      0.76    0.79      0.23 3.2    0.028 0.019  0.22
## sa.1h      0.74      0.74    0.78      0.21 2.9    0.030 0.020  0.18
## sa.2a      0.75      0.76    0.80      0.23 3.2    0.028 0.022  0.22
## sa.2b      0.76      0.77    0.80      0.24 3.4    0.026 0.019  0.23
## sa.2c      0.77      0.78    0.81      0.24 3.5    0.026 0.020  0.25
## sa.2d      0.75      0.76    0.79      0.23 3.2    0.028 0.021  0.22
##
##  Item statistics
##         n raw.r std.r r.cor r.drop mean   sd
## sa.1a 154  0.60  0.58  0.54   0.47  3.5 0.86
## sa.1b 154  0.68  0.67  0.65   0.57  4.5 0.84
## sa.1c 154  0.48  0.51  0.44   0.37  4.4 0.66
## sa.1d 154  0.47  0.44  0.36   0.30  4.2 0.95
## sa.1e 154  0.57  0.58  0.54   0.44  4.2 0.80
## sa.1f 154  0.71  0.73  0.72   0.62  4.4 0.69
## sa.1g 154  0.50  0.51  0.45   0.38  4.3 0.72
## sa.1h 154  0.61  0.65  0.62   0.53  4.6 0.61
## sa.2a 154  0.48  0.50  0.42   0.36  4.4 0.69
## sa.2b 154  0.43  0.42  0.34   0.27  3.7 0.86
## sa.2c 154  0.37  0.36  0.25   0.21  3.6 0.83
## sa.2d 154  0.52  0.50  0.44   0.38  4.0 0.85
##
## Non missing response frequency for each item
##          1    2    3    4    5 miss
## sa.1a 0.01 0.12 0.28 0.50 0.08 0.11
## sa.1b 0.00 0.05 0.09 0.21 0.66 0.11
## sa.1c 0.00 0.01 0.05 0.43 0.51 0.11
## sa.1d 0.02 0.04 0.14 0.34 0.45 0.11
## sa.1e 0.01 0.05 0.08 0.53 0.34 0.11
## sa.1f 0.01 0.01 0.06 0.42 0.51 0.11
## sa.1g 0.00 0.02 0.08 0.43 0.47 0.11
## sa.1h 0.00 0.01 0.05 0.31 0.64 0.11
## sa.2a 0.00 0.00 0.12 0.38 0.51 0.11
## sa.2b 0.01 0.08 0.27 0.47 0.17 0.11
## sa.2c 0.02 0.07 0.27 0.54 0.10 0.11
## sa.2d 0.00 0.05 0.21 0.42 0.32 0.11
```

```
# Calculate sum score
data_Q_total$sci_id <- rowSums(cbind(data_Q_total$sa.1a, data_Q_total$sa.1b, data_Q_total$sa.1d, data_Q
data_Q_total$sci_impo <- rowSums(cbind(data_Q_total$sa.1c, data_Q_total$sa.1e, data_Q_total$sa.1h, data
data_Q_total$sci_tru <- rowSums(cbind(data_Q_total$sa.2a, data_Q_total$sa.2b, data_Q_total$sa.2c, data_
```

```r
# Replace total score zeroes with NA
data_Q_total$sci_id[which(data_Q_total$sci_id == 0)] <- NA
data_Q_total$sci_impo[which(data_Q_total$sci_impo == 0)] <- NA
data_Q_total$sci_tru[which(data_Q_total$sci_tru == 0)] <- NA

# Describe: sum-score level
sci_att_sum <- c("sci_id", "sci_impo", "sci_tru")
describe(data_Q_total[, sci_att_sum])
```

```
##          vars   n  mean   sd median trimmed  mad min max range  skew kurtosis
## sci_id      1 154 16.51 2.48     17   16.73 2.97   9  20    11 -0.76     0.07
## sci_impo    2 154 17.58 2.09     18   17.78 2.97  10  20    10 -0.86     0.84
## sci_tru     3 154 15.75 2.15     16   15.87 1.48   8  20    12 -0.53     0.43
##            se
## sci_id   0.20
## sci_impo 0.17
## sci_tru  0.17
```

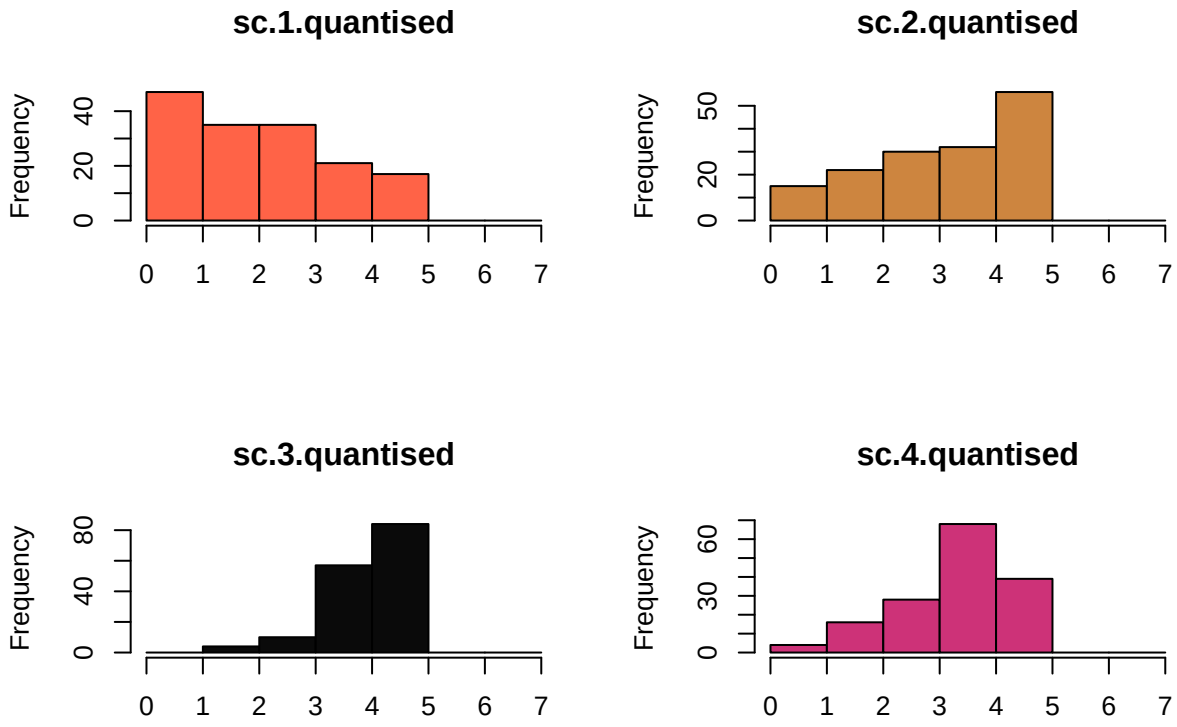
```r
par(mfrow = c(3, 1))
for (i in 1:length(sci_att_sum)) {
  hist(data_Q_total[, sci_att_sum[i]],
       main = colnames(data_Q_total[sci_att_sum[i]]),
       col = sample(colors(), 1),
       xlab = "")
}
```



Science Curiosity

This is a 4-item scale, scored as a single sum score from the quantised responses. No reverse scoring. Modified from Landrum et al., 2016 and Motta et al., 2019

```
# Science Curiosity
sci_cur <- paste0("sc.", 1:4, ".quantised")

# Describe: item-level
par(mfrow = c(2, 2))
for (i in 1:length(sci_cur)) {
  hist(data_Q_total[, sci_cur[i]],
       main = colnames(data_Q_total[sci_cur[i]]),
       col = sample(colors(), 1),
       breaks = seq(0, 7, 1),
       xlab = "")
}
```




```
describe(data_Q_total[, sci_cur])
```

```
##                vars   n mean   sd median trimmed  mad min max range  skew
## sc.1.quantised    1 155 2.52 1.34      2    2.41 1.48   1   5     4  0.43
## sc.2.quantised    2 155 3.59 1.36      4    3.74 1.48   1   5     4 -0.51
## sc.3.quantised    3 155 4.43 0.73      5    4.55 0.00   2   5     3 -1.24
## sc.4.quantised    4 155 3.79 1.02      4    3.89 1.48   1   5     4 -0.74
```

```
##                kurtosis   se
## sc.1.quantised    -1.01 0.11
## sc.2.quantised    -1.01 0.11
## sc.3.quantised     1.35 0.06
## sc.4.quantised    -0.01 0.08
```

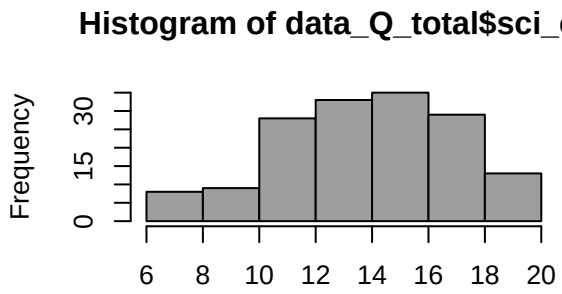
```
# Cronbach's alpha
psych::alpha(data_Q_total[, sci_cur])
```

```
##
## Reliability analysis
## Call: psych::alpha(x = data_Q_total[, sci_cur])
##
##   raw_alpha std.alpha G6(smc) average_r S/N   ase mean   sd median_r
##       0.62      0.66    0.63      0.33   2 0.045  3.6 0.78     0.34
##
##  lower alpha upper     95% confidence boundaries
## 0.54 0.62 0.71
##
##  Reliability if an item is dropped:
##                raw_alpha std.alpha G6(smc) average_r S/N alpha se   var.r med.r
## sc.1.quantised      0.52      0.60    0.55      0.33 1.5    0.063 0.03739  0.37
## sc.2.quantised      0.59      0.64    0.56      0.38 1.8    0.052 0.01292  0.32
## sc.3.quantised      0.52      0.52    0.44      0.26 1.1    0.061 0.01518  0.31
## sc.4.quantised      0.58      0.62    0.52      0.35 1.6    0.050 0.00077  0.36
##
##  Item statistics
##                  n raw.r std.r r.cor r.drop mean   sd
## sc.1.quantised 155  0.76  0.70  0.53   0.45  2.5 1.34
## sc.2.quantised 155  0.71  0.66  0.48   0.37  3.6 1.36
## sc.3.quantised 155  0.69  0.78  0.69   0.54  4.4 0.73
## sc.4.quantised 155  0.63  0.69  0.55   0.36  3.8 1.02
##
## Non missing response frequency for each item
##                    1    2    3    4    5 miss
## sc.1.quantised 0.30 0.23 0.23 0.14 0.11 0.11
## sc.2.quantised 0.10 0.14 0.19 0.21 0.36 0.11
## sc.3.quantised 0.00 0.03 0.06 0.37 0.54 0.11
## sc.4.quantised 0.03 0.10 0.18 0.44 0.25 0.11
```

```
# Calculate sum score
data_Q_total$sci_cur <- rowSums(cbind(data_Q_total$sc.1.quantised, data_Q_total$sc.2.quantised, data_Q_

# Replace sum score 0s with NAs
data_Q_total$sci_cur[which(data_Q_total$sci_cur == 0)] <- NA

# Describe: sum-score level
describe(data_Q_total$sci_cur)
```

```
##    vars   n  mean   sd median trimmed  mad min max range  skew kurtosis   se
## X1    1 155 14.33 3.13     14   14.44 2.97   7  20    13 -0.23    -0.59 0.25
```

```
hist(data_Q_total$sci_cur,
     col = sample(colors(), 1),
     xlab = "")
```



Heuristic Reasoning

```
# Q1
data_Q_total$hr.1 <- as.factor(data_Q_total$hr.1)
levels(data_Q_total$hr.1) <- list("N" = "Kumpikin on yhtÃ¤ todennÃ¤kÃ¶istÃ¤", "H-R" = "Kuudes lapsi on

# Q2
data_Q_total$hr.2 <- as.factor(data_Q_total$hr.2)
levels(data_Q_total$hr.2) <- list("N" = "Kummatkin sarjat ovat yhtÃ¤ todennÃ¤kÃ¶isiÃ¤", "H-R" = "THHTHT

# Q3
# 3.1
data_Q_total$hr.3.1 <- as.factor(data_Q_total$hr.3.1)
levels(data_Q_total$hr.3.1) <- list("b" = "Herra F. sai sydÃ¤nkohtauksen", "a" = "Herra F. on yli 55-vu
# 3.2
data_Q_total$hr.3.2 <- as.factor(data_Q_total$hr.3.2)
levels(data_Q_total$hr.3.2) <- list("b" = "Herra F. sai sydÃ¤nkohtauksen", "a" = "Herra F. on yli 55-vu
# 3.3
data_Q_total$hr.3.3 <- as.factor(data_Q_total$hr.3.3)
```

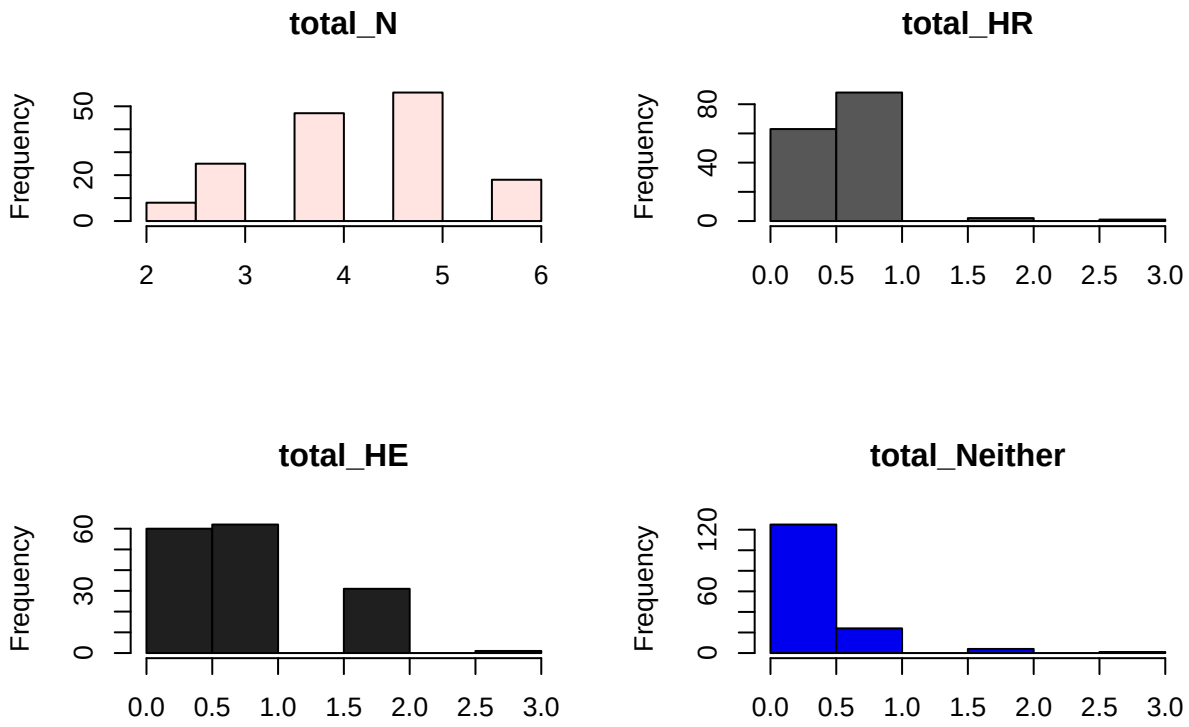
```
levels(data_Q_total$hr.3.3) <- list("b" = "Herra F. sai sydÃ¤nkohtauksen", "a" = "Herra F. on yli 55-vu
# Create the variables for the combinations of answers
# abc / acb / cab = N; bac / bca /cba = H-R
data_Q_total <- data_Q_total %>%
  mutate(hr.3.updated = case_when(
  ((hr.3.1 == "b" & (hr.3.2 == "a" | hr.3.3 == "a")) | hr.3.2 == "b" & hr.3.3 == "a") ~ "H-R",
  ((hr.3.1 == "a" & (hr.3.2 == "b" | hr.3.3 == "b")) | (hr.3.2 == "a" & hr.3.3 == "b")) ~ "N"
 ))
data_Q_total$hr.3.updated <- factor(data_Q_total$hr.3.updated)

# Q4
data_Q_total$hr.4 <- as.factor(data_Q_total$hr.4)
levels(data_Q_total$hr.4) <- list("H-E" = "Kumpikin on yhtÃ¤ tehokas", "N" = "Kognitiivis-behavioraalis
# Kognitiivis-behavioraalista terapiaa = N; Yhta tehokas = H-E

# Q5
data_Q_total$hr.5 <- as.factor(data_Q_total$hr.5)
levels(data_Q_total$hr.5) <- list("N" = "Huomenna luultavasti sataa", "H-E" = "On mahdotonta sanoa sataa
# Huomenna luultavasti sataa = N; On mahdotonta sanoa sataako huomenna vai ei = H-E

# Q6
data_Q_total$hr.6 <- as.factor(data_Q_total$hr.6)
levels(data_Q_total$hr.6) <- list("N" = "TyttÃ¶", "H-E" = "Kumpikin on yhtÃ¤ todennÃ¤kÃ¶istÃ¤", "Neithe
# Tytto = N; Kumpikin on yht? todenn?k?ist? = H-E

# Calculate sums
data_Q_total <- data_Q_total %>%
  mutate(
    total_N = apply(., 1, function(x) length(which(x == "N"))),
    total_HR = apply(., 1, function(x) length(which(x == "H-R"))),
    total_HE = apply(., 1, function(x) length(which(x == "H-E"))),
    total_Neither = apply(., 1, function(x) length(which(x == "Neither")))
 )

# Omit participants with 0s in all the sum columns
heur <- c("total_N", "total_HR", "total_HE", "total_Neither")
data_Q_total[which(data_Q_total$total_N == 0 & data_Q_total$total_HR == 0 & data_Q_total$total_HE == 0 &

# Check that sum scores add up to 6
sum_heur <- data_Q_total %>%
  select(total_N, total_HR, total_HE, total_Neither) %>%
  rowwise() %>%
  mutate(sum_heur = total_N + total_HR + total_HE + total_Neither) %>%
  select(sum_heur)
table(sum_heur$sum_heur)
```

```
##
##   6
## 154
```

```
# Describe: sum score level
describe(data_Q_total[, heur])
```

```
##                vars   n mean   sd median trimmed  mad min max range  skew
## total_N           1 154 4.33 1.05      4    4.35 1.48   2   6     4 -0.35
## total_HR          2 154 0.62 0.55      1    0.61 0.00   0   3     3  0.35
## total_HE          3 154 0.82 0.77      1    0.77 1.48   0   3     3  0.39
## total_Neither     4 154 0.23 0.52      0    0.11 0.00   0   3     3  2.49
##                kurtosis   se
## total_N           -0.49 0.08
## total_HR           0.57 0.04
## total_HE          -0.96 0.06
## total_Neither      6.70 0.04
```

```
par(mfrow = c(2, 2))
for (i in 1:length(heur)) {
  hist(data_Q_total[, heur[i]],
       main = colnames(data_Q_total[heur[i]]),
       col = sample(colors(), 1),
       xlab = "")
}
```




```
# Create binary heuristic scores (0=0, 1=everything else)
data_Q_total <- data_Q_total %>%
  mutate(HEscore = case_when(total_HE == 0 ~ 0,
                             total_HE == 1 ~ 1,
                             total_HE == 2 ~ 1),
         HRscore = case_when(total_HR == 0 ~ 0,
```

```
                              total_HR == 1 ~ 1,
                              total_HR == 2 ~ 1))

# Add a measure of total heuristic response, where 1 = at least one mistake in one heuristic category, 
data_Q_total <- data_Q_total %>%
  group_by(Participant.Private.ID) %>%
  mutate(HEHRscore = HEscore + HRscore)
```

Randomness and Probability

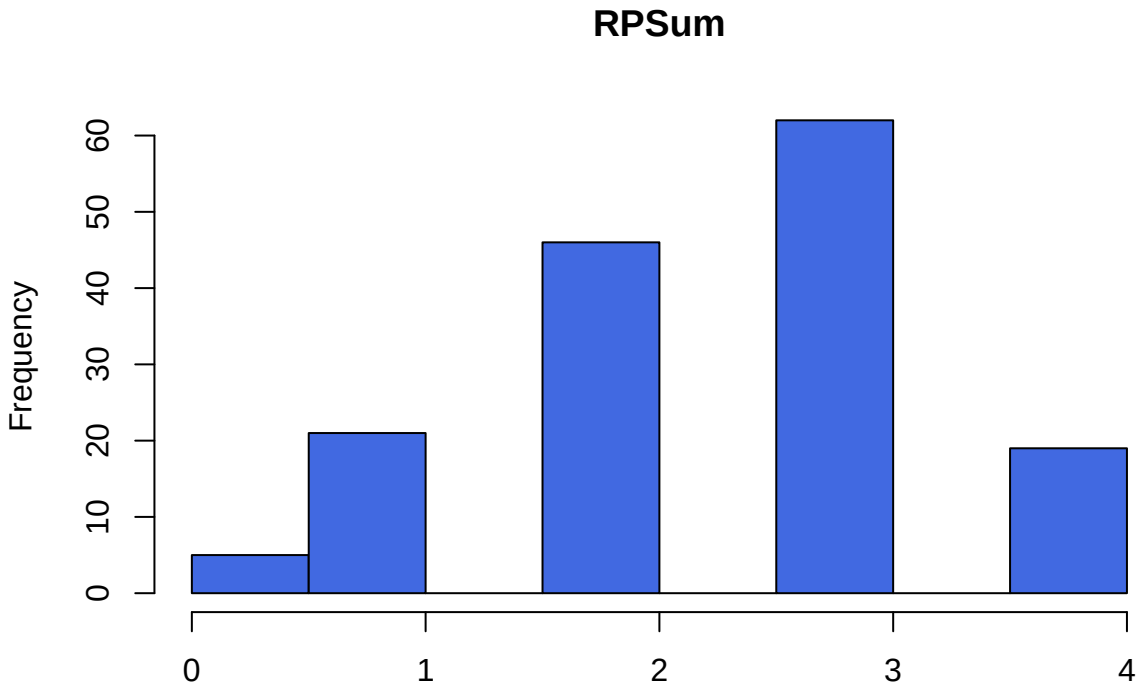
New columns are created for each item, specifying whether the participant got the question correct or not (0=incorrect, 1=correct). A sum score is calculated for all the items.

```
# Q1: Sairaala B is the correct answer
# Create a new column
data_Q_total$rp.1 <- as.factor(data_Q_total$rp.1)
data_Q_total <- data_Q_total %>%
  mutate(rp.1.int = case_when(rp.1 == "Sairaalassa B (jossa syntyy 10 lasta pÃ¤ivÃ¤ssÃ¤)." ~ 1,
                              rp.1 %in% c("Sairaalassa A (jossa syntyy 50 lasta pÃ¤ivÃ¤ssÃ¤).", "TÃ¤mÃ¤

# Q2: Pyydys1 AND Pyydys 2 is the correct combination
data_Q_total$rp.2.1 <- as.factor(data_Q_total$rp.2.1)
data_Q_total$rp.2.2 <- as.factor(data_Q_total$rp.2.2)
data_Q_total$rp.2.3 <- as.factor(data_Q_total$rp.2.3)
data_Q_total$rp.2.4 <- as.factor(data_Q_total$rp.2.4)
data_Q_total$rp.2.5 <- as.factor(data_Q_total$rp.2.5)
data_Q_total$rp.2.6 <- as.factor(data_Q_total$rp.2.6)
data_Q_total$rp.2.7 <- as.factor(data_Q_total$rp.2.7)
data_Q_total$rp.2.8 <- as.factor(data_Q_total$rp.2.8)
# Create a new column
# NAs are initially coded as 2 and wrong answers (Os) as NAs; then these are recoded
data_Q_total <- data_Q_total %>%
  mutate(rp.2.int = case_when((rp.2.1 == "Pyydys 1" & rp.2.2 == "Pyydys 2" & rp.2.3 == "" & rp.2.4 == "
                              is.na(rp.2.8) ~ 2))
data_Q_total[which(is.na(data_Q_total$rp.2.int)), "rp.2.int"] <- 0
data_Q_total[which(data_Q_total$rp.2.int == 2), ] <- NA

# Q3: Kanava 1; 2 tai 3 is the correct answer
# Create a new column
data_Q_total <- data_Q_total %>%
  mutate(rp.3.int = case_when(rp.3.quantised == "4" ~ 1,
                              rp.3.quantised != "4" ~ 0))

# Q4: Ruudukot A; B ja C is the correct answer
# Create a new column
data_Q_total <- data_Q_total %>%
  mutate(rp.4.int = case_when(rp.4.quantised == "5" ~ 1,
                              rp.4.quantised != "5" ~ 0))

# Calculate the sum score for the randomness/probability questions
data_Q_total <- data_Q_total %>%
  mutate(RPSum = rp.1.int + rp.2.int + rp.3.int + rp.4.int)
```

```r
# Describe
describe(data_Q_total$RPSum)
```

```
##    vars   n mean   sd median trimmed  mad min max range  skew kurtosis   se
## X1    1 153 2.45 0.99      3    2.48 1.48   0   4     4 -0.42     -0.3 0.08
```

```r
hist(data_Q_total$RPSum,
     main = "RPSum",
     col = sample(colors(), 1),
     xlab = "")
```



Matrix Reasoning

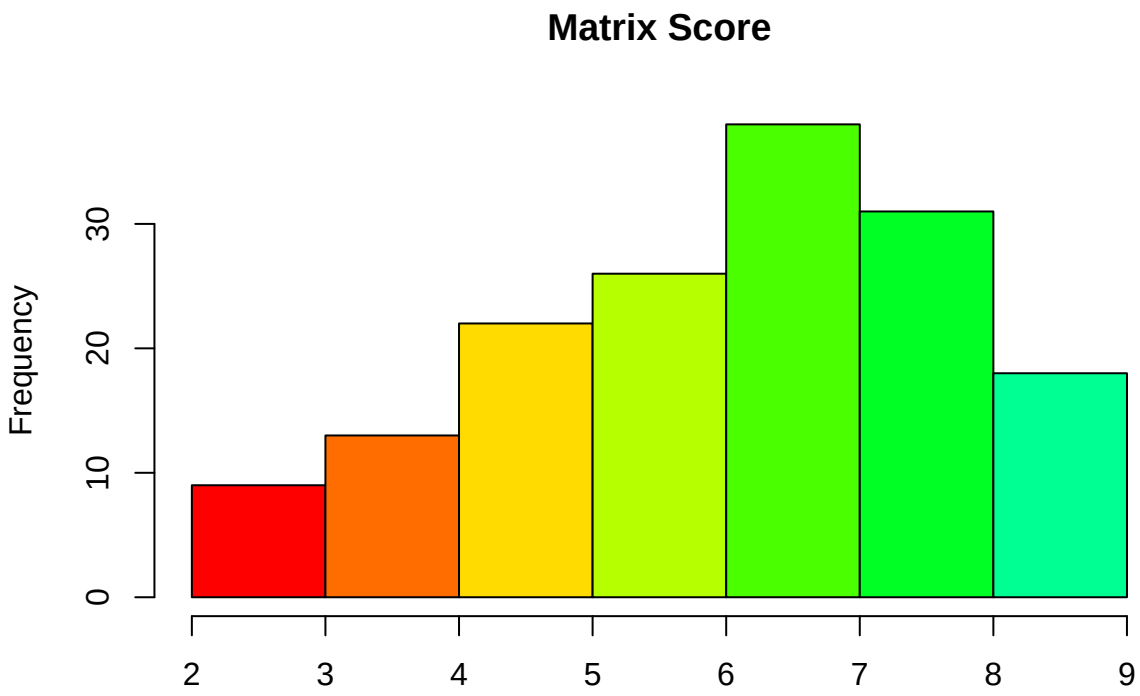
For the matrix reasoning task (treated as a questionnaire by Gorilla so included in this section), a participant's data for that task is replaced with missing values (NAs) if they met any of the following exclusion criteria: * medium reaction time equal to or smaller than 500ms * stated they had technical problems or did not understand the task

```r
# Omit columns that are not relevant (experiment-general columns) + rows that are not relevant (e.g., p
data_matreasT <- data_matreasT %>%
  select(Participant.Private.ID, Spreadsheet:ANSWER) %>%
  filter(display %in% c("TehtÃ¤vÃ¤_6", "TehtÃ¤vÃ¤_8"))

# Omit participants who commented that they had technical issues with this task or did not understand t
```

```r
data_matreasT <- data_matreasT %>%
  filter(Participant.Private.ID != "5555882" & Participant.Private.ID != "5608075" & Participant.Private

# Extract the relevant information
data_matreasT_final <- data_matreasT %>%
  group_by(Participant.Private.ID) %>%
  summarise(MatrixCorrectCount = sum(Correct, na.rm = TRUE)) %>%
  select(Participant.Private.ID, MatrixCorrectCount)

# Exclude participants with invalid responses
data_ValidityFlag <- data_validQ %>%
  select(Participant.Private.ID, Validity.quantised)
data_Language <- data_demoQ %>%
  select(Participant.Private.ID, Language)
data_matrixValidity <- merge(data_Language, data_ValidityFlag,
                             by = "Participant.Private.ID",
                             all = TRUE)
data_matreasT_final <- merge(data_matreasT_final, data_matrixValidity,
                             by = "Participant.Private.ID",
                             all = TRUE)

# Create separate df for nonvalid ppts
data_nonvalid_matrix <- data_matreasT_final %>%
  filter(Validity.quantised %in% c(2, 3, 4))
data_Q_lang_omit_matrix <- data_matreasT_final %>%
  filter(Language != "Sujuva / Ã¤idinkieli" & Language != "Keskitaso / keskusteleva" & Language != "Muu
data_matreasT_final <- data_matreasT_final %>%
  filter(Validity.quantised %in% c("1", NA, "") & Language %in% c("Sujuva / Ã¤idinkieli", "Keskitaso / K

# Delete the irrelevant columns
data_matreasT_final <- data_matreasT_final %>%
  select(Participant.Private.ID, MatrixCorrectCount)

# Describe
describe(data_matreasT_final$MatrixCorrectCount)
```

```
##    vars   n mean   sd median trimmed  mad min max range  skew kurtosis   se
## X1    1 157 6.47 1.75      7    6.57 1.48   2   9     7 -0.56    -0.25 0.14
```

```r
hist(data_matreasT_final$MatrixCorrectCount,
     main = "Matrix Score",
     col = rainbow(14),
     xlab = "")
```



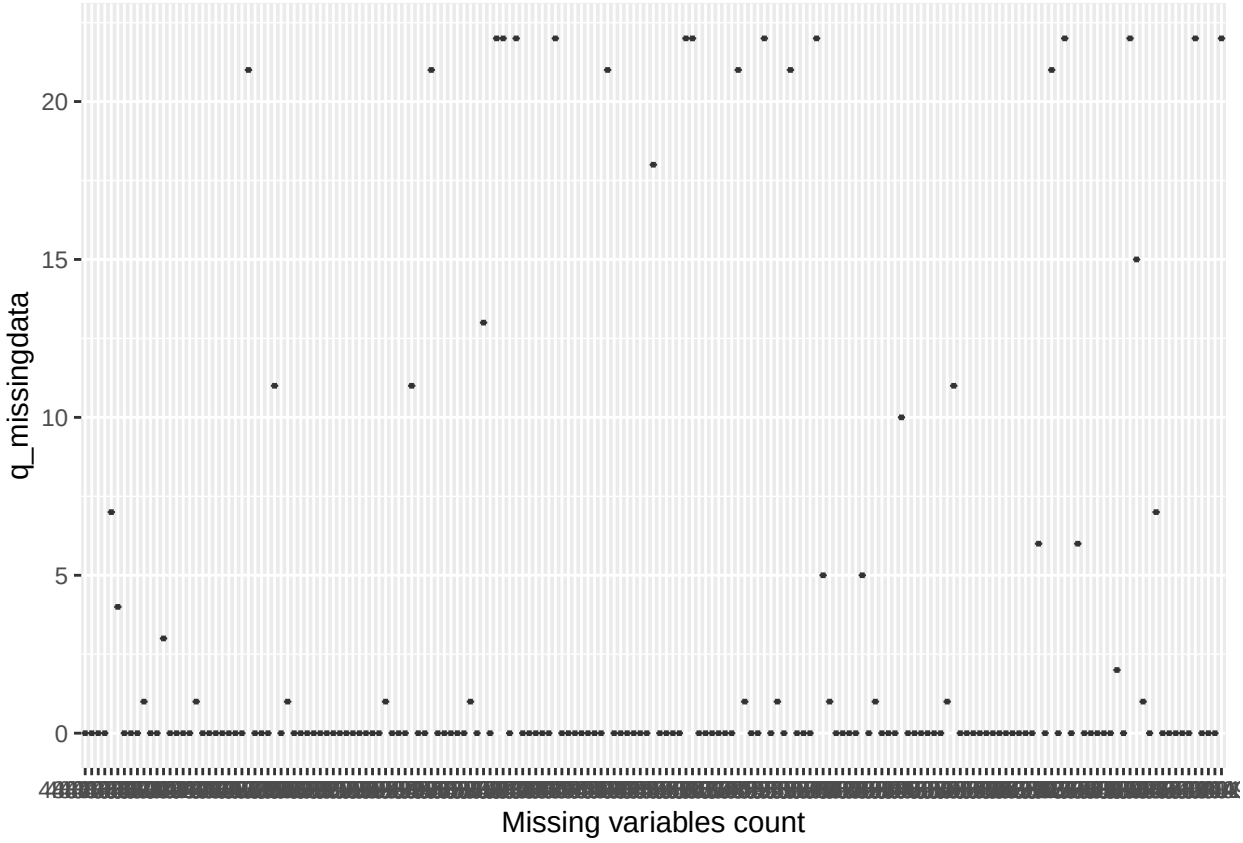
Merge Questionnaire Variables

```
# Merge the important columns
data_Q_sub <- data_Q_total %>%
  select(Participant.Private.ID, gender, Age, Country, Language, Education, AMean, CMean, EMean, ESMean

# Add matrix reasoning data to the dataframe
data_Q_sub <- merge(data_Q_sub, data_matreasT_final,
                    by = "Participant.Private.ID",
                    all = TRUE)

# Get rid of NaNs or string NAs introduced by calculations
is.nan.data.frame <- function(x)
  do.call(cbind, lapply(x, is.nan))
data_Q_sub[is.nan(data_Q_sub)] <- NA

# Numeric variables into numeric
data_Q_sub$MatrixCorrectCount <- as.numeric(data_Q_sub$MatrixCorrectCount)
data_Q_sub$Age <- as.numeric(data_Q_sub$Age)
```

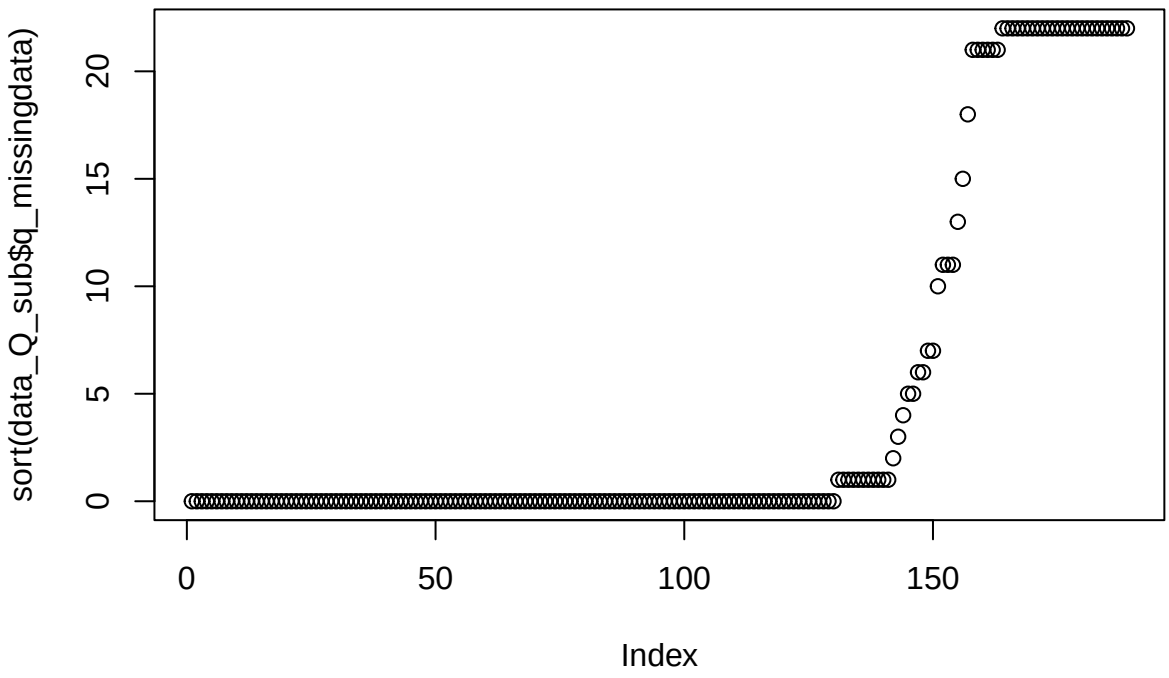
Analyze Missing Data in Questionnaires

Participants with large amounts of missing data or suspicious patterns of missing data are removed here. It should be noted that missing data refers to data that was missing from the start *and* data that was replaced with missing values if the participant met one of the exclusion criteria for (a) questionnaire(s).

```
# Questionnaire variables
q_var <- c("AMean", "CMean", "EMean", "ESMean", "OMean", "ICuriositySum", "DCuriositySum", "IH1Sum", "I

# Add a column for count of missing data for questionnaire variables per participant
data_Q_sub$q_missingdata <- rowSums(is.na(data_Q_sub[, q_var]))

# Plot number of missing values
ggplot(data_Q_sub, aes(x = as.factor(Participant.Private.ID), y = q_missingdata)) +
   geom_boxplot(fill = "slateblue", alpha = 0.2) +
   xlab("Missing variables count")
```

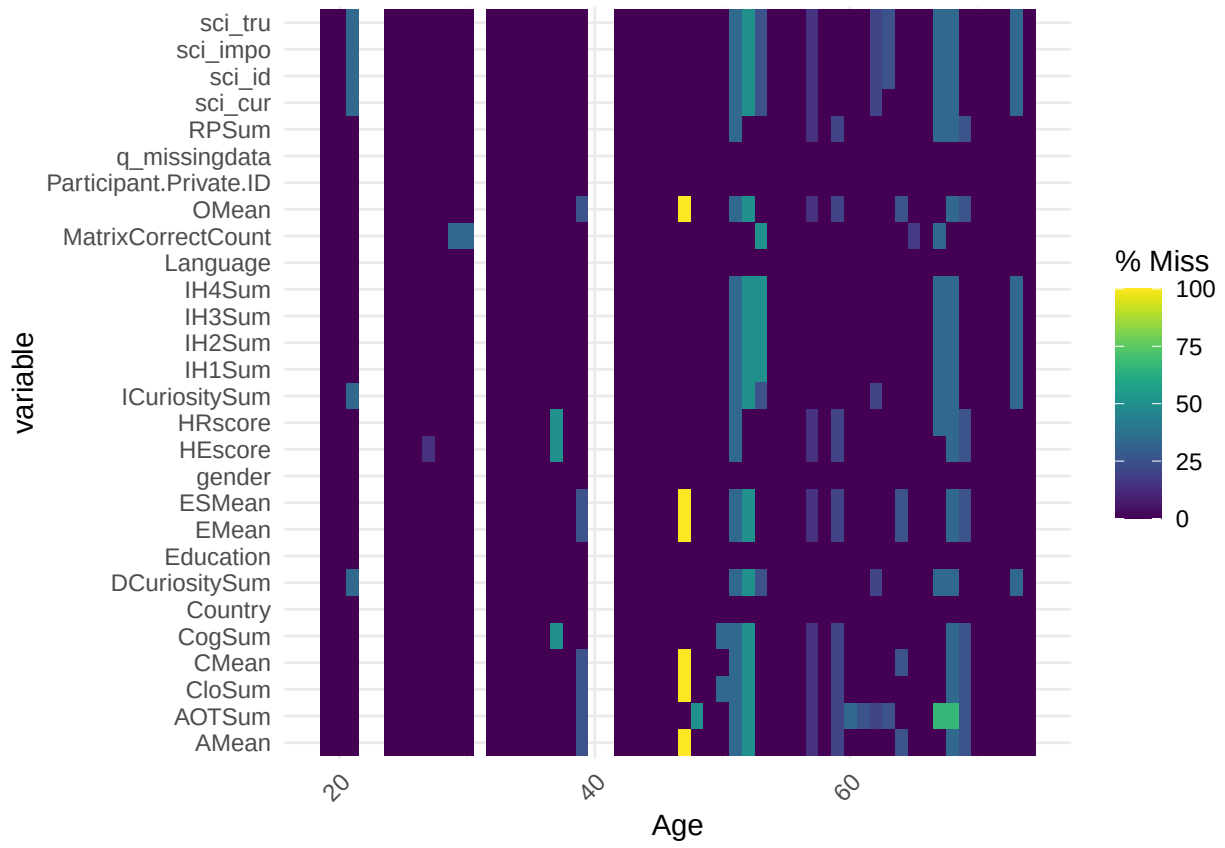



```
# Plot to look for an elbow value
plot(sort(data_Q_sub$q_missingdata))
```




```
# Correlation between age and amount of missing values
gg_miss_fct(x = data_Q_sub, fct = Age)
```

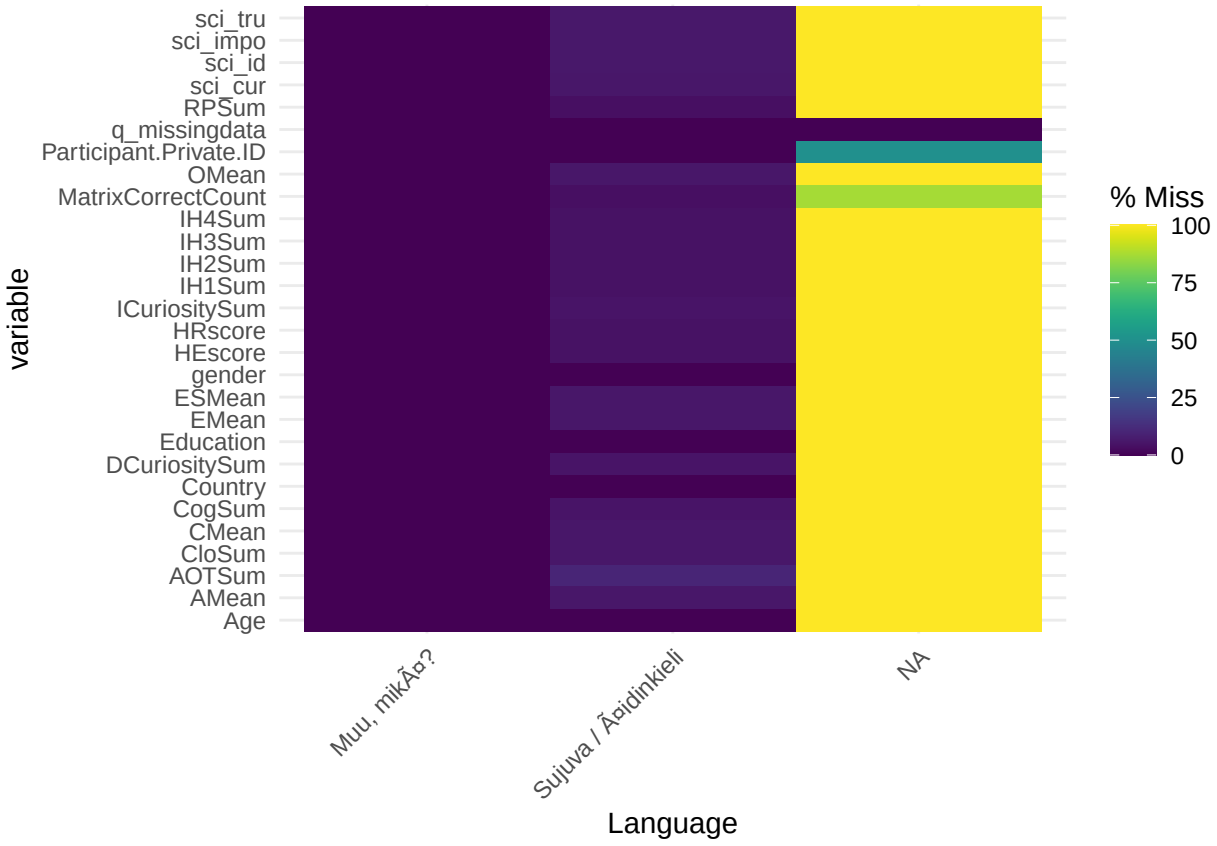
```
## Warning: Removed 28 rows containing missing values (geom_tile).
```



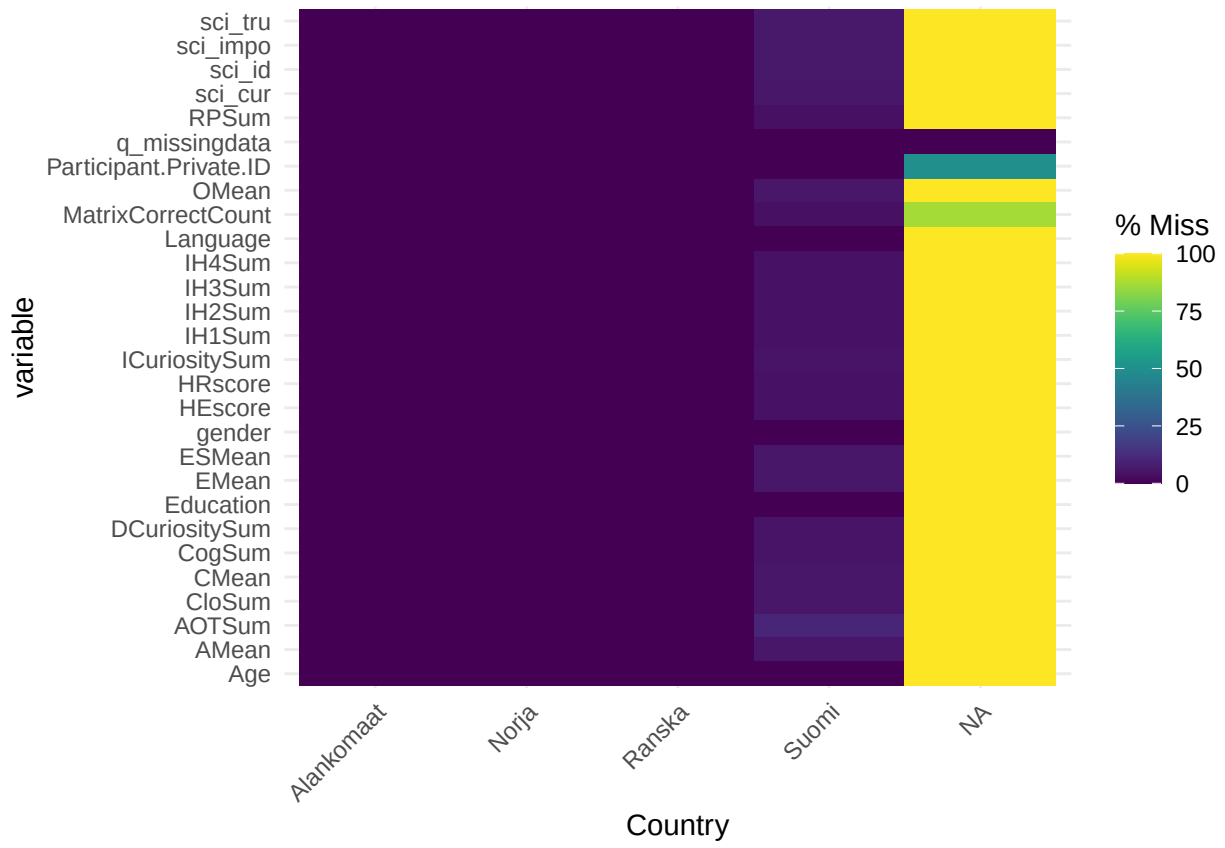

```
cor.test(data_Q_sub$Age, data_Q_sub$q_missingdata,
         method = "spearman", exact = FALSE)
```

```
##
##  Spearman's rank correlation rho
##
## data:  data_Q_sub$Age and data_Q_sub$q_missingdata
## S = 555745, p-value = 0.03173
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
##       rho
## 0.1704308
```

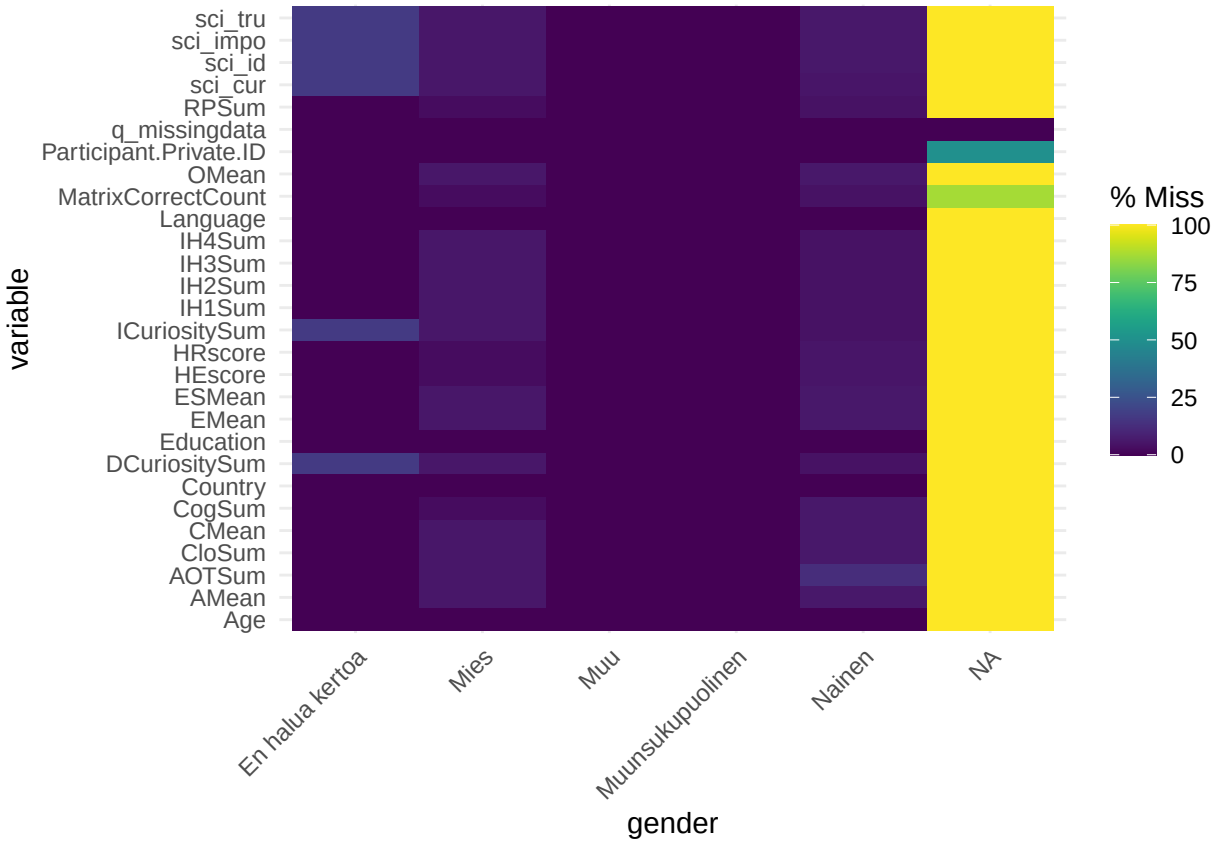
```
# Visualize missing data by demographics
gg_miss_fct(x = data_Q_sub, fct = Language)
```



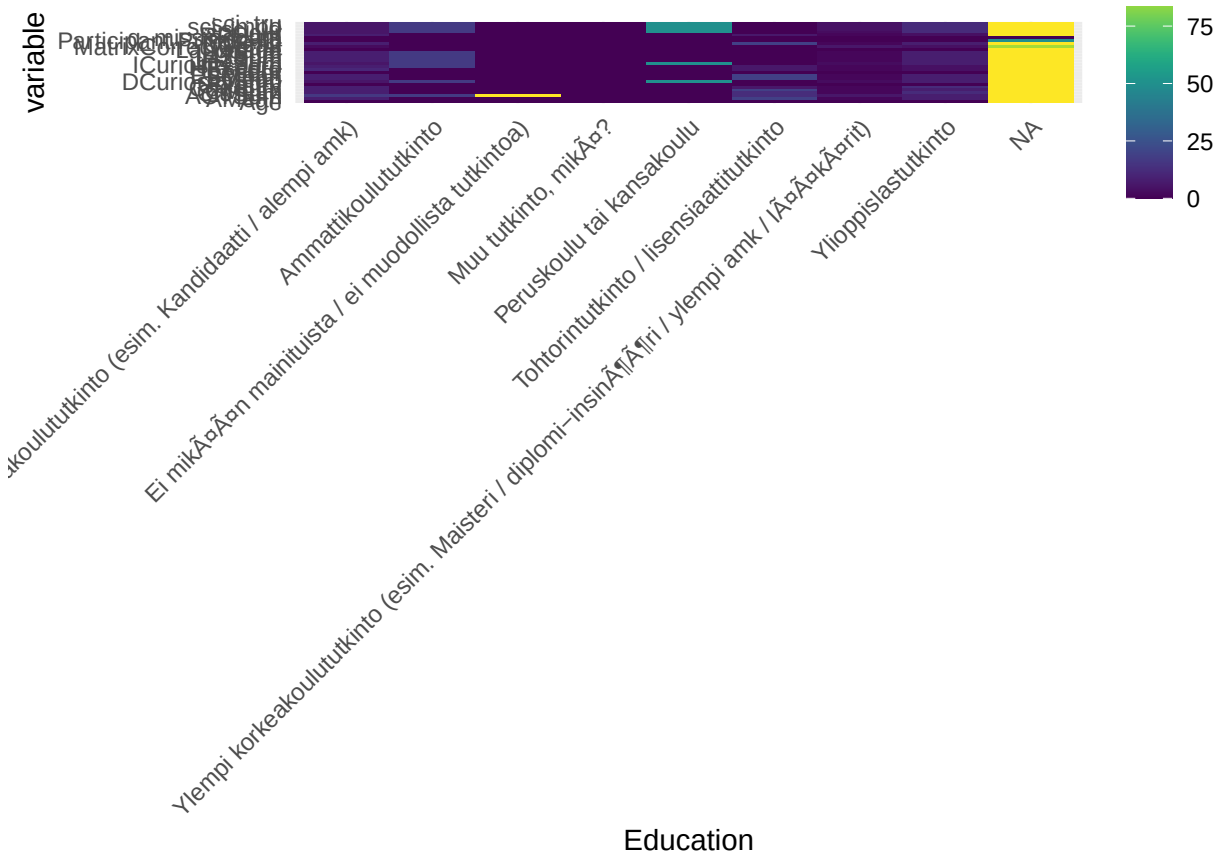

```
gg_miss_fct(x = data_Q_sub, fct = Country)
```




```
gg_miss_fct(x = data_Q_sub, fct = gender)
```




```
gg_miss_fct(x = data_Q_sub, fct = Education)
```

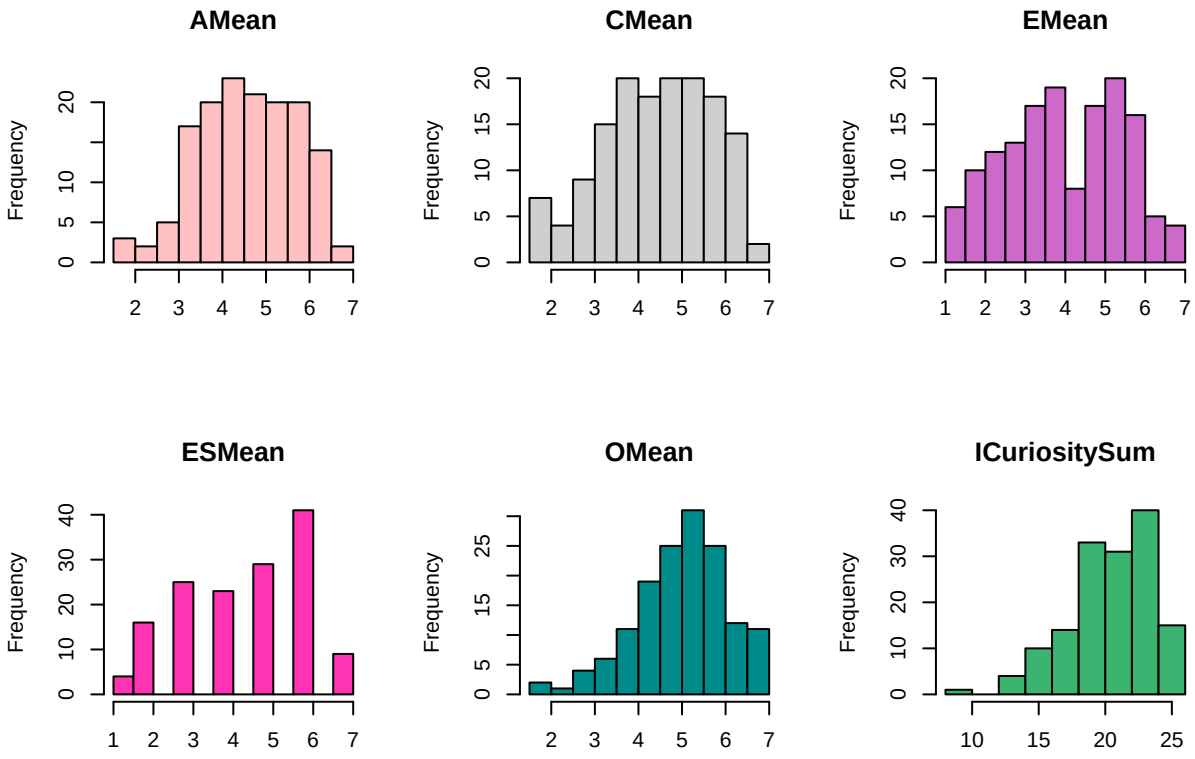


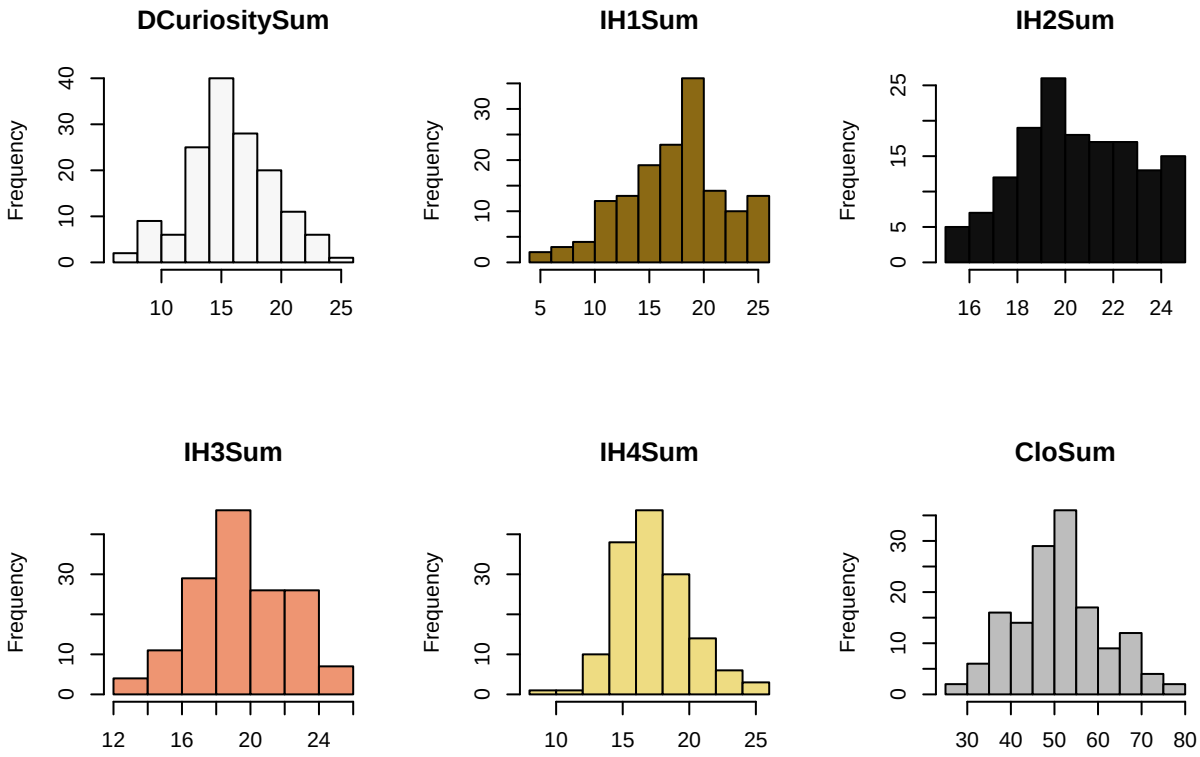

```
# Remove ppts with 7 or fewer variables missing out of 22 (~32%)
data_Q_sub <- data_Q_sub %>%
  filter(q_missingdata <= 7)
```

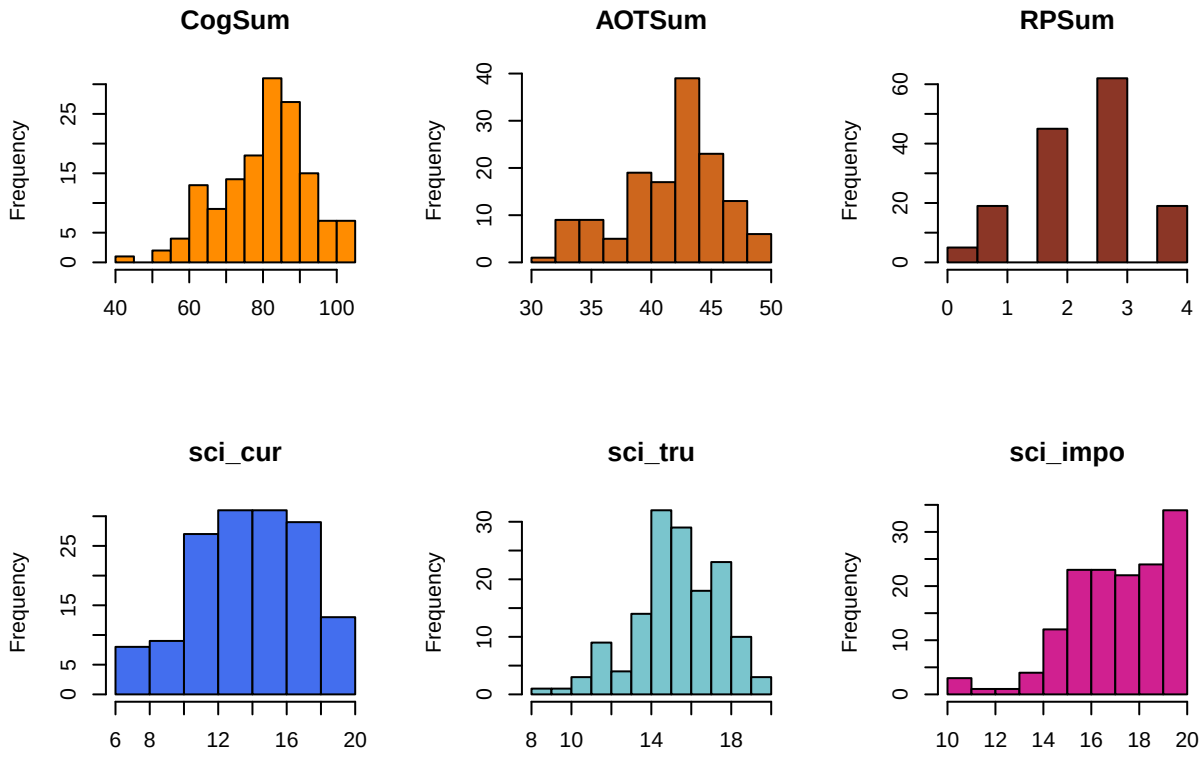
Check Questionnaire Distributions and Correlations

This chunk checks distributions of questionnaire variables and correlations between them: * If distributions are highly skewed, transformations are applied to decrease skewness (skew > 1) * If two variables are highly correlated, one of them is dropped (Eisenberg et al.) (r > .85)

```
# Questionnaire variables without binary ones
q_var_con <- c("AMean", "CMean", "EMean", "ESMean", "OMean", "ICuriositySum", "DCuriositySum", "IH1Sum"

# Normality of variables: visualize with histograms
par(mfrow = c(2, 3))
for (i in 1:length(q_var_con)) {
  hist(data_Q_sub[, q_var_con[i]],
       main = colnames(data_Q_sub[q_var_con[i]]),
       col = sample(colors(), 1),
       xlab = "")
}
```


AMean
Frequency
0
5
10
20
2
3
4
5
6
7
CMean
Frequency
0
5
10
15
20
2
3
4
5
6
7
EMean
Frequency
0
5
10
15
20
1
2
3
4
5
6
7
ESMean
Frequency
0
10
20
30
40
1
2
3
4
5
6
7
OMean
Frequency
0
5
15
25
2
3
4
5
6
7
ICuriositySum
Frequency
0
10
20
30
40
10
15
20
25


DCuriositySum
Frequency
0
10
20
30
40
10
15
20
25
IH1Sum
Frequency
0
10
20
30
5
10
15
20
25
IH2Sum
Frequency
0
5
15
25
16
18
20
22
24
IH3Sum
Frequency
0
10
30
12
16
20
24
IH4Sum
Frequency
0
10
30
10
15
20
25
CloSum
Frequency
0
10
20
30
30
40
50
60
70
80




```
# Describe: mean, skew, kurtosis
describe(data_Q_sub[, q_var_con])[c("mean", "skew", "kurtosis")]
```

```
##                       mean  skew kurtosis
## AMean                 4.81 -0.38    -0.13
## CMean                 4.67 -0.35    -0.57
## EMean                 4.22 -0.13    -0.95
## ESMean                4.47 -0.33    -0.96
## OMean                 5.23 -0.62     0.52
## ICuriositySum        20.97 -0.79     0.59
## DCuriositySum        16.30 -0.06    -0.12
## IH1Sum               17.80 -0.38    -0.34
## IH2Sum               20.96 -0.12    -0.69
## IH3Sum               19.95 -0.17    -0.37
## IH4Sum               17.80  0.31     0.48
## CloSum               51.43  0.20    -0.02
## CogSum               81.28 -0.43    -0.17
## AOTSum               42.23 -0.64    -0.23
## RPSum                 2.47 -0.46    -0.22
## sci_cur              14.32 -0.21    -0.66
## sci_tru              15.79 -0.56     0.48
## sci_impo             17.58 -0.88     0.87
## sci_id               16.48 -0.76     0.03
## MatrixCorrectCount    6.61 -0.54    -0.20
```

```r
# Correlation matrix
round(cor(data_Q_sub[, q_var_con], method = "pearson",
          use = "pairwise.complete.obs"), 2)
```

```
##                    AMean CMean EMean ESMean OMean ICuriositySum DCuriositySum
## AMean               1.00  0.04  0.12   0.24  0.16         -0.06         -0.19
## CMean               0.04  1.00  0.14   0.41 -0.06         -0.02         -0.02
## EMean               0.12  0.14  1.00   0.03  0.17          0.10          0.05
## ESMean              0.24  0.41  0.03   1.00 -0.06          0.07         -0.10
## OMean               0.16 -0.06  0.17  -0.06  1.00          0.45          0.30
## ICuriositySum      -0.06 -0.02  0.10   0.07  0.45          1.00          0.59
## DCuriositySum      -0.19 -0.02  0.05  -0.10  0.30          0.59          1.00
## IH1Sum             -0.20  0.11  0.06   0.22  0.19          0.13          0.05
## IH2Sum              0.00 -0.19  0.04  -0.12  0.28          0.43          0.34
## IH3Sum              0.26  0.15  0.18   0.19  0.35          0.31          0.08
## IH4Sum              0.10 -0.15 -0.04  -0.05 -0.04          0.00         -0.17
## CloSum              0.02  0.12 -0.30  -0.06 -0.32         -0.28         -0.04
## CogSum             -0.10  0.10  0.24   0.15  0.36          0.64          0.50
## AOTSum             -0.22 -0.21 -0.13  -0.01  0.07          0.32          0.16
## RPSum              -0.03  0.00 -0.15  -0.02 -0.04          0.13          0.08
## sci_cur            -0.20 -0.02 -0.02   0.05  0.23          0.53          0.31
## sci_tru            -0.10  0.06  0.08  -0.10 -0.01          0.12          0.02
## sci_impo           -0.07 -0.13  0.06  -0.04  0.09          0.37          0.19
## sci_id             -0.11  0.16  0.16   0.07  0.19          0.44          0.28
## MatrixCorrectCount  0.01 -0.17 -0.04  -0.01 -0.02          0.15          0.10
##                    IH1Sum IH2Sum IH3Sum IH4Sum CloSum CogSum AOTSum RPSum
## AMean               -0.20   0.00   0.26   0.10   0.02  -0.10  -0.22 -0.03
## CMean                0.11  -0.19   0.15  -0.15   0.12   0.10  -0.21  0.00
## EMean                0.06   0.04   0.18  -0.04  -0.30   0.24  -0.13 -0.15
## ESMean               0.22  -0.12   0.19  -0.05  -0.06   0.15  -0.01 -0.02
## OMean                0.19   0.28   0.35  -0.04  -0.32   0.36   0.07 -0.04
## ICuriositySum        0.13   0.43   0.31   0.00  -0.28   0.64   0.32  0.13
## DCuriositySum        0.05   0.34   0.08  -0.17  -0.04   0.50   0.16  0.08
## IH1Sum               1.00   0.05   0.18  -0.17  -0.39   0.27   0.06 -0.01
## IH2Sum               0.05   1.00   0.22   0.23  -0.07   0.29   0.42  0.27
## IH3Sum               0.18   0.22   1.00   0.03  -0.26   0.18  -0.04  0.04
## IH4Sum              -0.17   0.23   0.03   1.00  -0.07  -0.14   0.19  0.04
## CloSum              -0.39  -0.07  -0.26  -0.07   1.00  -0.38  -0.12  0.08
## CogSum               0.27   0.29   0.18  -0.14  -0.38   1.00   0.19  0.12
## AOTSum               0.06   0.42  -0.04   0.19  -0.12   0.19   1.00  0.23
## RPSum               -0.01   0.27   0.04   0.04   0.08   0.12   0.23  1.00
## sci_cur              0.20   0.22   0.10  -0.02  -0.14   0.39   0.15  0.21
## sci_tru              0.08   0.19  -0.03   0.02   0.03   0.20   0.13  0.17
## sci_impo             0.07   0.21  -0.02   0.03  -0.08   0.33   0.24  0.12
## sci_id               0.18   0.11   0.09  -0.25  -0.13   0.51   0.08  0.31
## MatrixCorrectCount   0.03   0.06   0.09   0.07  -0.10   0.11   0.13  0.34
##                    sci_cur sci_tru sci_impo sci_id MatrixCorrectCount
## AMean                -0.20   -0.10    -0.07  -0.11               0.01
## CMean                -0.02    0.06    -0.13   0.16              -0.17
## EMean                -0.02    0.08     0.06   0.16              -0.04
## ESMean                0.05   -0.10    -0.04   0.07              -0.01
## OMean                 0.23   -0.01     0.09   0.19              -0.02
## ICuriositySum         0.53    0.12     0.37   0.44               0.15
```

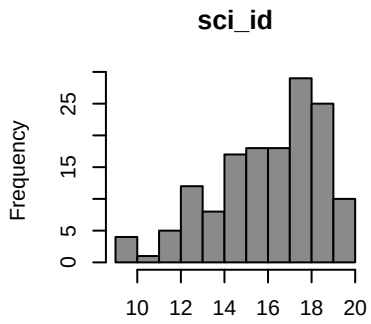
```
## DCuriositySum         0.31    0.02     0.19   0.28              0.10
## IH1Sum                0.20    0.08     0.07   0.18              0.03
## IH2Sum                0.22    0.19     0.21   0.11              0.06
## IH3Sum                0.10   -0.03    -0.02   0.09              0.09
## IH4Sum               -0.02    0.02     0.03  -0.25              0.07
## CloSum               -0.14    0.03    -0.08  -0.13             -0.10
## CogSum                0.39    0.20     0.33   0.51              0.11
## AOTSum                0.15    0.13     0.24   0.08              0.13
## RPSum                 0.21    0.17     0.12   0.31              0.34
## sci_cur               1.00    0.23     0.39   0.49              0.07
## sci_tru               0.23    1.00     0.33   0.20              0.05
## sci_impo              0.39    0.33     1.00   0.43              0.16
## sci_id                0.49    0.20     0.43   1.00              0.08
## MatrixCorrectCount    0.07    0.05     0.16   0.08              1.00
```

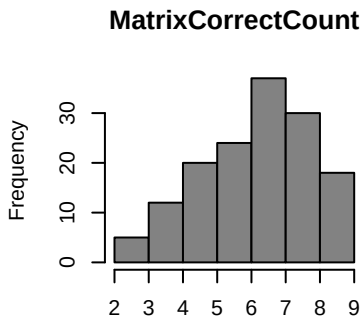
```
abs(round(cor(data_Q_sub[, q_var_con], method = "pearson",
              use = "pairwise.complete.obs"), 2)) > .85
```

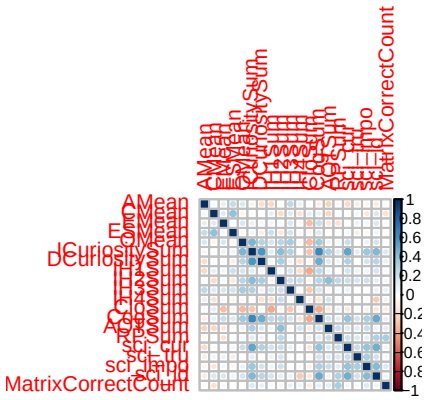
```
##                    AMean CMean EMean ESMean OMean ICuriositySum DCuriositySum
## AMean               TRUE FALSE FALSE  FALSE FALSE         FALSE         FALSE
## CMean              FALSE  TRUE FALSE  FALSE FALSE         FALSE         FALSE
## EMean              FALSE FALSE  TRUE  FALSE FALSE         FALSE         FALSE
## ESMean             FALSE FALSE FALSE   TRUE FALSE         FALSE         FALSE
## OMean              FALSE FALSE FALSE  FALSE  TRUE         FALSE         FALSE
## ICuriositySum      FALSE FALSE FALSE  FALSE FALSE          TRUE         FALSE
## DCuriositySum      FALSE FALSE FALSE  FALSE FALSE         FALSE          TRUE
## IH1Sum             FALSE FALSE FALSE  FALSE FALSE         FALSE         FALSE
## IH2Sum             FALSE FALSE FALSE  FALSE FALSE         FALSE         FALSE
## IH3Sum             FALSE FALSE FALSE  FALSE FALSE         FALSE         FALSE
## IH4Sum             FALSE FALSE FALSE  FALSE FALSE         FALSE         FALSE
## CloSum             FALSE FALSE FALSE  FALSE FALSE         FALSE         FALSE
## CogSum             FALSE FALSE FALSE  FALSE FALSE         FALSE         FALSE
## AOTSum             FALSE FALSE FALSE  FALSE FALSE         FALSE         FALSE
## RPSum              FALSE FALSE FALSE  FALSE FALSE         FALSE         FALSE
## sci_cur            FALSE FALSE FALSE  FALSE FALSE         FALSE         FALSE
## sci_tru            FALSE FALSE FALSE  FALSE FALSE         FALSE         FALSE
## sci_impo           FALSE FALSE FALSE  FALSE FALSE         FALSE         FALSE
## sci_id             FALSE FALSE FALSE  FALSE FALSE         FALSE         FALSE
## MatrixCorrectCount FALSE FALSE FALSE  FALSE FALSE         FALSE         FALSE
##                    IH1Sum IH2Sum IH3Sum IH4Sum CloSum CogSum AOTSum RPSum
## AMean               FALSE  FALSE  FALSE  FALSE  FALSE  FALSE  FALSE FALSE
## CMean               FALSE  FALSE  FALSE  FALSE  FALSE  FALSE  FALSE FALSE
## EMean               FALSE  FALSE  FALSE  FALSE  FALSE  FALSE  FALSE FALSE
## ESMean              FALSE  FALSE  FALSE  FALSE  FALSE  FALSE  FALSE FALSE
## OMean               FALSE  FALSE  FALSE  FALSE  FALSE  FALSE  FALSE FALSE
## ICuriositySum       FALSE  FALSE  FALSE  FALSE  FALSE  FALSE  FALSE FALSE
## DCuriositySum       FALSE  FALSE  FALSE  FALSE  FALSE  FALSE  FALSE FALSE
## IH1Sum               TRUE  FALSE  FALSE  FALSE  FALSE  FALSE  FALSE FALSE
## IH2Sum              FALSE   TRUE  FALSE  FALSE  FALSE  FALSE  FALSE FALSE
## IH3Sum              FALSE  FALSE   TRUE  FALSE  FALSE  FALSE  FALSE FALSE
## IH4Sum              FALSE  FALSE  FALSE   TRUE  FALSE  FALSE  FALSE FALSE
## CloSum              FALSE  FALSE  FALSE  FALSE   TRUE  FALSE  FALSE FALSE
## CogSum              FALSE  FALSE  FALSE  FALSE  FALSE   TRUE  FALSE FALSE
## AOTSum              FALSE  FALSE  FALSE  FALSE  FALSE  FALSE   TRUE FALSE
```

```
## RPSum                FALSE  FALSE  FALSE  FALSE  FALSE  FALSE  FALSE  TRUE
## sci_cur              FALSE  FALSE  FALSE  FALSE  FALSE  FALSE  FALSE FALSE
## sci_tru              FALSE  FALSE  FALSE  FALSE  FALSE  FALSE  FALSE FALSE
## sci_impo             FALSE  FALSE  FALSE  FALSE  FALSE  FALSE  FALSE FALSE
## sci_id               FALSE  FALSE  FALSE  FALSE  FALSE  FALSE  FALSE FALSE
## MatrixCorrectCount   FALSE  FALSE  FALSE  FALSE  FALSE  FALSE  FALSE FALSE
##                     sci_cur sci_tru sci_impo sci_id MatrixCorrectCount
## AMean                 FALSE   FALSE    FALSE  FALSE              FALSE
## CMean                 FALSE   FALSE    FALSE  FALSE              FALSE
## EMean                 FALSE   FALSE    FALSE  FALSE              FALSE
## ESMean                FALSE   FALSE    FALSE  FALSE              FALSE
## OMean                 FALSE   FALSE    FALSE  FALSE              FALSE
## ICuriositySum         FALSE   FALSE    FALSE  FALSE              FALSE
## DCuriositySum         FALSE   FALSE    FALSE  FALSE              FALSE
## IH1Sum                FALSE   FALSE    FALSE  FALSE              FALSE
## IH2Sum                FALSE   FALSE    FALSE  FALSE              FALSE
## IH3Sum                FALSE   FALSE    FALSE  FALSE              FALSE
## IH4Sum                FALSE   FALSE    FALSE  FALSE              FALSE
## CloSum                FALSE   FALSE    FALSE  FALSE              FALSE
## CogSum                FALSE   FALSE    FALSE  FALSE              FALSE
## AOTSum                FALSE   FALSE    FALSE  FALSE              FALSE
## RPSum                 FALSE   FALSE    FALSE  FALSE              FALSE
## sci_cur                TRUE   FALSE    FALSE  FALSE              FALSE
## sci_tru               FALSE    TRUE    FALSE  FALSE              FALSE
## sci_impo              FALSE   FALSE     TRUE  FALSE              FALSE
## sci_id                FALSE   FALSE    FALSE   TRUE              FALSE
## MatrixCorrectCount    FALSE   FALSE    FALSE  FALSE               TRUE
```

```
corrplot(cor(data_Q_sub[, q_var_con], method = "pearson",
             use = "pairwise.complete.obs"))

# Subset data: exclude demographic variables
data_Q_sub <- data_Q_sub[, c("Participant.Private.ID", q_var_con)]
```







Task Data

Exclusion Criteria

ALL TASKS: This chunk goes through all task data, removing **all data for a participant** if they met one or more of the following exclusion criteria: * stated their data are not valid * stated their Finnish is not Äidinkieli or Keskusteleva

TASKS: A participant's data for a questionnaire was replaced with missing values (NAs) if they met the following exclusion criterion: * stopped during that task

For further exclusion criteria for the tasks, please see each chunk.

Task Data Processing

See each chunk.

Go No Go

Trial-level exclusion criteria: * The reaction time for each row is replaced with missing values (NAs) if it is equal to or higher than 1025ms (screen time 1000ms + 16.67ms refresh rate + ~8ms latency between computer and mouse -> reaction times higher than this are likely not valid) *or* if it is equal to or lower than 150ms (Jaana's paper).

Participant-level exclusion criteria: * A participant's data for the Go No-Go task is replaced with missing values (NAs) if their accuracy (% of correct trials out of all trials) was equal to or lower than 60%.

Data processing: for each participant, we calculate the sum of hits/misses/FAs/CRs as well as dprime, beta, c, aprime, and bppd (see the dprime() function help file). These are described and visualized.

```
# Omit experiment-general columns & practice (etc.) rows
data_nogoT <- data_nogoT %>%
  select(Participant.Private.ID, Spreadsheet:Answer) %>%
  filter(display == "Trials")

# Exclusion Criteria: Trial-Level
  # RT <= 150ms or >= 1025ms
data_nogoT$Reaction.Time[which(data_nogoT$Reaction.Time >= 1025 | data_nogoT$Reaction.Time <= 150)] <- I

# Exclusion Criteria: Participant-Level
  # <=60% accuracy
data_nogoT <- data_nogoT %>%
  group_by(Participant.Private.ID) %>%
  mutate(Accuracy = sum(Correct, na.rm = TRUE) / (sum(Correct, na.rm = TRUE) + sum(Incorrect, na.rm = TI
  mutate(go.nogo_omit = case_when(Accuracy > .60 ~ 0,
                                  Accuracy <= .60 ~ 1))

# Omit rows based on exclusion criteria
data_nogoT$Response[which(data_nogoT$go.nogo_omit == 1)] <- NA

# Create Signal Detection Theory categories for each row
data_nogoT <- data_nogoT %>%
  filter(display == "Trials") %>%
  mutate(SDT = case_when((Array %in% c("H.png",  "T.png") & Response == "Go") ~ "Hit",
                         (Array %in% c("H.png", "T.png") & Response == "No Go") ~ "Miss",
                         (Array == "N.png" & Response == "Go") ~ "False Alarm",
                         (Array == "N.png" & Response == "No Go") ~ "Correct Rejection")) %>%
    mutate(Hit = case_when(SDT == "Hit" ~ 1,
                           SDT != "Hit" ~ 0),
           Miss = case_when(SDT == "Miss" ~ 1,
                            SDT != "Miss" ~ 0),
           FA = case_when(SDT == "False Alarm" ~ 1,
                          SDT != "False Alarm" ~ 0),
           CR = case_when(SDT == "Correct Rejection" ~ 1,
                          SDT != "Correct Rejection" ~ 0))

# Calculate the mean and SD of hits and FA (trials with a go response)
data_go_RT <- data_nogoT %>%
  filter(SDT %in% c("Hit", "FA")) %>%
  group_by(Participant.Private.ID) %>%
  summarise(MeanGoRT = mean(Reaction.Time, na.rm = TRUE),
            SDGoRT = sd(Reaction.Time, na.rm = TRUE))

# Compute D-Prime and Bias for each participant
# https://www.rdocumentation.org/packages/psycho/versions/0.6.1/topics/dprime
# http://wise.cgu.edu/wise-tutorials/tutorial-signal-detection-theory/signal-detection-d-defined-2/
data_dprime <- data_nogoT %>%
  group_by(Participant.Private.ID) %>%
```

```
  summarise(dp = dprime(n_hit = sum(Hit, na.rm = TRUE),
                        n_fa = sum(FA, na.rm = TRUE),
                        n_miss = sum(Miss, na.rm = TRUE),
                        n_cr = sum(CR, na.rm = TRUE),
                        n_targets = 300,
                        n_distractors = 100))
```

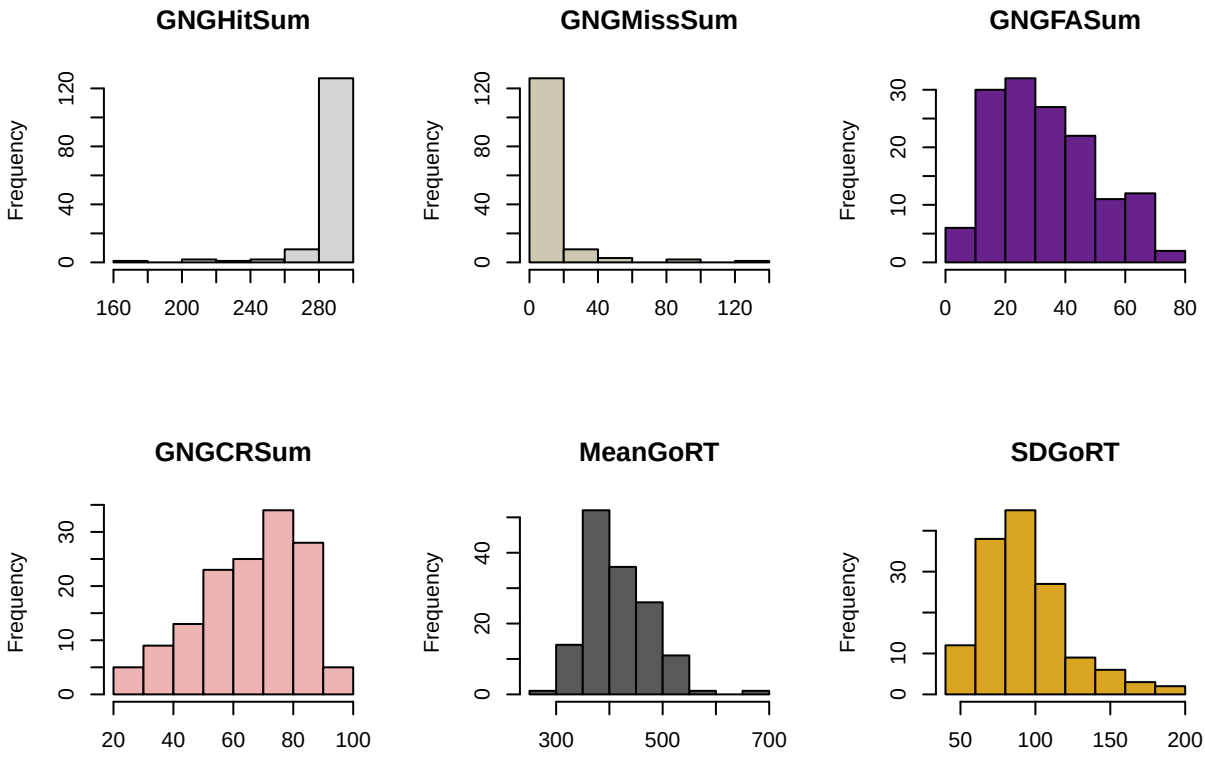
```
## `summarise()` has grouped output by 'Participant.Private.ID'. You can override
## using the `.groups` argument.
```

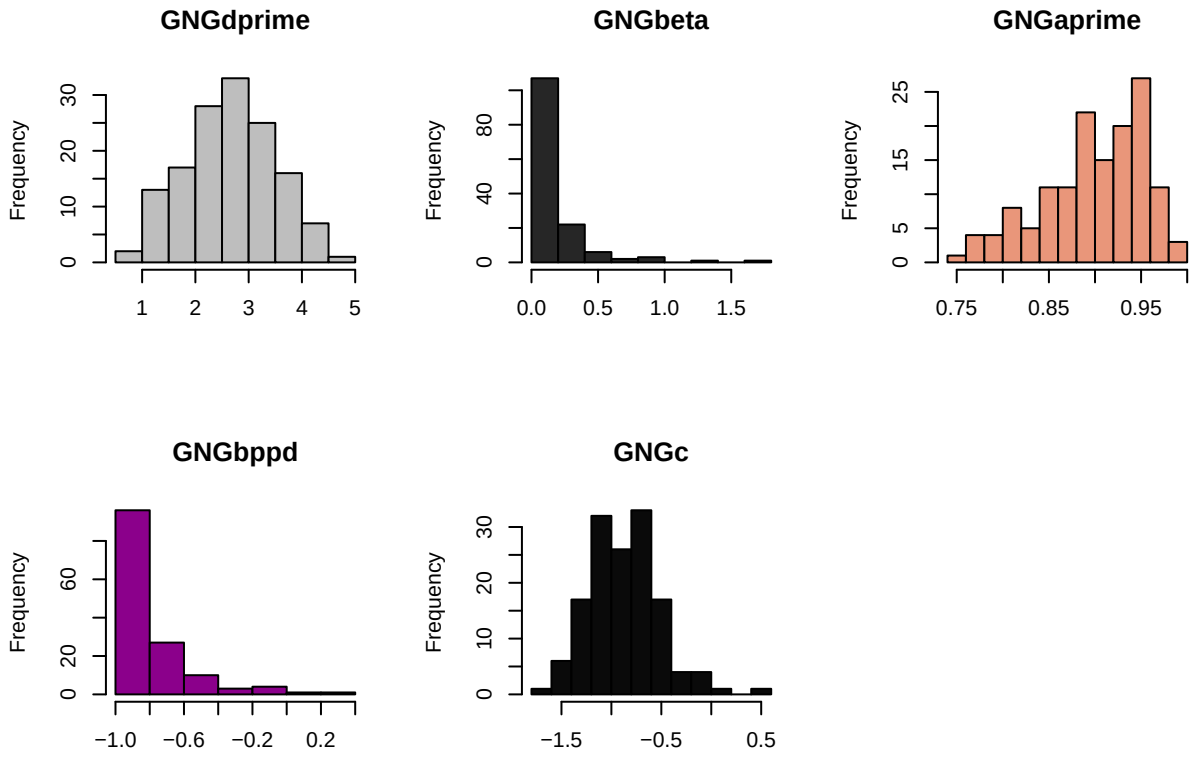
```
# Add a row to specify what the five different values given mean
data_dprime$value <- rep_len(c("GNGdprime", "GNGbeta", "GNGaprime", "GNGbppd", "GNGc"),
                             length.out = nrow(data_dprime))

# Into wide format
data_dprime <- data_dprime %>%
  pivot_wider(names_from = value, values_from = dp)
data_GNGdprime <- as.data.frame(data_dprime)

# Create a sum score of each response category for each participant
data_nogoT_final <- data_nogoT %>%
  select(Participant.Private.ID, Hit, Miss, FA, CR) %>%
  group_by(Participant.Private.ID) %>%
  summarise(GNGHitSum = sum(Hit, na.rm = TRUE),
            GNGMissSum = sum(Miss, na.rm = TRUE),
            GNGFASum = sum(FA, na.rm = TRUE),
            GNGCRSum = sum(CR, na.rm = TRUE))

# Combine the dataframes (keep all rows)
data_nogoT_final <- merge(data_nogoT_final, data_GNGdprime,
                          by = "Participant.Private.ID", all = TRUE)
data_nogoT_final <- merge(data_nogoT_final, data_go_RT,
                          by = "Participant.Private.ID", all = TRUE)

# Exclude participants with invalid responses
data_ConsentValidity <- data_Q_total %>%
  select(Participant.Private.ID, Language, Validity.quantised)

# Final dataset
data_nogoT_final <- merge(data_nogoT_final, data_ConsentValidity,
                          by = "Participant.Private.ID")

# Exclude participants according to validity + language criteria; exclude the participants who stopped
data_nogoT_final <- data_nogoT_final %>%
  filter(Validity.quantised %in% c(1, NA, "") & Language %in% c("Sujuva / Ã¤idinkieli", "Keskitaso / kes
  filter(!Participant.Private.ID %in% c("5201242", "4886650", "5117494", "5282634", "5500754", "5552405
  filter(!(GNGHitSum == 0 & GNGMissSum == 0 & GNGFASum == 0 & GNGCRSum == 0))

# Final set of columns
data_nogoT_final <- data_nogoT_final %>%
  select(Participant.Private.ID, GNGHitSum, GNGMissSum, GNGFASum, GNGCRSum, MeanGoRT, SDGoRT, GNGdprime

# Change columns into numeric
```

```r
nogo <- c("GNGHitSum", "GNGMissSum", "GNGFASum", "GNGCRSum", "MeanGoRT", "SDGoRT", "GNGdprime", "GNGbeta
for (i in 1:length(nogo)) {
  data_nogoT_final[, nogo[i]] <- as.numeric(data_nogoT_final[, nogo[i]])
}

# Describe
describe(data_nogoT_final[, nogo])
```

```
##             vars   n   mean    sd median trimmed   mad    min    max  range
## GNGHitSum      1 142 290.90 18.17 296.00  294.96  4.45 161.00 300.00 139.00
## GNGMissSum     2 142   9.10 18.17   4.00    5.04  4.45   0.00 139.00 139.00
## GNGFASum       3 142  34.29 17.13  32.00   33.12 19.27   4.00  78.00  74.00
## GNGCRSum       4 142  65.71 17.13  68.00   66.88 19.27  22.00  96.00  74.00
## MeanGoRT       5 142 413.79 61.88 409.09  410.61 66.08 290.59 655.50 364.92
## SDGoRT         6 142  93.25 27.90  86.95   90.10 25.26  49.84 182.26 132.42
## GNGdprime      7 142   2.63  0.83   2.68    2.65  0.83   0.88   4.64   3.76
## GNGbeta        8 142   0.18  0.23   0.11    0.13  0.11   0.01   1.61   1.60
## GNGaprime      9 142   0.90  0.05   0.91    0.90  0.06   0.74   0.99   0.25
## GNGbppd       10 142  -0.81  0.24  -0.88   -0.86  0.14  -1.00   0.30   1.30
## GNGc          11 142  -0.87  0.35  -0.91   -0.88  0.31  -1.62   0.44   2.06
##              skew kurtosis   se
## GNGHitSum  -4.35    22.72 1.52
## GNGMissSum  4.35    22.72 1.52
## GNGFASum    0.50    -0.60 1.44
## GNGCRSum   -0.50    -0.60 1.44
## MeanGoRT    0.67     0.59 5.19
## SDGoRT      1.06     0.98 2.34
## GNGdprime  -0.08    -0.65 0.07
## GNGbeta     3.22    13.27 0.02
## GNGaprime  -0.66    -0.26 0.00
## GNGbppd     2.16     5.38 0.02
## GNGc        0.58     0.98 0.03
```

```r
par(mfrow = c(2, 3))
for (i in 1:length(nogo)) {
  hist(data_nogoT_final[, nogo[i]],
       main = colnames(data_nogoT_final[nogo[i]]),
       col = sample(colors(), 1),
       xlab = "")
}
```


GNGHitSum
Frequency
0
40
80
120
160
200
240
280
GNGMissSum
Frequency
0
40
80
120
0
40
80
120
GNGFASum
Frequency
0
10
20
30
0
20
40
60
80
GNGCRSum
Frequency
0
10
20
30
20
40
60
80
100
MeanGoRT
Frequency
0
20
40
300
500
700
SDGoRT
Frequency
0
10
30
50
100
150
200



Navon

Trial-level exclusion criteria: * The reaction time for each row is replaced with missing values (NAs) if it is equal to or lower than 150ms

Participant-level exclusion criteria: * A participant's data for the Navon task is replaced with missing values (NAs) if their accuracy (% of correct trials out of all trials) was equal to or lower than 60% *or* if the ratio of the most frequent response to all responses is equal to or higher than .95.

Data processing: for each participant, we calculate the global-local precedence index (bias toward a global processing level), and global-to-local interference index (positive values indicate the extent to which the bias toward global stimuli interferes with processing local information). * Global-local precedence index: Standardized mean difference (cohen's d) in RT between global and local judgments on consistent trials only * Global-to-local interference index: Standardized mean difference (cohen's d) in RT between inconsistent and consistent trials in local condition only These are described and visualized.

```
# Omit experiment-general columns & practice (etc.) rows
data_NavonT <- data_NavonT %>%
  select(Participant.Private.ID, Spreadsheet:Image) %>%
  filter(display %in% c("task1", "task2")) %>%
  filter(Screen.Name == "Screen 3")

# Create a column for consistency
data_NavonT <- data_NavonT %>%
  mutate(Consistency = case_when((Image == "bigHsmallH.png" | Image == "bigSsmallS.png") ~ "Consistent"
                                 (Image == "bigHsmallS.png" | Image == "bigSsmallH.png") ~ "Inconsisten
```

```r
data_NavonT$Consistency <- as.factor(data_NavonT$Consistency)
data_NavonT$display <- as.factor(data_NavonT$display)

# Accuracy for each task and consistency
Accuracy_Navon <- data_NavonT %>%
  group_by(Consistency, display) %>%
  summarise(Accuracy_cond = sum(Correct, na.rm = TRUE) / (sum(Correct, na.rm = TRUE) + sum(Incorrect, na
```

```
## `summarise()` has grouped output by 'Consistency'. You can override using the
## `.groups` argument.
```

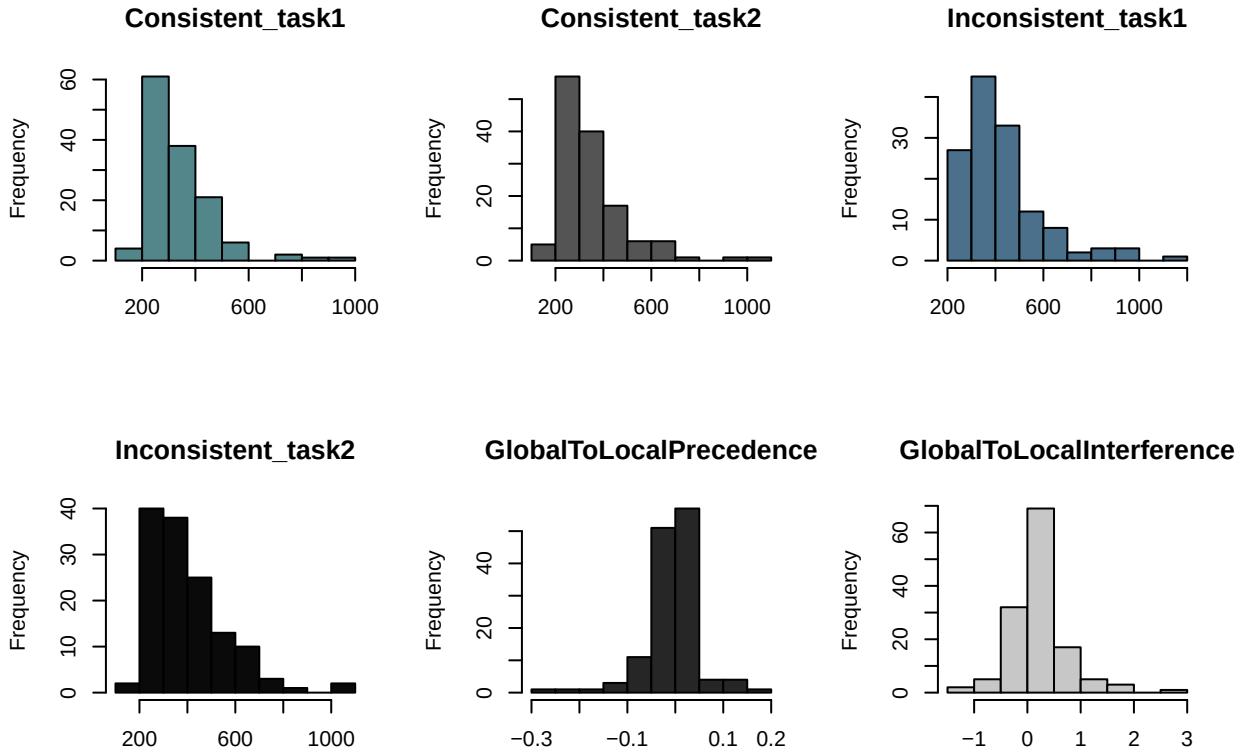
```r
# Exclusion Criteria: Trial-Level
  # RT <= 150ms
data_NavonT$Reaction.Time[which(data_NavonT$Reaction.Time <= 150)] <- NA

# Exclusion Criteria: Participant-Level
  # <= 60% accuracy
data_NavonT <- data_NavonT %>%
  group_by(Participant.Private.ID) %>%
  mutate(Accuracy = sum(Correct, na.rm = TRUE) / (sum(Correct, na.rm = TRUE) + sum(Incorrect, na.rm = TR
  mutate(Navon_omit = case_when(Accuracy > .60 ~ 0,
                                Accuracy <= .60 ~ 1))

# Omit data based on Navon_omit
data_NavonT$Reaction.Time[which(data_NavonT$Navon_omit == 1)] <- NA

# Subset to only look at correct answers
data_NavonT <- data_NavonT %>%
  filter(Correct == "1")

# Ungroup df
data_NavonT <- ungroup(data_NavonT)

# Global SD 1 (consistent trials only)
sd_Global_con <- data_NavonT %>%
  filter((Consistency == "Consistent") & display == "task1") %>%
  summarise(sd(Reaction.Time, na.rm = TRUE))
sd_Global_con <- sd_Global_con[[1]]

# Local SD 1 (consistent trials only)
sd_Local_con <- data_NavonT %>%
  filter((Consistency == "Consistent") & display == "task2") %>%
  summarise(sd(Reaction.Time, na.rm = TRUE))
sd_Local_con <- sd_Local_con[[1]]

# Local SD 2 (inconsistent trials only)
sd_Local_incon <- data_NavonT %>%
  filter((Consistency == "Inconsistent") & display == "task2") %>%
  summarise(sd(Reaction.Time, na.rm = TRUE))
sd_Local_incon <- sd_Local_incon[[1]]

# Calculate pooled SD
# https://www.statisticshowto.com/pooled-standard-deviation/
```

```r
# Pooled SD consistent (local and global)
pooled_SD_con <- sqrt((sd_Global_con^2 + sd_Local_con^2)/2)

# Pooled SD local (consistent and inconsistent)
pooled_SD_local <- sqrt((sd_Local_incon^2 + sd_Local_con^2)/2)

# Final form of the Navon data
  # Select relevant columns and transform the data so columns reflect mean reaction times in each of th
data_NavonT_final <- data_NavonT %>%
  select(Participant.Private.ID, Consistency, display, Reaction.Time) %>%
  group_by(Participant.Private.ID, Consistency, display) %>%
  summarise(NavonReactionTimeMean = mean(Reaction.Time, na.rm = TRUE),
            .groups = "keep") %>%
  pivot_wider(names_from = c(Consistency, display),
              values_from = NavonReactionTimeMean)

# Global-local precedence index: Standardized mean difference (cohen's d) in RT between global and loca
data_NavonT_final <- data_NavonT_final %>%
  mutate(GlobalToLocalPrecedence = ((Consistent_task1 - Consistent_task2) / as.numeric(pooled_SD_con)))

# Global-to-local interference index: Standardized mean difference (cohen's d) in RT between inconsiste
data_NavonT_final <- data_NavonT_final %>%
  mutate(GlobalToLocalInterference = ((Inconsistent_task1 - Consistent_task2) / as.numeric(pooled_SD_lo

# Exclude nonvalid participants
data_NavonT_final <- merge(data_NavonT_final, data_ConsentValidity,
                           by = "Participant.Private.ID")

# Validity/languege filter + get rid of participants who stopped mid-task + select relevant columns
data_NavonT_final <- data_NavonT_final %>%
  filter(Validity.quantised %in% c(1, NA, "") & Language %in% c("Sujuva / Ã¤idinkieli", "Keskitaso / ke
  filter(Participant.Private.ID != "4907987" & Participant.Private.ID != "4960307" & Participant.Privat
  select(Participant.Private.ID, Consistent_task1, Consistent_task2, Inconsistent_task1, Inconsistent_ta

# Two participants' reaction times were replaced with NAs due to their accuracy being below 60%; this p
data_NavonT_final <- data_NavonT_final %>%
  filter(Participant.Private.ID != "5608075" & Participant.Private.ID != "5411218")

# Describe
Navon <- c("Consistent_task1", "Consistent_task2", "Inconsistent_task1", "Inconsistent_task2", "GlobalT
describe(data_NavonT_final[, Navon])
```

```
##                            vars   n   mean     sd median trimmed    mad    min
## Consistent_task1              1 134 339.25 125.81 304.47  321.95  91.63 183.15
## Consistent_task2              2 134 350.40 143.03 323.65  327.71  97.68 186.50
## Inconsistent_task1            3 134 424.62 168.62 387.51  399.77 112.44 200.26
## Inconsistent_task2            4 134 399.46 162.15 352.67  377.51 143.82 194.45
## GlobalToLocalPrecedence       5 134  -0.01   0.06   0.00    0.00   0.03  -0.28
## GlobalToLocalInterference     6 134   0.23   0.53   0.13    0.19   0.33  -1.44
##                              max  range  skew kurtosis    se
## Consistent_task1          955.31 772.16  2.05     6.01 10.87
## Consistent_task2         1088.44 901.94  2.16     6.72 12.36
## Inconsistent_task1       1190.92 990.65  1.72     3.77 14.57
```

```
## Inconsistent_task2         1078.33 883.89  1.49     2.95 14.01
## GlobalToLocalPrecedence       0.19   0.48 -0.98     6.72  0.00
## GlobalToLocalInterference     2.98   4.42  1.19     5.91  0.05
```

```r
par(mfrow = c(2, 3))
for (i in 1:length(Navon)) {
  hist(data_NavonT_final[, Navon[i]],
       main = colnames(data_NavonT_final[Navon[i]]),
       col = sample(colors(), 1),
       xlab = "")
}
```



Necker

Exclusion criteria: none for now

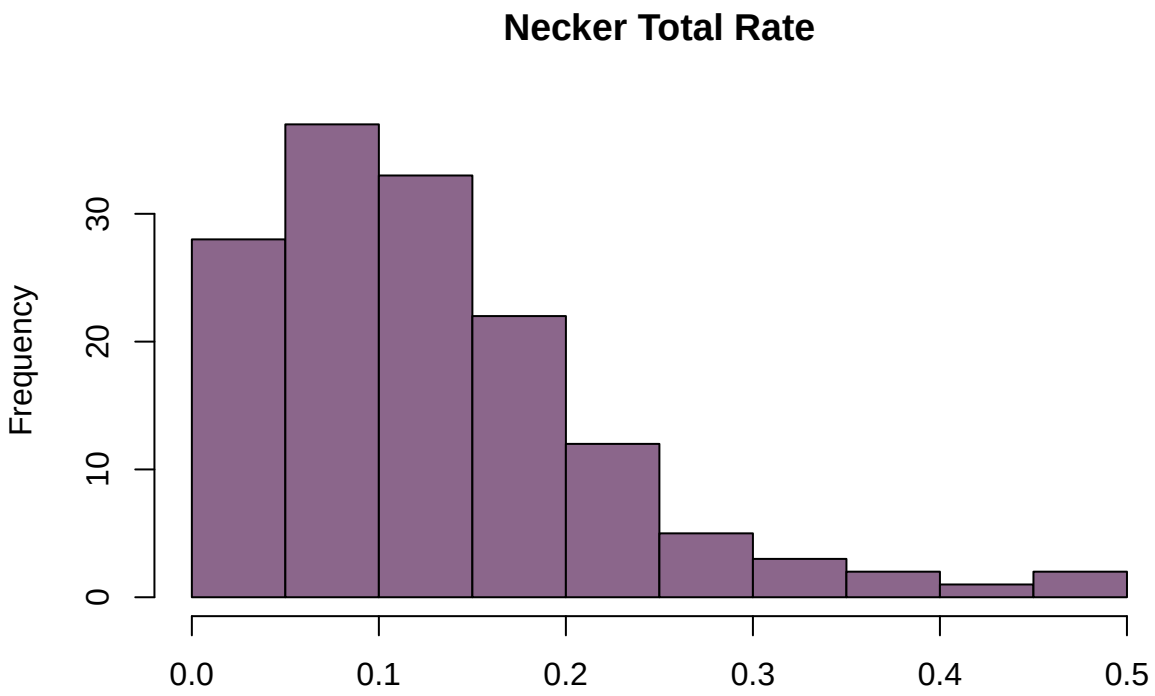
Data processing: for each participant, we calculate the average rate of space bar hits (switches the participant experienced) per second.

```r
# Omit irrelevant rows and columns
data_NeckerT <- data_NeckerT %>%
  select(Participant.Private.ID, Spreadsheet:Image) %>%
  filter(display %in% c("Trial 1", "Trial 2"))

# Calculate sum score for how many times space bar was hit in total
```

```r
data_NeckerT_final <- data_NeckerT %>%
  select(Participant.Private.ID, Response) %>%
  group_by(Participant.Private.ID) %>%
  summarise(NeckerCountTotal = sum(Response == "space", na.rm = TRUE))

# Calculate switches per second
data_NeckerT_final <- data_NeckerT_final %>%
  mutate(NeckerTotalRate = NeckerCountTotal/60)

# Merge to exclude participants
data_NeckerT_final <- merge(data_NeckerT_final, data_ConsentValidity,
                            by = "Participant.Private.ID")

# Subset by validity and consent info + omit the irrelevant columns
data_NeckerT_final <- data_NeckerT_final %>%
  filter(Validity.quantised %in% c("1", NA, "") & Language %in% c("Sujuva / Ã¤idinkieli", "Keskitaso / I
  filter(Participant.Private.ID != "5385143") %>%
  select(Participant.Private.ID, NeckerCountTotal, NeckerTotalRate)

# Describe
describe(data_NeckerT_final$NeckerTotalRate)
```

```
##    vars   n mean  sd median trimmed  mad min  max range skew kurtosis   se
## X1    1 145 0.14 0.1   0.12    0.12 0.07   0 0.48  0.48 1.25     1.97 0.01
```

```r
hist(data_NeckerT_final$NeckerTotalRate,
       main = "Necker Total Rate",
       col = sample(colors(), 1),
       xlab = "")
```



Unusual Uses Task

Exclusion criteria: None.

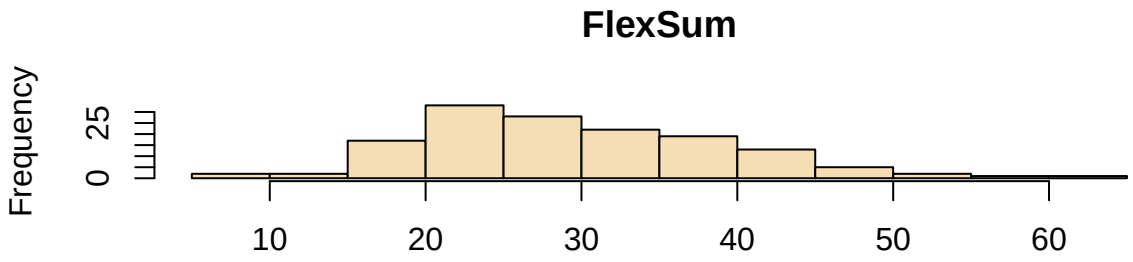
Data processing: for each participant, we calculate the average of the fluency sum scores (ratings by two raters) and the flexibility sum scores (ratings by two raters). The fluency/flexibility sum scores capture the number of answers across all trials that were classified as fluent/flexible. * Fluency is a measure of how many unique unusual uses a ppt can think of for each item. * Flexibility score refers to the uniqueness of the functional categories of items, i.e. a participant gets two point for using a shoe as a doorstop and a flowerpot but only one point for a houseplant pot and a bonsai tree pot (same functional use).

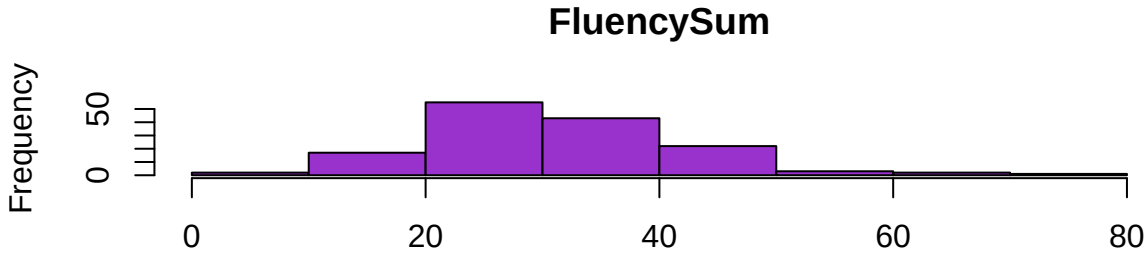
```
# Load data
data_UUT <- read.csv("UUT_scoring_COMBINED_REVISED_Rcsv.csv", sep = ";")

# Rename participant ID column
names(data_UUT)[1] <- "Participant.Private.ID"

# Select relevant columns + calculate the fluency and flexibility sum score for each participant per ra
data_UUT <- data_UUT %>%
  select(Participant.Private.ID, S.Fluency, S.Flex, K.Fluency, K.Flex) %>%
  group_by(Participant.Private.ID) %>%
  summarise(SFlexibilitySum = sum(S.Flex), SFluencySum = sum(S.Fluency),
            KFlexibilitySum = sum(K.Flex), KFluencySum = sum(K.Fluency))

# Calculate participant scores as a mean of the two raters' scores (one for fluency, one for flexibilit
data_UUT$FlexSum <- rowMeans(cbind(data_UUT$SFlexibilitySum,
```

```
                                    data_UUT$KFlexibilitySum),
                             na.rm = TRUE)
data_UUT$FluencySum <- rowMeans(cbind(data_UUT$SFluencySum,
                                      data_UUT$KFluencySum),
                                na.rm = TRUE)

# Check validity + extract relevant columns
data_UUT <- merge(data_UUT, data_ConsentValidity,
                  by = "Participant.Private.ID")
data_UUT <- data_UUT %>%
  filter(Validity.quantised %in% c("1", NA, "") & Language %in% c("Sujuva / Ã¤idinkieli", "Keskitaso / I
  select(Participant.Private.ID, FlexSum, FluencySum)

# Describe
uut <- c("FlexSum", "FluencySum")
describe(data_UUT[, uut])
```

```
##            vars   n  mean    sd median trimmed   mad min max range skew
## FlexSum       1 145 29.81  9.69   28.5   29.35  9.64   6  61    55 0.50
## FluencySum    2 145 31.59 10.99   30.0   30.88 10.38   7  75    68 0.85
##            kurtosis   se
## FlexSum        0.32 0.81
## FluencySum     1.56 0.91
```

```
par(mfrow = c(2, 1))
for (i in 1:length(uut)) {
  hist(data_UUT[, uut[i]],
       main = colnames(data_UUT[uut[i]]),
       col = sample(colors(), 1),
       xlab = "")
}
```





Merge task variables

Doesn't yet include the CI task!

```
# Merge dataframes
data_T_total1 <- merge(data_nogoT_final, data_NavonT_final,
                       by = "Participant.Private.ID", all = TRUE)
data_T_total2 <- merge(data_UUT, data_NeckerT_final,
                       by = "Participant.Private.ID", all = TRUE)
data_T_total <- merge(data_T_total1, data_T_total2,
                      by = "Participant.Private.ID", all = TRUE)
rm(data_T_total1, data_T_total2)

# Extract the relevant columns
data_T_sub <- data_T_total %>%
  select(Participant.Private.ID, GNGdprime, GNGbeta, MeanGoRT, SDGoRT, GlobalToLocalPrecedence, GlobalT

# Participant ID into factor
data_T_sub$Participant.Private.ID <- as.factor(data_T_sub$Participant.Private.ID)
```

Analyze Missing Data in Tasks

Participants with large amounts of missing data or suspicious patterns of missing data are removed here. It should be noted that missing data refers to data that was missing from the start *and* data that was replaced

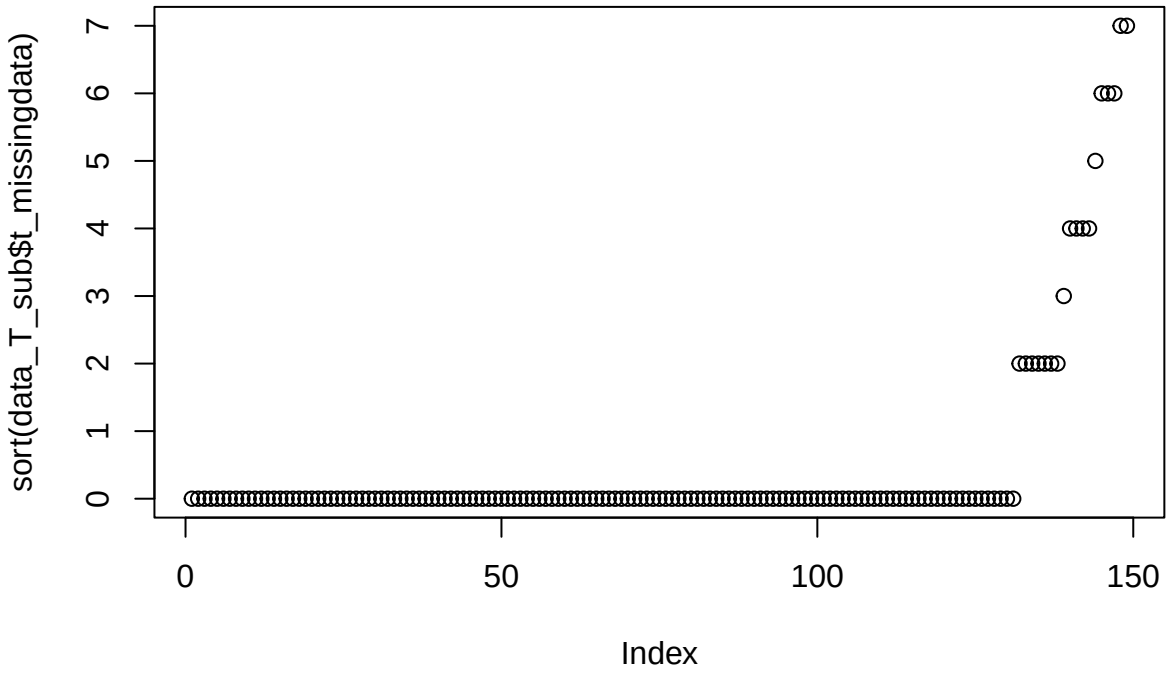
with missing values if the participant met one of the exclusion criteria. Done separately for questionnaires and tasks.

```
# Task variables
t_var <- c("GNGdprime", "GNGbeta", "MeanGoRT", "SDGoRT", "GlobalToLocalPrecedence", "GlobalToLocalInter
```

```
# Add a column for count of missing data per ppt
data_T_sub$t_missingdata <- rowSums(is.na(data_T_sub[, t_var]))
table(data_T_sub$t_missingdata)
```

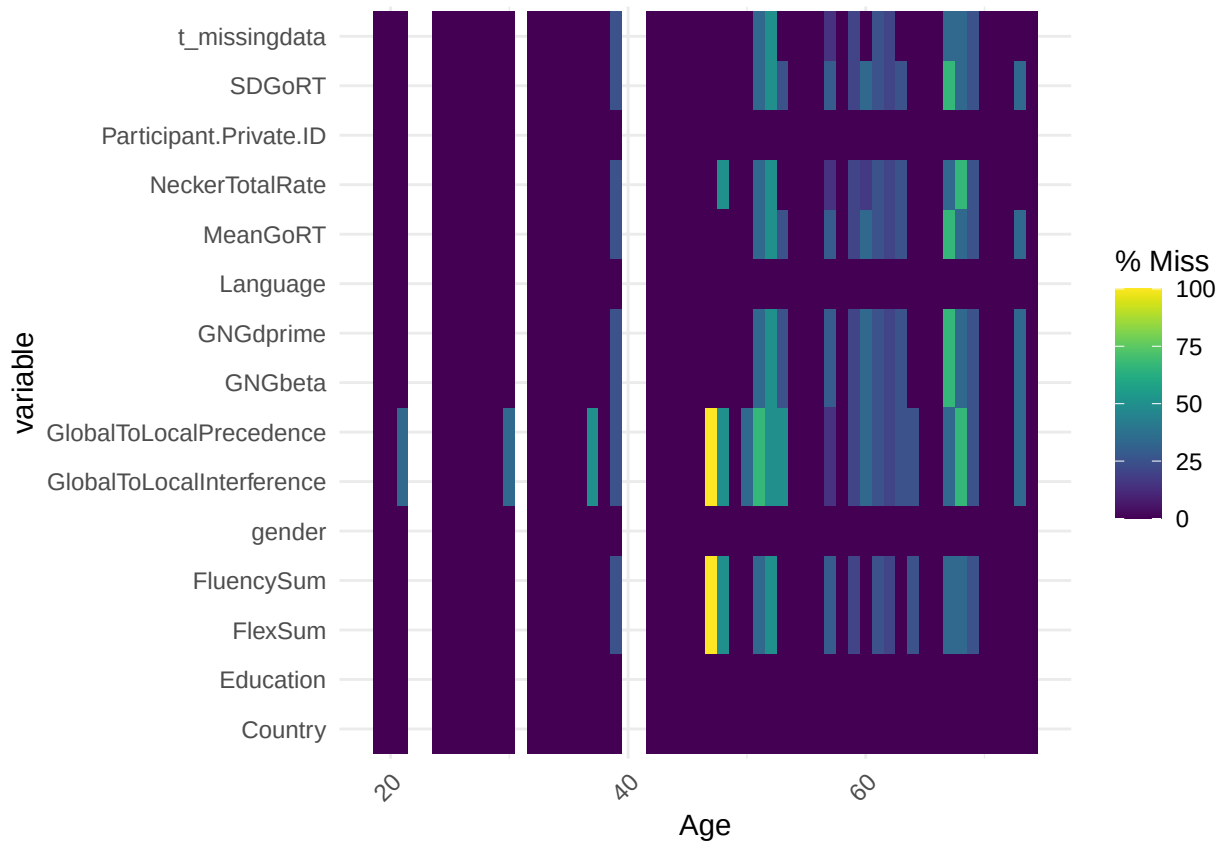
```
##
##   0   2   3   4   5   6   7
## 131   7   1   4   1   3   2
```

```
plot(sort(data_T_sub$t_missingdata))
```




```
# Merge demographic information to investigate missingness
# NOTE: what to do about the age correlation?
data_T_all <- data_Q_total %>%
  select(Participant.Private.ID, gender, Age, Country, Language, Education)
data_T_sub <- merge(data_T_all, data_T_sub,
                    by = "Participant.Private.ID", all = TRUE)
gg_miss_fct(x = data_T_sub, fct = Age)
```

```
## Warning: Removed 15 rows containing missing values (geom_tile).
```




```
cor.test(data_T_sub$Age, data_T_sub$t_missingdata,
         method = "spearman", exact = FALSE)
```

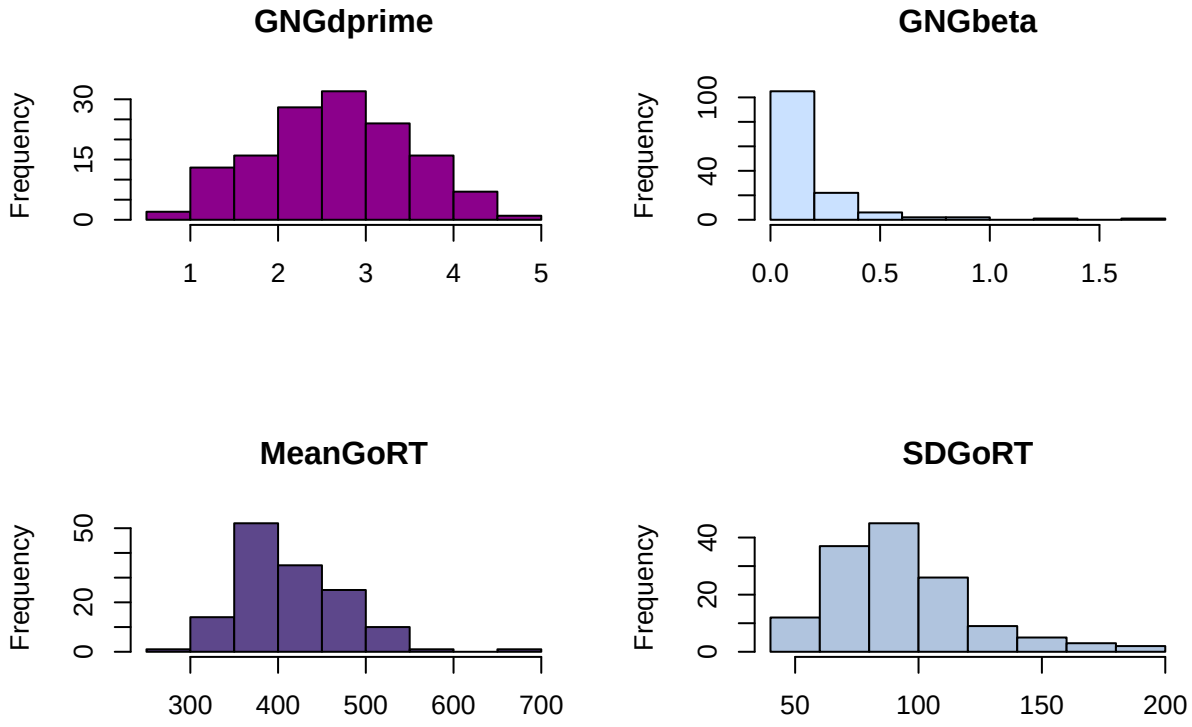
```
##
##  Spearman's rank correlation rho
##
## data:  data_T_sub$Age and data_T_sub$t_missingdata
## S = 485253, p-value = 0.1456
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
##       rho
## 0.1198027
```

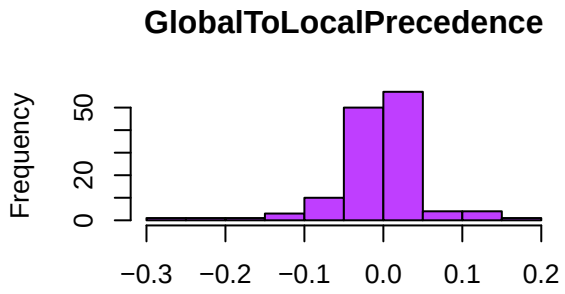
```
# Remove participants with 4 or more missing task variables
data_T_sub <- data_T_sub %>%
  group_by(Participant.Private.ID) %>%
  filter(t_missingdata < 4)

# Omit irrelevant columns
data_T_sub <- data_T_sub %>%
  select(-t_missingdata)
```

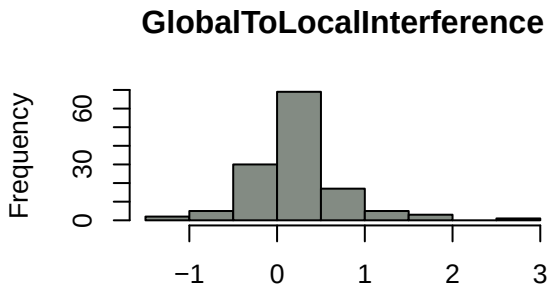
Check Task Distributions and Correlations

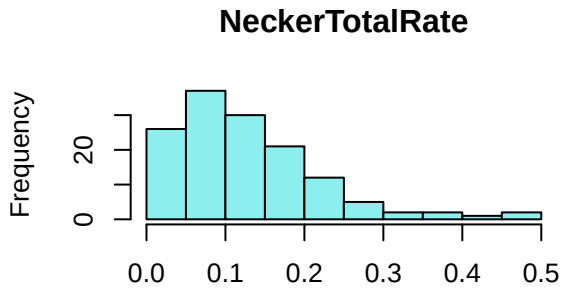
This chunk checks distributions of task variables and correlations between them: * If distributions are highly skewed (>=1), transformations are applied to decrease skewness * If two variables are highly correlated, one of them is dropped (Eisenberg et al.) (r>=.85)

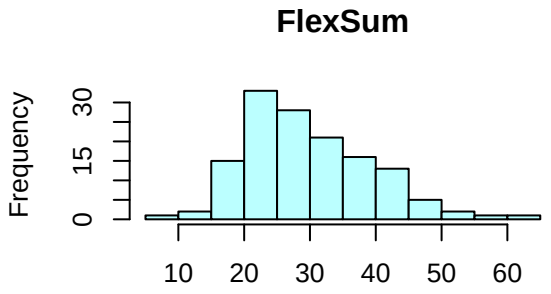
```
# Normality of variables: visualize with histograms
data_T_sub <- as.data.frame(data_T_sub)
par(mfrow = c(2, 2))
for (i in 1:length(t_var)) {
  hist(data_T_sub[, t_var[i]],
       main = colnames(data_T_sub[t_var[i]]),
       col = sample(colors(), 1),
       xlab = "")
}
```












```
# Describe: mean, skew, kurtosis
describe(data_T_sub[, t_var])[c("mean", "skew", "kurtosis")]
```

```
##                              mean  skew kurtosis
## GNGdprime                    2.64 -0.07    -0.65
## GNGbeta                      0.17  3.38    14.87
## MeanGoRT                   412.71  0.70     0.68
## SDGoRT                      92.83  1.10     1.12
## GlobalToLocalPrecedence      0.00 -1.02     6.85
## GlobalToLocalInterference    0.23  1.16     5.82
## NeckerTotalRate              0.14  1.28     2.14
## FlexSum                     29.95  0.60     0.42
## FluencySum                  31.72  0.98     1.79
```

```
# Transforming all variables with absolute skew > 1
data_T_sub$GNGbetalog <- log10(data_T_sub$GNGbeta)
data_T_sub$SDGoRTlog <- log10(data_T_sub$SDGoRT)
data_T_sub$GlobalToLocalInterferencelog <- log10(data_T_sub$GlobalToLocalInterference)
```

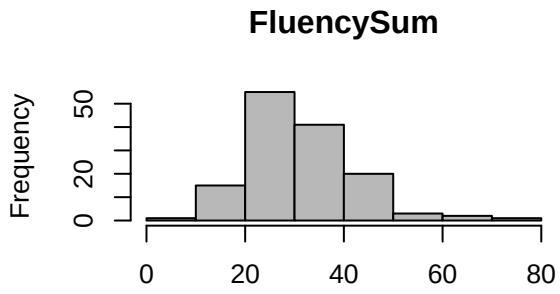
```
## Warning: NaNs produced
```

```
data_T_sub$NeckerTotalRatelog <- log10(data_T_sub$NeckerTotalRate + 1) # added 1 because log10 cannot h
data_T_sub$GlobalToLocalPrecedencelog <- log10(max(data_T_sub$GlobalToLocalPrecedence + 1, na.rm = TRUE)
```

```r
# Task variables: transformed
t_var_log <- c("GNGdprime", "GNGbetalog", "MeanGoRT", "SDGoRTlog", "GlobalToLocalPrecedencelog", "Global
# Describe
describe(data_T_sub[, t_var_log])[c("mean", "skew", "kurtosis")]
```

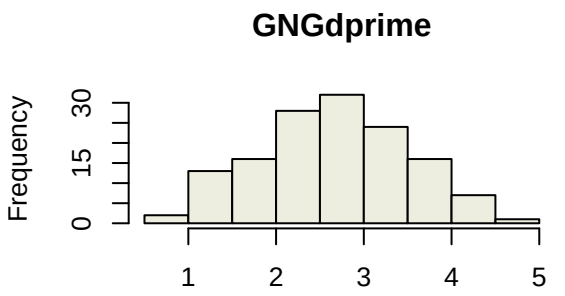
```
##                                 mean  skew kurtosis
## GNGdprime                       2.64 -0.07    -0.65
## GNGbetalog                     -1.02 -0.04    -0.69
## MeanGoRT                      412.71  0.70     0.68
## SDGoRTlog                       1.95  0.38    -0.25
## GlobalToLocalPrecedencelog      0.08  0.49     5.83
## GlobalToLocalInterferencelog -0.58 -0.25    -0.27
## NeckerTotalRatelog              0.05  1.00     1.26
## FlexSum                        29.95  0.60     0.42
## FluencySum                     31.72  0.98     1.79
```

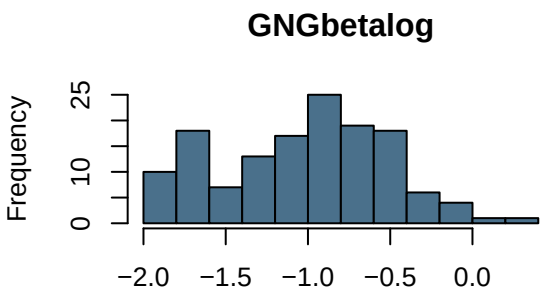
```r
# Visualize
par(mfrow = c(2, 2))
```

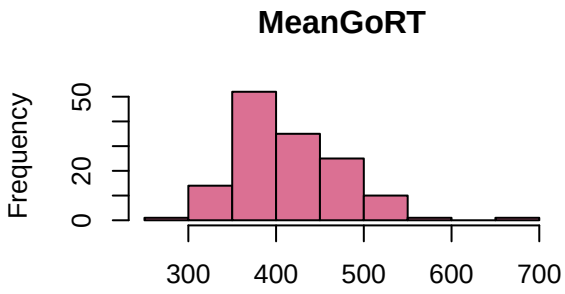


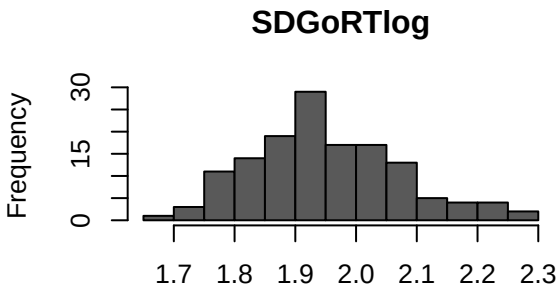

```r
for (i in 1:length(t_var_log)) {
  hist(data_T_sub[, t_var_log[i]],
       main = colnames(data_T_sub[t_var_log[i]]),
       col = sample(colors(), 1),
```

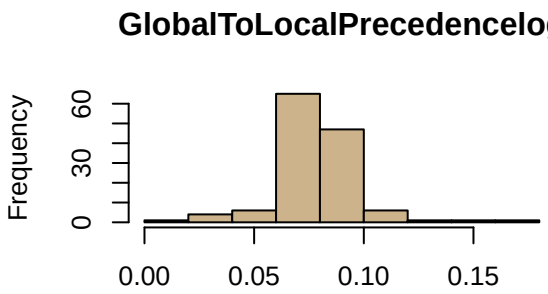
```
        xlab = "")
}
```

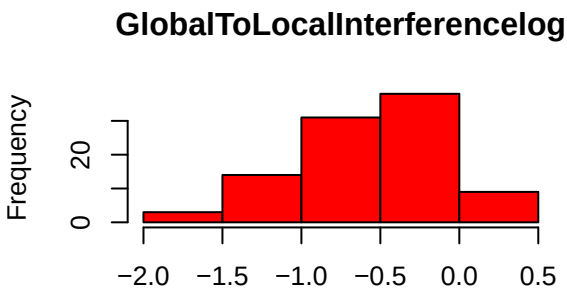

GNGdprime
Frequency
0
15
30
1
2
3
4
5

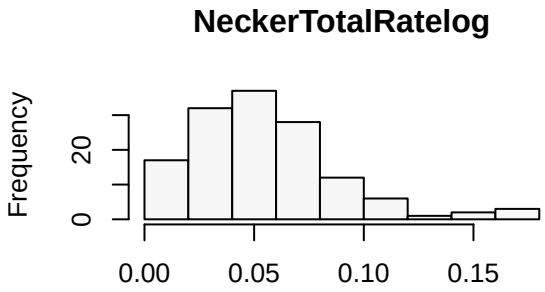

GNGbetalog
Frequency
0
10
25
−2.0
−1.5
−1.0
−0.5
0.0

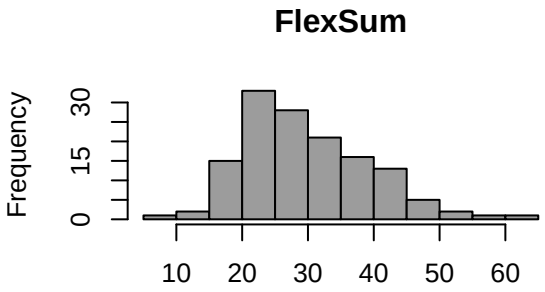

MeanGoRT
Frequency
0
20
50
300
400
500
600
700


SDGoRTlog
Frequency
0
15
30
1.7
1.8
1.9
2.0
2.1
2.2
2.3










```
# Correlation matrix
round(cor(data_T_sub[, t_var_log], method = "pearson",
          use = "pairwise.complete.obs"), 2)
```

```
##                                GNGdprime GNGbetalog MeanGoRT SDGoRTlog
## GNGdprime                           1.00      -0.69     0.14     -0.39
## GNGbetalog                         -0.69       1.00     0.39      0.68
## MeanGoRT                            0.14       0.39     1.00      0.67
## SDGoRTlog                          -0.39       0.68     0.67      1.00
## GlobalToLocalPrecedencelog          0.04      -0.09    -0.15     -0.16
## GlobalToLocalInterferencelog       -0.17       0.35     0.24      0.31
## NeckerTotalRatelog                  0.07      -0.21    -0.16     -0.19
## FlexSum                             0.19      -0.12     0.00     -0.08
## FluencySum                          0.20      -0.15    -0.02     -0.09
##                                GlobalToLocalPrecedencelog
## GNGdprime                                            0.04
## GNGbetalog                                          -0.09
## MeanGoRT                                            -0.15
## SDGoRTlog                                           -0.16
## GlobalToLocalPrecedencelog                           1.00
## GlobalToLocalInterferencelog                        -0.48
## NeckerTotalRatelog                                   0.00
## FlexSum                                             -0.05
## FluencySum                                          -0.03
##                                GlobalToLocalInterferencelog NeckerTotalRatelog
## GNGdprime                                             -0.17                0.07
```

```
## GNGbetalog                                    0.35             -0.21
## MeanGoRT                                      0.24             -0.16
## SDGoRTlog                                     0.31             -0.19
## GlobalToLocalPrecedencelog                   -0.48              0.00
## GlobalToLocalInterferencelog                  1.00             -0.14
## NeckerTotalRatelog                           -0.14              1.00
## FlexSum                                       0.07              0.09
## FluencySum                                    0.07              0.10
##                               FlexSum FluencySum
## GNGdprime                        0.19       0.20
## GNGbetalog                      -0.12      -0.15
## MeanGoRT                         0.00      -0.02
## SDGoRTlog                       -0.08      -0.09
## GlobalToLocalPrecedencelog      -0.05      -0.03
## GlobalToLocalInterferencelog     0.07       0.07
## NeckerTotalRatelog               0.09       0.10
## FlexSum                          1.00       0.98
## FluencySum                       0.98       1.00
```

```
abs(round(cor(data_T_sub[, t_var_log], method = "pearson",
              use = "pairwise.complete.obs"), 2)) > .85
```

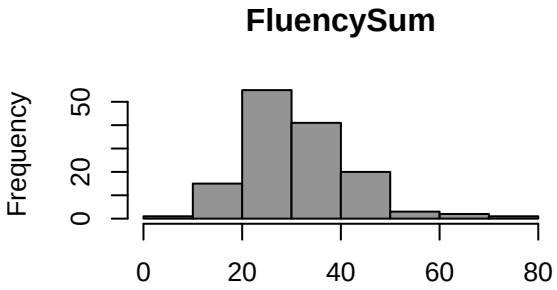
```
##                               GNGdprime GNGbetalog MeanGoRT SDGoRTlog
## GNGdprime                          TRUE      FALSE    FALSE     FALSE
## GNGbetalog                        FALSE       TRUE    FALSE     FALSE
## MeanGoRT                          FALSE      FALSE     TRUE     FALSE
## SDGoRTlog                         FALSE      FALSE    FALSE      TRUE
## GlobalToLocalPrecedencelog        FALSE      FALSE    FALSE     FALSE
## GlobalToLocalInterferencelog      FALSE      FALSE    FALSE     FALSE
## NeckerTotalRatelog                FALSE      FALSE    FALSE     FALSE
## FlexSum                           FALSE      FALSE    FALSE     FALSE
## FluencySum                        FALSE      FALSE    FALSE     FALSE
##                               GlobalToLocalPrecedencelog
## GNGdprime                                          FALSE
## GNGbetalog                                         FALSE
## MeanGoRT                                           FALSE
## SDGoRTlog                                          FALSE
## GlobalToLocalPrecedencelog                          TRUE
## GlobalToLocalInterferencelog                       FALSE
## NeckerTotalRatelog                                 FALSE
## FlexSum                                            FALSE
## FluencySum                                         FALSE
##                               GlobalToLocalInterferencelog NeckerTotalRatelog
## GNGdprime                                            FALSE              FALSE
## GNGbetalog                                           FALSE              FALSE
## MeanGoRT                                             FALSE              FALSE
## SDGoRTlog                                            FALSE              FALSE
## GlobalToLocalPrecedencelog                           FALSE              FALSE
## GlobalToLocalInterferencelog                          TRUE              FALSE
## NeckerTotalRatelog                                   FALSE               TRUE
## FlexSum                                              FALSE              FALSE
## FluencySum                                           FALSE              FALSE
##                               FlexSum FluencySum
## GNGdprime                       FALSE      FALSE
```

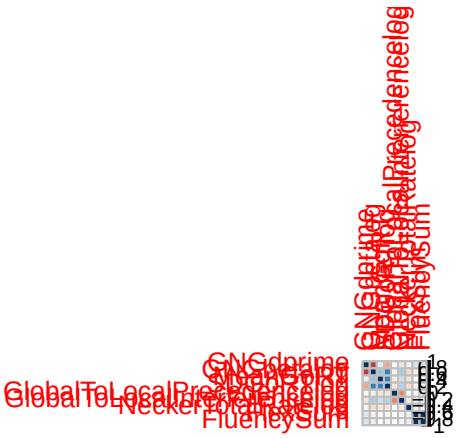
```
## GNGbetalog                      FALSE      FALSE
## MeanGoRT                        FALSE      FALSE
## SDGoRTlog                       FALSE      FALSE
## GlobalToLocalPrecedencelog      FALSE      FALSE
## GlobalToLocalInterferencelog    FALSE      FALSE
## NeckerTotalRatelog              FALSE      FALSE
## FlexSum                          TRUE       TRUE
## FluencySum                       TRUE       TRUE
```

```
corrplot(cor(data_T_sub[, t_var_log], method = "pearson",
             use = "pairwise.complete.obs"))
```

```
## Warning in corrplot(cor(data_T_sub[, t_var_log], method = "pearson", use =
## "pairwise.complete.obs")): Not been able to calculate text margin, please try
## again with a clean new empty window using {plot.new(); dev.off()} or reduce
## tl.cex
```

```
# Exclude FluencySum as it correlates highly with FlexSum; only include log-transformed variables where
data_T_sub <- data_T_sub %>%
  select(Participant.Private.ID, GNGdprime, GNGbetalog, MeanGoRT, SDGoRT, GlobalToLocalInterferencelog,
```





Format the Task and Questionnaire Dataframes

```
data_all_sub_cleaned <- merge(data_Q_sub, data_T_sub,
                              by = "Participant.Private.ID",
                              all = TRUE)

# Ungroup the dataframe
ungroup(data_all_sub_cleaned)
```

```
##     Participant.Private.ID AMean CMean EMean ESMean OMean ICuriositySum
## 1                  4831351   4.0   7.0   6.0      2   7.0            21
## 2                  4831981   4.0   5.5   4.0      3   6.0            21
## 3                  4877584   5.5   5.0   3.0      3   4.5            21
## 4                  4886552   5.0   6.5   2.0      5   5.0            20
## 5                  4886650    NA    NA    NA     NA    NA            15
## 6                  4886771   4.5   6.0   4.0      6   4.0            16
## 7                  4886773   6.5   4.0   6.0      3   6.0            23
## 8                  4886824   5.5   4.0   2.5      5   5.5            17
## 9                  4887029   4.5   6.5   5.5      7   5.0            23
## 10                 4887607   6.0   6.5   5.0      6   3.5            18
## 11                 4888071   4.5   5.5   4.0      3   7.0            19
## 12                 4891916   3.5   2.5   2.5      4   5.5            22
## 13                 4907987   6.5   5.5   3.5      6   5.0            24
## 14                 4914191   6.5   4.0   6.5      3   6.0             9
## 15                 4916307   4.5   3.5   3.5      3   5.0            17
## 16                 4921307   3.5   4.0   5.0      3   5.0            20
## 17                 4922817   6.5   3.5   3.0      4   5.5            18
## 18                 4929121   4.5   4.5   3.5      5   3.0            16
## 19                 4943807   6.5   6.0   5.5      6   6.0            22
## 20                 4945017   5.0   6.0   5.5      6   4.5            20
## 21                 4946837   2.5   7.0   2.0      6   5.5            22
## 22                 4948053   5.5   3.5   6.0      3   4.0            18
## 23                 4948269   5.5   4.5   5.5      3   4.5            17
## 24                 4959255   5.0   1.5   2.0      2   5.0            19
## 25                 4959547   5.5   2.5   7.0      3   6.0            21
## 26                 4973999   6.0   6.0   2.0      5   4.0            21
## 27                 4978306   4.0   5.0   3.5      4   7.0            25
## 28                 4986367   5.0   4.0   3.0      6   5.5            21
## 29                 5110749   6.0   5.0   4.0      4   6.5            20
## 30                 5117494   4.0   4.5   2.0      5   6.5            23
## 31                 5120203   4.0   3.5   4.0      3   5.5            24
## 32                 5126869   4.0   5.0   5.0      6   6.0            23
## 33                 5130877   5.5   6.0   5.0      6   5.5            21
## 34                 5132910   3.5   4.5   5.5      1   6.5            24
## 35                 5133207   4.0   5.0   6.0      5   6.0            20
## 36                 5133586   4.0   4.0   4.0      5   4.5            20
## 37                 5156990   4.5   6.5   5.5      4   5.0            23
## 38                 5175309   6.0   6.5   3.0      2   3.5            16
## 39                 5189349   4.5   4.0   2.5      3   6.5            20
## 40                 5191578   5.5   5.5   2.5      6   4.5            22
## 41                 5192188   4.0   5.5   3.0      5   3.5            14
## 42                 5201242   2.5   4.0   4.0      2   3.5            23
```

```
## 43              5202064   4.5   4.5   6.0      4   5.5            20
## 44              5258209   5.0   3.0   5.5      3   6.0            24
## 45              5282634   3.5   4.5   5.5      4   4.5            16
## 46              5287907   6.0   5.5   5.0      6   5.0            19
## 47              5290491   3.5   5.5   5.0      4   4.0            23
## 48              5292007   3.5   6.0   3.5      2   6.0            24
## 49              5298015   1.5   2.0   1.0      1   7.0            25
## 50              5305503   5.5   5.5   3.5      6   5.5            25
## 51              5320807   4.5   5.0   6.0      5   5.5            22
## 52              5336084   4.0   3.5   6.0      6   6.0            23
## 53              5354577   7.0   5.5   5.5      6   7.0            24
## 54              5355205   4.5   4.0   5.0      5   5.0            21
## 55              5358132   4.0   4.0   5.0      2   4.0            18
## 56              5372338   4.0   5.0   2.5      6   4.5            21
## 57              5375463   5.5   6.0   6.5      6   6.5            25
## 58              5380955   1.5   5.0   3.0      2   5.5            16
## 59              5384429   6.5   4.0   5.5      4   5.5            24
## 60              5387595   4.0   6.0   5.0      5   5.0            19
## 61              5388970   5.0   3.0   2.5      4   4.5            18
## 62              5399212   6.0   5.0   5.0      4   6.0            24
## 63              5411218   3.5   2.0   4.5      4   4.0            25
## 64              5435499   4.0   3.0   3.5      5   4.5            23
## 65              5440344   3.5   5.0   6.0      6   4.5            22
## 66              5443499   4.5   4.5   2.5      7   6.0            21
## 67              5443570   7.0   1.5   3.0      2   5.5            24
## 68              5446208   5.0   3.5   3.5      6   3.0            16
## 69              5446920   4.5   3.5   6.0      3   6.5            24
## 70              5449061   5.0   3.5   1.5      6   7.0            20
## 71              5451953   6.5   4.0   6.0      4   6.5            22
## 72              5473445   5.0   3.0   1.5      6   4.5            19
## 73              5476245   5.0   5.0   3.5      6   5.0            24
## 74              5478640   3.0   5.5   3.0      2   5.0            24
## 75              5486644   6.5   6.5   4.0      6   5.0            18
## 76              5486645   4.5   6.0   6.0      5   5.5            24
## 77              5486960   5.0   4.5   4.5      7   6.0            23
## 78              5488274   4.5   5.0   4.0      5   5.5            20
## 79              5488336   5.5   3.0   3.5      2   5.5            20
## 80              5493143   5.5   4.0   5.5      2   3.5            16
## 81              5498171   5.5   3.0   6.0      3   5.5            25
## 82              5504593   6.5   4.0   4.0      5   6.0            21
## 83              5505611   6.0   2.5   3.0      1   5.0            16
## 84              5506382   3.5   3.5   2.0      5   5.5            20
## 85              5507405   4.0   5.5   5.0      5   4.5            18
## 86              5517409   6.0   4.0   5.5      4   6.5            25
## 87              5520492   5.0   3.5   4.0      6   5.0            19
## 88              5526450   1.5   5.0   4.5      3   4.5            25
## 89              5542767   6.5   4.0   6.5      6   6.5            21
## 90              5548631   4.5   5.0   2.5      5   5.5            22
## 91              5555000   5.5   4.5   2.5      7   5.0            21
## 92              5555882   4.0   6.0   1.5      5   3.0            23
## 93              5556093   6.0   5.5   5.5      6   2.0            17
## 94              5559681   6.0   5.0   5.0      6   5.0            24
## 95              5562166   5.5   5.5   5.0      5   6.0            21
## 96              5569558   3.5   6.0   7.0      6   7.0            24
```

```
## 97                5578226   6.0   6.0   3.5      6   5.0           18
## 98                5608075   5.0   4.5   4.5      5   6.0           25
## 99                5614189   6.5   2.0   5.5      7   6.0           24
## 100               5616968   4.5   2.0   7.0      1   5.5           25
## 101               5618126   5.0   4.0   3.0      4   4.0           20
## 102               5621289   5.0   3.5   2.5      4   6.0           25
## 103               5626669    NA    NA    NA     NA    NA           19
## 104               5629594   5.5   4.5   5.5      3   5.0           14
## 105               5629626   3.5   5.5   3.5      4   3.0           24
## 106               5629995   3.5   2.0   6.5      2   7.0           25
## 107               5631076   5.0   6.5   4.0      4   1.5           14
## 108               5634435   6.5   4.5   4.5      6   5.5           21
## 109               5641501   3.0   4.5   4.0      3   4.5           21
## 110               5645930   5.5   5.5   3.5      7   6.0           24
## 111               5649003   5.0   3.5   4.5      6   4.0           24
## 112               5650356   6.0   5.5   4.0      6   5.5           22
## 113               5653192   4.0   3.5   2.0      6   2.5           14
## 114               5653211   6.0   6.0   5.0      6   6.0           23
## 115               5659605   6.0   4.0   2.0      5   5.5           22
## 116               5676387   4.5   6.5   5.5      6   7.0           25
## 117               5679394   4.5   4.0   2.5      3   5.0           22
## 118               5686104   3.5   3.5   4.0      5   4.5           20
## 119               5687746   6.0   6.5   5.5      7   6.5           23
## 120               5688370   5.0   6.5   5.0      7   6.0           22
## 121               5691188   6.5   4.0   1.0      2   7.0           22
## 122               5691497   5.5   4.5   2.0      6   6.0           21
## 123               5693586   4.0   5.0   3.0      4   4.5           20
## 124               5693701   3.5   3.0   6.0      3   5.5           20
## 125               5695843   3.0   6.5   3.0      6   3.5           17
## 126               5695925   5.0   3.0   4.0      2   7.0           24
## 127               5726771   5.5   6.5   3.5      4   5.5           19
## 128               5731793    NA    NA    NA     NA    NA           20
## 129               5731864   6.0   5.5   3.5      6   5.5           24
## 130               5747184   4.5   3.0   4.0      4   5.0           23
## 131               5758326   6.0   4.5   4.0      5   4.5           23
## 132               5758833   6.0   4.5   1.5      2   6.0           NA
## 133               5767733   4.5   4.5   3.5      3   6.5           20
## 134               5774889   4.5   6.5   7.0      3   5.5           20
## 135               5791357   3.5   4.5   6.0      3   4.0           20
## 136               5798816   3.5   2.5   2.0      3   5.0           19
## 137               5805972   3.0   5.0   5.0      5   4.0           19
## 138               5808176   4.5   3.5   3.5      3   4.0           16
## 139               5818084   5.5   6.0   5.5      6   5.5           20
## 140               5828423   5.0   6.0   5.5      7   5.5           25
## 141               5842153   5.0   6.0   6.0      6   6.5           20
## 142               5844536   6.0   5.5   6.0      5   5.5           NA
## 143               5845513   4.5   6.0   4.5      2   6.0           24
## 144               5849414   4.0   5.0   6.5      5   5.0           18
## 145               5867681   4.0   6.0   5.0      6   5.5           23
## 146               5879081   3.0   5.5   2.5      5   4.5           24
## 147               5885149   5.5   5.5   4.5      4   5.0           22
## 148               5891043   6.0   5.0   4.0      5   5.0           20
## 149               5893310   3.5   5.0   3.0      4   4.5           25
## 150               5896498   6.5   6.5   5.5      6   6.0           23
```

```
##     DCuriositySum IH1Sum IH2Sum IH3Sum IH4Sum CloSum CogSum AOTSum RPSum
## 1              15     25     23     23     16     41     86     41     4
## 2              17     16     21     18     15     43     89     36     3
## 3              15     20     18     20     19     51     90     36     2
## 4              18     15     19     17     19     75     80     35     2
## 5              15     15     15     15     15     NA     63     NA     1
## 6              14     20     20     19     17     52     82     NA     3
## 7              21     15     22     21     17     52     94     43     3
## 8              13     25     19     25     21     56     71     37     3
## 9              12     19     19     23     16     34     88     34     2
## 10             16     17     20     19     22     59     79     39     2
## 11             17     18     19     20     24     49     84     43     2
## 12             17     20     21     19     17     46     82     44     3
## 13             16     18     21     25     17     49     NA     46     2
## 14              9     18     20     18     17     62     68     33     0
## 15             10     13     20     17     21     62     64     45     3
## 16             14     14     20     19     19     51     70     44     2
## 17             19     11     24     22     22     50     74     39     1
## 18             13      6     18     20     20     61     54     43     1
## 19             16     20     20     24     17     60     74     44     3
## 20             19     22     20     15     16     53     85     44     3
## 21             16     24     22     20     13     57     81     47     3
## 22             15     20     22     23     17     55     79     41     3
## 23             17     17     20     21     15     50     72     35     0
## 24             15      5     21     17     21     80     44     44     3
## 25             14     11     17     23     19     42     64     42     3
## 26             18     21     21     22     22     52     85     44     3
## 27             18     19     20     24     17     40     92     42     2
## 28             20     20     20     17     15     52     83     36     3
## 29             17     15     22     16     17     61     77     48     2
## 30             14     25     21     17     15     45     85     NA     1
## 31             18     17     25     16     20     52     88     48     3
## 32             22     18     23     25     20     57     78     43     2
## 33             14     15     20     20     16     46     85     45     3
## 34             20     11     21     21     16     54     86     45     2
## 35             18     19     20     20     19     36     78     43     1
## 36             12     19     19     21     15     47     84     35     3
## 37             19     23     24     22     19     47     89     44     4
## 38             14     11     20     20     20     72     67     34     3
## 39             18     20     23     19      9     59     90     43     4
## 40             18     15     20     20     13     66     63     42     4
## 41              8     19     19     19     15     51     62     44     2
## 42             15     13     25     16     18     60     84     43     2
## 43             15     18     18     18     18     44     92     44     3
## 44             18     18     21     18     21     28     98     44     3
## 45             16     20     17     16     17     48     69     NA     1
## 46             14     24     17     24     17     38     93     34     1
## 47             15     20     21     18     23     43     85     43     2
## 48             16     12     20     17     14     50     99     46     2
## 49             23     25     25     21     18     37     90     49     4
## 50             21     22     24     22     16     46     92     44     2
## 51             16     20     18     20     16     48     75     40     1
## 52             21     19     21     18     19     44     95     42     2
## 53             19     14     18     20     16     33    105     39     1
```

```
## 54    13    10    20    16    15    58    72    47     3
## 55    13    18    19    17    18    39    68    42     1
## 56    19    20    23    20    18    73    85    NA     2
## 57    20    25    25    22    19    36    99    45     2
## 58    16    25    18    18    16    52    63    45     2
## 59    16    22    24    20    16    39    90    48     3
## 60    13    19    20    21    17    52    75    44     3
## 61    10    19    20    19    19    41    64    43     1
## 62    21    17    24    24    18    59    95    46     4
## 63    23    16    25    13    18    49   105    49     1
## 64    13    15    20    21    21    51    84    45     3
## 65    19    16    25    23    18    40   103    40     3
## 66    18    24    19    18    16    51    87    39     0
## 67    14     8    25    25    25    31    71    46     3
## 68    14    13    23    15    17    49    79    40     3
## 69    18    16    23    20    15    56    81    40     2
## 70    16    11    21    22    20    56    90    45     3
## 71    16    16    20    24    14    55    84    31     0
## 72    10    22    24    17    20    57    64    46     2
## 73    19    17    22    21    15    54    93    46     3
## 74    19    22    22    20    19    71    88    39     4
## 75    15    13    17    19    16    51    85    39     3
## 76    18    25    22    24    19    50    92    41     4
## 77    17    14    21    20    17    55    82    39     2
## 78    17    18    23    20    18    58    72    37     4
## 79    16    10    20    17    18    68    60    40     0
## 80    12    11    22    18    20    57    62    38     2
## 81    23    23    25    20    22    37    84    44     3
## 82    13    12    23    22    16    68    71    36     3
## 83    10    13    19    14    24    63    66    42     4
## 84    17    16    22    18    24    48    75    43     4
## 85    12    19    19    15    19    52    77    42     2
## 86    19    19    25    23    17    48    77    39     1
## 87     9    25    22    20    18    51    85    49     3
## 88    23    19    21    17    22    34    88    NA     1
## 89    16    18    24    24    24    50    62    48     1
## 90    19    20    23    19    24    52    79    43     2
## 91    16    18    22    19    20    55    84    45     4
## 92    22     8    18    19    19    66    78    47     3
## 93    13    12    19    18    18    67    63    40     3
## 94    19    22    21    21    16    41    91    49     2
## 95    15    17    20    19    17    51    78    33     2
## 96    18    25    18    23    20    39   101    38     1
## 97    15    NA    NA    NA    NA    63    90    47     3
## 98    16    15    25    25    21    36    90    48     3
## 99    17    18    24    24    21    39    97    49     4
## 100   15    25    21    24    16    37   101    45     2
## 101   16    14    20    16    16    55    74    43     3
## 102   22    22    22    19    18    52    91    46     2
## 103   21    23    22    20    13    52    64    40     3
## 104    9    19    20    24    16    53    65    39     1
## 105   20    18    19    18    16    53    89    NA     1
## 106   25    17    25    21    17    68    92    46     2
## 107   10    13    17    17    18    68    57    36     3
```

```
## 108            13     24     22     20     17     45     81     33     3
## 109            20     11     21     12     16     80     81     47     3
## 110            15     22     15     21     16     48     60     33     2
## 111            13     19     23     20     20     48     85     45     4
## 112            16     20     21     19     20     38     88     44     3
## 113            10     23     17     17     21     67     52     42     2
## 114            16     16     23     21     19     66     86     42     3
## 115            15     21     15     20     13     54     84     33     3
## 116            13     25     25     23     22     37     99     48     3
## 117            16     18     25     17     25     45     85     48     3
## 118            20     22     20     19     17     49     94     40     4
## 119            18     17     23     19     17     55    103     46     2
## 120            16     25     23     23     16     54     85     44     3
## 121            18     23     24     20     16     47     77     47     2
## 122            13     22     19     21     18     47     79     44     2
## 123            18     17     18     20     16     55     66     44     2
## 124            19     19     24     22     17     33     88     45     3
## 125            12     25     19     24     16     57     58     44     2
## 126            19     22     23     19     15     44     85     44     3
## 127            16      9     23     25     20     56     76     43     3
## 128            17     19     19     16     15     NA     87     43     4
## 129            22     12     16     23     13     66     96     42     2
## 130            19     15     20     21     18     46     97     46     3
## 131            14     16     18     21     18     46     93     42     3
## 132            NA      8     19     21     20     56     77     35     2
## 133            17     17     22     23     16     32     74     41     4
## 134            23     14     23     19     17     69     89     33     3
## 135            14     20     17     20     17     50     83     37     2
## 136            16     20     24     21     17     48     86     46     3
## 137            16     19     16     13     13     51     85     43     3
## 138            16     20     21     17     14     NA     NA     43     3
## 139            18     12     22     20     17     39     88     40     3
## 140            24     20     25     25     14     49     95     NA     3
## 141            15     21     19     24     17     27     86     43     1
## 142            NA     24     18     18     17     44     78     NA     2
## 143            21     14     24     17     19     61     79     42     4
## 144            12     16     18     17     19     45     84     42     3
## 145            13     20     19     23     17     49     90     43     2
## 146            15     19     25     24     20     65     89     49     3
## 147            18     13     23     23     16     55     66     40     4
## 148            13     19     19     20     18     55     68     45     3
## 149            22     16     22     22     11     67    104     44     2
## 150             7     10     24     19     25     63     75     43     2
##     sci_cur sci_tru sci_impo sci_id MatrixCorrectCount GNGdprime  GNGbetalog
## 1        15      19       18     19                  7 3.6990958 -1.74575440
## 2        16      20       20     18                  6 1.8949266 -1.00725297
## 3        18      18       16     15                  7 2.3391607 -0.57259336
## 4        17      18       20     16                  6 4.0006734 -1.62609009
## 5        11      12       12     12                  2        NA          NA
## 6        14      NA       NA     NA                  5        NA          NA
## 7        17      19       19     19                  8 3.7327300 -1.73436549
## 8        17      16       17     17                  5 2.7319268 -1.07166164
## 9        19      15       20     19                  5 3.7113647 -0.86960836
## 10       13      14       17     15                 NA 2.9252307 -0.71048868
```

```
## 11       12       17       19       18                7 2.7109785 -1.86110785
## 12       18       15       15       18                8 2.6975620 -1.22591799
## 13       16       19       20       18                7 2.3455435 -1.24523474
## 14        8       16       15       10                7 1.3924296 -0.36088673
## 15       12       15       17       10                4 2.7477910 -0.95171211
## 16        9       15       17       14                5 4.1800874 -0.88397836
## 17        9       12       16       11                7 3.0107551 -1.87091973
## 18       10       15       14       13                8 3.3715020 -1.83098496
## 19       18       16       19       17                4 1.8630742 -0.71354963
## 20       12       15       20       16                8 3.0657039 -0.98039497
## 21       14       16       19       19                8 3.9169481 -1.66327253
## 22       12       15       18       18                8 3.1243005 -1.37698819
## 23       14       16       17       15                2 1.9279548 -0.41970204
## 24       14       16       20       16                7 1.6808831 -0.97160330
## 25       16       16       17       18                9 2.6459853 -1.23211309
## 26       10       14       17       18                5 1.9282842 -0.53235911
## 27       15       13       15       19                6 3.1360994 -1.86345131
## 28       13       18       16       16                7 1.5035685 -0.72256149
## 29       13       15       16       14                5 1.9507611 -0.48056593
## 30       15       15       16       18                3        NA          NA
## 31       19       16       20       17                5 2.3966572 -0.79423322
## 32       17       16       15       15                8 3.1360994 -1.86345131
## 33       13       16       16       17                6 2.7268495 -0.86141340
## 34       16       11       17       18                7 3.6990958 -1.74575440
## 35       19       14       17       15                5 2.4186690 -1.11330428
## 36       12       15       15       16                6 3.0397008 -1.39554325
## 37       17       19       19       19                9 1.6526945 -0.96542567
## 38       13       17       19       15                8 2.9359266 -1.02162963
## 39       14       16       19       20                7 2.5377699 -0.76323391
## 40       16       16       20       19                7 4.0961770 -0.94086964
## 41       15       14       16       17                7 3.4069773 -0.57309693
## 42       20       18       20       18                4        NA          NA
## 43       15       18       17       18                8 2.0097021 -0.63572273
## 44       18       18       18       18                9 3.0747648 -0.84664844
## 45       15       17       18       19                4        NA          NA
## 46       15       13       19       19                7 2.0642028 -0.70042798
## 47       18       16       18       15                6 3.6342777 -1.76631680
## 48       16       18       20       20                7 1.1400025 -0.52506321
## 49       18       18       15       16                6 3.8028731 -1.70903357
## 50       19       16       20       19                8 2.3672943 -0.93265956
## 51       13       16       18       15                7 3.2105380 -0.92573966
## 52       17       18       17       18                6 1.5725664 -0.62196467
## 53       16       12       18       20                5 2.3966572 -0.79423322
## 54       11       18       18       16                9 4.0454717 -1.60494491
## 55        7       19       18        9                8 2.2280977 -0.33982429
## 56       17       14       20       13                4        NA          NA
## 57       20       19       20       20                5 2.5505338 -0.18390530
## 58       15       15       16       16                5 3.3365323 -1.05335464
## 59       14       19       20       20                7 3.3398437 -1.31566550
## 60       11       16       15       18                4 1.6616422 -0.04904076
## 61        9       14       15       12                4 1.4387027 -0.26054363
## 62       15       15       18       19                8 2.3004982 -1.42543762
## 63       16       15       20       17                7 2.1653360 -0.94805277
## 64       14       17       19       17                5 2.7187289 -0.95880354
```

```
## 65       12       16       14       19                7 4.0454717 -1.60494491
## 66       11       14       14       14                9 2.3208714 -1.01900125
## 67       12       18       20       13                8 2.0963972 -0.88648642
## 68       13       16       17       20                5 1.0811242 -0.49735781
## 69       12       15       18       16                7 3.5126147 -1.79998566
## 70       12       18       19       13                8 2.2396386 -0.71831661
## 71       12       17       16       15                5 1.6526945 -0.96542567
## 72       15       18       18       14                6 1.8464403 -0.62047477
## 73       17       17       20       19                6 2.8486580 -0.59754265
## 74       18       18       16       18                7 2.8865752 -1.87159027
## 75        8       15       15       15                6 2.5642790 -0.50234739
## 76       17       19       20       19                6 2.6764158 -1.43988976
## 77       17       17       20       18                4 3.5421721 -1.79239715
## 78       15       12       11       15                7 3.6414214 -1.19600120
## 79       10       11       17       16                8 3.8775640 -1.67971017
## 80       11       14       15       16                5 2.3929062 -0.48467594
## 81       12       14       20       17                5 3.1243005 -1.37698819
## 82       18       16       17       17                6 2.6018005 -1.44190303
## 83       12       18       18       13                9 3.2985568 -1.06856646
## 84       14       10       19       15                7 3.9169481 -1.66327253
## 85       11       18       19       16                5 2.8811350 -0.41457740
## 86       13       15       19       14                5 3.3759164 -1.03691700
## 87       15       15       17       19                8 3.4168837 -1.01910378
## 88       18       12       17       15                9        NA          NA
## 89       12       17       17        9                8 3.0356679 -1.86997865
## 90       18       17       20       18                7 1.2440383  0.09426731
## 91       17       15       18       17                8 2.7616219 -1.86550518
## 92       15       16       16       16               NA 3.6990958 -1.74575440
## 93       12       12       17       15                8 2.2357721 -0.76653300
## 94       11       16       17       17                7 1.2578906 -0.38430128
## 95       18       15       15       18                5 1.2193805 -0.12094127
## 96       14       18       19       19                8 1.8583247 -0.74988989
## 97        8       14       13       13               NA 3.8362831 -1.09767497
## 98       20       16       20       17               NA 2.5521592 -1.44190303
## 99       13       12       19       18                9 2.5087078 -0.77032534
## 100      15       15       20       20                8 2.2402760 -0.94440783
## 101      15       17       19       18                6 2.0928383 -1.22591799
## 102      18       14       20       19                6 1.5426176 -0.51673856
## 103      10       16       16       13                7        NA          NA
## 104       8       15       16       13                4 3.4596417 -0.99973457
## 105       9       17       16       13                6        NA          NA
## 106      20       16       20       19                6 1.2001138 -0.55392279
## 107      11       19       14       15                6 2.9054261 -1.41861162
## 108      11       16       16       13                9 2.0439524 -1.10713734
## 109      19       18       20       19                6 2.3427010 -0.80380068
## 110      13        8       16       16                8 0.8797705 -0.41108769
## 111      20       17       19       18                9 2.9218713 -0.79928422
## 112      18       15       17       17                8 1.8763195 -0.53918511
## 113      12       16       16       12                7 2.4989563 -0.91308611
## 114      13       15       16       19                7 2.0439524 -1.10713734
## 115      18       12       20       18                9 2.5642790 -0.50234739
## 116      20       20       19       19                8 1.2399804 -0.45517775
## 117      12       17       20       15                8 3.4835526 -1.80707708
## 118      16       17       18       18                8 2.8165265 -1.05310657
```

```
## 119      12      14       20     17                 2 2.4260396 -0.73153170
## 120      19      17       16     16                 9 1.0433926 -0.02672405
## 121      13      15       16     16                 7 2.3160661 -0.93824089
## 122      14      18       19     20                 6 2.8074449 -0.93600682
## 123      15      15       17     14                 7 2.4221933 -1.01019507
## 124      14      17       19     17                 9 3.2105380 -0.92573966
## 125       8      17       18     18                 6 4.0454717 -1.60494491
## 126      16      16       18     18                 7 2.6764158 -1.43988976
## 127       7      11       11     14                 7 2.7499563 -1.21844178
## 128      17      18       17     16                 8        NA          NA
## 129      13      16       19     19                 7 2.3686418 -1.11600814
## 130      16      18       20     15                 9 2.6975620 -1.22591799
## 131      18      16       17     19                 8 3.1127404 -0.83143662
## 132      NA      NA       NA     NA                 7 1.9162028 -0.94263822
## 133      18      15       19     18                 7 1.2590101 -0.76132109
## 134       8      18       15     19                 4 2.6764158 -1.43988976
## 135      15      15       19     18                 6 3.0675049 -1.38978795
## 136      12      13       18     12                 7 2.2637489 -0.81552177
## 137      13      15       20     19                 9 4.6363489 -1.24448376
## 138      13      15       20     18                 8 3.0025903 -0.67883922
## 139      16      14       16     17                 7 1.2952092 -0.26606055
## 140      17      18       20     20                 9 2.6338700 -1.85227352
## 141      15      15       16     13                 3 2.7519454 -0.77195199
## 142      NA      NA       NA     NA                 7        NA          NA
## 143      14      15       18     20                 6 1.0724491  0.20702590
## 144       9      15       20     18                 8 2.8486580 -0.59754265
## 145      14      12       18     14                 6 2.1962288 -1.02371838
## 146      16      17       19     17                 9 3.6663033 -1.75638530
## 147      13      14       16     13                 9 3.2130165 -1.35419147
## 148      13      13       10     12                 4 0.9716106 -0.36589408
## 149      19      20       18     16                 5 4.0006734 -1.62609009
## 150      14      19       18     17                 4 3.6663033 -1.75638530
##     MeanGoRT    SDGoRT GlobalToLocalInterferencelog GlobalToLocalPrecedencelog
## 1   404.6540  59.32952                  -0.82472979                 0.07434374
## 2   332.6818  87.76311                   0.16481852                 0.07913384
## 3   400.6000  84.25173                          NaN                 0.09116808
## 4   425.0445 103.50044                  -0.37144695                 0.06984183
## 5         NA        NA                           NA                         NA
## 6         NA        NA                           NA                         NA
## 7   409.0867  64.87193                          NaN                 0.08747569
## 8   438.4237  90.50604                  -1.25189253                 0.08493435
## 9   450.0868  95.60677                  -0.52782772                 0.07556402
## 10  476.6976 121.75798                          NaN                 0.11832027
## 11  354.2585  83.69391                          NaN                 0.11623024
## 12  362.8427  75.19753                          NaN                 0.08834042
## 13  358.1717  78.74999                           NA                         NA
## 14  427.0520 116.09574                          NaN                 0.09552970
## 15  388.6169  81.29261                          NaN                 0.08966475
## 16  510.9298  92.72318                          NaN                 0.08900591
## 17  442.8713  81.84922                  -0.98790889                 0.07697524
## 18  361.9703  56.94116                          NaN                 0.08490554
## 19  461.0077 133.23553                          NaN                 0.08342596
## 20  510.6263 104.10493                          NaN                 0.16962423
## 21  379.7570  59.47815                  -0.59473686                 0.07353177
```

```
## 22  369.3913  60.70374  -1.47365695  0.06931156
## 23  483.9000 149.49199         NaN  0.08666811
## 24  322.3443  74.94041  -0.57032249  0.08058912
## 25  376.4272  73.62481         NaN  0.09192375
## 26  358.3746  76.51562         NaN  0.09021090
## 27  330.3910  55.03769         NaN  0.08479096
## 28  356.9399  84.96941  -1.10750401  0.07915931
## 29  428.4096 117.46738         NaN  0.08736924
## 30        NA        NA          NA          NA
## 31  446.4586 100.12027         NaN  0.15413803
## 32  395.7297  80.59988         NaN  0.11543925
## 33  361.2570  73.07871         NaN  0.06916883
## 34  414.5547  88.76471  -0.89287733  0.07288570
## 35  382.2370 106.79025  -0.30886537  0.06726676
## 36  430.0535  82.76832  -1.16509761  0.08369225
## 37  302.9524  55.84429         NaN  0.08569663
## 38  370.2557  77.78421  -0.42606238  0.07325912
## 39  484.4334 114.88013  -0.69392826  0.06615275
## 40  528.0171  82.40899         NaN  0.08966911
## 41  491.7898 107.52729         NaN  0.08186760
## 42        NA        NA          NA          NA
## 43  398.9634  91.61292         NaN  0.09639073
## 44  444.4915 112.57819  -0.10595456  0.03854644
## 45        NA        NA          NA          NA
## 46  468.5614 101.09151  -0.88451889  0.07504893
## 47  368.3770  58.30704  -0.73960576  0.07159786
## 48  409.7410 106.00650  -0.42186120  0.07902440
## 49  423.2847  87.14320  -1.41025984  0.02270506
## 50  373.7423 113.08916   0.22138379  0.06338992
## 51  428.5473  78.17875  -0.88783775  0.08782490
## 52  355.9582  93.70388  -1.27071372  0.07703546
## 53  501.8924 141.76424         NaN  0.09912117
## 54  520.0273  68.34155  -0.89758042  0.08389532
## 55  413.4674  92.61774  -0.98571183  0.08368216
## 56        NA        NA          NA          NA
## 57  533.9532 135.98137         NaN  0.08492508
## 58  447.9219  85.46470  -0.30525764  0.07161749
## 59  418.2802  93.20915  -0.60336726  0.07344478
## 60  655.5049 175.69540   0.10857591  0.00000000
## 61  480.9471 167.70536  -0.28479255  0.06844442
## 62  290.5893  49.83554  -0.90379226  0.08240925
## 63  377.2539  95.94254          NA          NA
## 64  415.7285  86.88115  -0.35122458  0.06895503
## 65  415.5383  90.85252   0.01607676  0.03005013
## 66  420.1894  90.58257  -0.37629276  0.07076473
## 67  355.8645  72.95775  -0.24897584  0.06463802
## 68  310.6336  72.29845  -0.06290048  0.06700588
## 69  394.2137  75.44489  -0.26376824  0.06951883
## 70  355.7801  74.35544         NaN  0.07886957
## 71  320.4291  66.97018         NaN  0.09003797
## 72  407.6762 111.08294         NaN  0.10177154
## 73  493.0732 124.55876  -0.22030958  0.06007764
## 74  342.3646  88.83189  -0.49770242  0.05730210
## 75  473.9483 109.01576  -0.48796308  0.06563150
```

```
## 76  375.5224  87.01094 -1.16985441 0.08042140
## 77  417.3433  82.46279 -0.48741525 0.07655391
## 78  368.6817  67.17141 -0.35809149 0.06266394
## 79  390.5467  65.17623 -1.29906736 0.08462124
## 80  468.9088 118.76124 -0.70888760 0.08539078
## 81  377.3090  81.89450 -0.14012912 0.08156959
## 82  360.5097  60.57170 -0.49673113 0.07148048
## 83  503.1893 111.94467 -0.02329184 0.05423012
## 84  374.8183  56.29476 -0.54107121 0.07689456
## 85  512.6014 142.80075 -0.12613847 0.07531019
## 86  419.3285  85.83525 -0.46278256 0.07473674
## 87  419.1007  86.43608 -0.14170494 0.05824997
## 88        NA        NA          NA         NA
## 89  356.5867  65.38056 -0.85382223 0.07601702
## 90  540.1320 159.93108  0.12380137 0.05995916
## 91  319.3683  79.44447 -1.60407077 0.06931136
## 92  389.0200  70.13517         NaN 0.10954597
## 93  356.1702  68.81150 -0.60749265 0.08025003
## 94  355.1151  84.80985         NaN 0.09749264
## 95  425.2316 122.32845 -1.05738571 0.08538179
## 96  418.3986  84.37228 -0.42875587 0.07259611
## 97  375.4987  81.60162          NA         NA
## 98  385.2407  75.97954          NA         NA
## 99  481.9178 113.14240 -0.32949839 0.06378034
## 100 414.3444  95.28302 -0.30170187 0.06294579
## 101 364.0896  71.47969  0.19750144 0.06329213
## 102 319.7942  64.06533  0.47372839 0.06466924
## 103       NA        NA          NA         NA
## 104 474.4034  88.74308         NaN 0.12851462
## 105       NA        NA          NA         NA
## 106 359.7462 109.78479 -0.29566160 0.07067528
## 107 373.4125  58.41357 -0.79916788 0.07784184
## 108 331.4966  66.43330         NaN 0.09108677
## 109 371.7509  83.08827 -0.39358783 0.06017175
## 110 385.4282 129.43615 -0.69895779 0.09142494
## 111 385.1064  88.62593         NaN 0.09612765
## 112 369.6223 125.14064 -1.15637360 0.08961563
## 113 446.3596 131.03898 -0.91921867 0.09530770
## 114 353.4003  79.09359         NaN 0.07817125
## 115 473.3868 108.66002 -0.11109678 0.06070631
## 116 477.5364 181.72357 -1.03755248 0.07446885
## 117 388.0403  65.83793 -0.75720706 0.07901153
## 118 395.0468  94.64061 -0.81267700 0.07594789
## 119 491.5318 107.17270  0.20257907 0.06742931
## 120 442.2892 148.84875 -0.60178975 0.08397434
## 121 409.0844  70.99338 -1.15103334 0.07336249
## 122 447.0436  69.43740 -0.67094481 0.07478513
## 123 391.3209  72.77049 -0.20314484 0.07113333
## 124 475.7027 104.91087         NaN 0.08982446
## 125 457.5233  65.67959 -0.39466819 0.06968806
## 126 344.7809  57.74282         NaN 0.10494479
## 127 352.5980  82.54932 -1.10927362 0.08629379
## 128       NA        NA          NA         NA
## 129 344.8264  60.22154 -1.86822832 0.08087726
```

```
## 130 380.9360 137.60166                -0.21577530                0.05843487
## 131 431.8068  97.92490                -0.33752562                0.07734269
## 132 333.1546  71.53928                         NA                        NA
## 133 328.0453  94.94767                -0.69582510                0.07522329
## 134 360.8517  70.03585                -0.24838687                0.06949296
## 135 427.2258  93.74986                -0.94629059                0.08117927
## 136 488.9951 116.96829                -0.65807328                0.07528036
## 137 503.7144 104.62921                -0.42059582                0.06571622
## 138 472.5300 116.37661                         NA                        NA
## 139 484.7264 166.68138                -0.47972240                0.05698497
## 140 388.7880  85.93326                         NA                        NA
## 141 492.3264 110.22625                        NaN                0.08258173
## 142       NA        NA                         NA                        NA
## 143 587.1027 182.25915                 0.11419279                0.02137891
## 144 475.8010 111.18467                -0.15428148                0.06482476
## 145 432.7891  89.48393                -0.87004712                0.07408902
## 146 371.2510  55.56097                -1.51090099                0.09562461
## 147 357.3391  76.62475                -0.94962489                0.07906439
## 148 370.0846  96.04684                -0.23053272                0.06277892
## 149 450.1250  85.77585                -0.92032696                0.08226645
## 150 453.2299  58.67465                -1.02050065                0.07511735
##     NeckerTotalRatelog FlexSum
## 1          0.041392685    37.5
## 2          0.034762106    61.0
## 3          0.028028724    26.5
## 4          0.034762106    28.5
## 5                   NA      NA
## 6                   NA      NA
## 7          0.028028724    41.0
## 8          0.054357662    21.0
## 9          0.014240439    40.5
## 10         0.054357662    20.0
## 11         0.028028724    20.0
## 12         0.091080469    24.0
## 13         0.047923552    15.5
## 14         0.014240439    25.0
## 15         0.047923552    44.0
## 16         0.034762106    27.0
## 17         0.021189299    29.5
## 18         0.041392685    21.0
## 19         0.028028724    26.0
## 20         0.007178585    20.0
## 21         0.161368002    29.5
## 22         0.085171610    31.5
## 23         0.073107098    25.0
## 24         0.041392685    27.0
## 25         0.054357662    27.0
## 26         0.034762106    16.0
## 27         0.073107098    42.0
## 28         0.066946790    25.0
## 29         0.000000000    42.0
## 30                  NA      NA
## 31         0.091080469    28.0
## 32         0.073107098    52.5
```

```
## 33         0.096910013    38.0
## 34         0.000000000    50.0
## 35         0.047923552    27.0
## 36         0.096910013    20.0
## 37         0.113943352    18.0
## 38         0.028028724    30.0
## 39         0.146128036    57.5
## 40         0.021189299    23.0
## 41         0.047923552    36.5
## 42                  NA      NA
## 43         0.073107098    24.0
## 44         0.060697840    30.0
## 45                  NA      NA
## 46         0.066946790    23.0
## 47         0.166331422    24.0
## 48         0.028028724    28.0
## 49         0.028028724    31.0
## 50         0.034762106    23.0
## 51         0.007178585    33.0
## 52         0.066946790    34.0
## 53         0.014240439    15.5
## 54         0.085171610    41.0
## 55         0.073107098      NA
## 56                  NA      NA
## 57         0.014240439    26.0
## 58         0.034762106    33.0
## 59         0.124938737    32.0
## 60         0.021189299    23.0
## 61         0.041392685    19.0
## 62         0.085171610    28.0
## 63         0.079181246    17.0
## 64         0.073107098    33.0
## 65         0.041392685    41.0
## 66         0.007178585    22.0
## 67         0.079181246    39.0
## 68         0.021189299    26.5
## 69         0.066946790    39.5
## 70         0.041392685    27.0
## 71         0.041392685    33.0
## 72         0.054357662    14.0
## 73         0.028028724    43.0
## 74         0.113943352    23.0
## 75         0.028028724    41.0
## 76         0.021189299    17.0
## 77         0.054357662    32.0
## 78         0.014240439    23.0
## 79         0.000000000    21.0
## 80         0.041392685    24.0
## 81         0.041392685    46.5
## 82         0.054357662    32.0
## 83         0.073107098    28.0
## 84         0.000000000    21.0
## 85         0.014240439    33.0
## 86         0.034762106    32.0
```

```
## 87         0.085171610    37.0
## 88                  NA      NA
## 89         0.171238756    22.0
## 90         0.060697840    25.0
## 91         0.079181246    29.0
## 92         0.047923552    36.5
## 93         0.041392685    25.0
## 94         0.091080469    26.0
## 95         0.047923552    48.0
## 96         0.034762106    23.0
## 97         0.066946790    24.0
## 98         0.119475841    30.5
## 99         0.041392685    19.0
## 100        0.079181246    32.0
## 101        0.079181246    21.0
## 102        0.014240439    25.0
## 103                 NA      NA
## 104        0.108339475    42.0
## 105                 NA      NA
## 106        0.054357662    43.0
## 107        0.041392685    11.0
## 108        0.021189299    22.0
## 109        0.047923552    31.5
## 110        0.073107098    26.0
## 111        0.000000000    37.0
## 112        0.000000000    36.0
## 113        0.060697840     6.0
## 114        0.021189299    28.0
## 115        0.047923552    34.0
## 116        0.000000000    16.5
## 117        0.034762106    37.0
## 118        0.047923552    23.0
## 119        0.021189299    30.0
## 120        0.146128036    37.0
## 121        0.034762106    28.0
## 122        0.047923552    22.5
## 123        0.047923552    37.0
## 124        0.041392685    37.0
## 125        0.054357662    35.0
## 126        0.041392685    54.5
## 127        0.054357662    19.0
## 128                 NA      NA
## 129        0.060697840    47.0
## 130        0.066946790    44.0
## 131        0.060697840    30.0
## 132        0.047923552    22.0
## 133        0.085171610    24.0
## 134        0.096910013    31.0
## 135        0.054357662    24.0
## 136        0.073107098    32.0
## 137        0.102662342    32.5
## 138        0.066946790    46.0
## 139        0.028028724    24.0
## 140                 NA    37.0
```

```
## 141        0.060697840    29.0
## 142                 NA      NA
## 143        0.047923552    30.0
## 144        0.034762106    35.5
## 145        0.034762106    38.5
## 146        0.113943352    45.0
## 147        0.091080469    27.0
## 148        0.021189299    17.0
## 149        0.034762106    21.0
## 150        0.060697840    31.0
```

```
# Select relevant columns
data_all_sub_cleaned <- data_all_sub_cleaned %>%
  select(-Participant.Private.ID)
```

Citizen's Initiative

Potentially problematic ppts for CI: ri0gfe1h 5767733 skipped reading at least one article vu1mvwkd 5691188 didn't focus on the first CI text 8zz2teyh 5629594 is dyslexic tjh0tjq3 4887029 CI task invalid