

Round_1_ICC

Kriss Tesink and Johan Hensman

2023-05-29

Introduction

This document presents statistical analyses of correlation between Dutch and English ASA questionnaire items.

For this first translation round, all 90 ASA questionnaire items and their corresponding 90 translations are evaluated.

We use the following packages:

```
library(foreign) # Open various data files
library(nlme)    # Run multilevel linear models
library(car)     # Package linear regression
library(haven)  # Use read_sav fuction
library(dplyr)  # Use select function
library(knitr)  # Get markdown file
library(tinytex) # Use TeX environment
library(pander) # For pandering tables
panderOptions("table.alignment.default", "left")
```

Data files Final_ASA_Dutch_Round_1_First_Half_anonym.sav and Final_ASA_Dutch_Round_1_Second_Half_anonym.sav

The raw data gathered from the online platform Qualtrics was first anonymized by using the “anonymize_and_preprocess_first_dutch_translation_round_to_publish.py” python code, which produced the output files “Final_ASA_Dutch_Round_1_First_Half_anonym.sav” and the file “Final_ASA_Dutch_Round_1_Second_Half_anonym.sav”.

We divided the participants into two groups to control for fatigue effects. In the first group, human-ASA interaction evaluation data from the first 44 ASA questionnaire items were collected from 30 bilingual participants. These first 44 ASA questionnaire items correspond to the first 12 constructs of the ASA questionnaire. In the second group, human-ASA interaction evaluation data of the last 46 items were collected from 30 other bilingual participants. These last 46 ASA questionnaire items correspond to the last 12 constructs of the ASA questionnaire.

The 60 total Bilingual participants with a Dutch Mother tongue and fluency in English rated a Human-ASA interaction by first answering a block of English ASA questionnaire items, followed by a block of Dutch translated ASA questionnaire items. Included in each block were 7 attention check control questions, for a total of 14 attention checks. We removed data from participants who failed attention checks (in this case, none of the participants rating the first 44 items failed the attention check), and then removed all irrelevant

data, keeping only the answers to the English questions and Dutch questions. After pruning the irrelevant data, the two datasets were combined into one.

```
dataset <- read_sav("Final_ASA_Dutch_Round_1_First_Half_anonym.sav")
dataset2 <- read_sav("Final_ASA_Dutch_Round_1_Second_Half_anonym.sav")
#Importing data
#"dataset" contains ASA question items from the first 12 constructs
#"dataset2" contains ASA question items from the last 12 constructs
```

```
dataset <- data.frame(select(dataset, AC_English_1:Q_ITEMS_438.0))
dataset2 <- data.frame(select(dataset2, AC_English_1:Q42_442))
#Selecting only relevant question items, which include attention checks
#and ASA questionnaire items
```

Here, we change the column names of such that the ASA questionnaire items are renamed according to which language they are in, and which construct they belong to

```
#Here, all the column names are changed so that ASA questionnaire items are
#named according to their construct.
colnames(dataset)[59:102] <- c("D_HLA1", "D_HLA2", "D_HLA3", "D_HLA4", "D_HLB1",
  "D_HLB2", "D_HLB3", "D_HLB4", "D_HLB5", "D_NA1",
  "D_NA2", "D_NA3", "D_NA4", "D_NA5", "D_NB1",
  "D_NB2", "D_NB3", "D_AAS1", "D_AAS2", "D_AAS3",
  "D_AU1", "D_AU2", "D_AU3", "D_PF1", "D_PF2",
  "D_PF3", "D_AL1", "D_AL2", "D_R_AL3", "D_AL4",
  "D_AL5", "D_AS1", "D_AS2", "D_AS3", "D_APP1",
  "D_R_APP2", "D_APP3", "D_UAA1", "D_UAA2",
  "D_R_UAA3", "D_R_AE1", "D_AE2", "D_AE3",
  "D_R_AE4")
#Changing columns of Dutch ASA question items in dataset

colnames(dataset2)[61:106] <- c("D_UE1", "D_UE2", "D_UE3", "D_UT1", "D_UT2",
  "D_UT3", "D_UAL1", "D_UAL2", "D_UAL3", "D_UAL4",
  "D_UAL5", "D_UAL6", "D_AA1", "D_AA2", "D_AA3",
  "D_R_AC1", "D_R_AC2", "D_R_AC3", "D_R_AC4",
  "D_AI1", "D_AI2", "D_R_AI3", "D_AI4", "D_AT1",
  "D_AT2", "D_R_AT3", "D_SP1", "D_SP2", "D_SP3",
  "D_IIS1", "D_IIS2", "D_IIS3", "D_IIS4",
  "D_AEI1", "D_AEI2", "D_R_AEI3", "D_AEI4",
  "D_R_AEI5", "D_UEP1", "D_UEP2", "D_UEP3",
  "D_UEP4", "D_UAI1", "D_UAI2", "D_UAI3",
  "D_UAI4")
#Changing columns of Dutch ASA question items in dataset2

colnames(dataset)[8:51] <- c("HLA1", "HLA2", "HLA3", "HLA4", "HLB1", "HLB2",
  "HLB3", "HLB4", "HLB5", "NA1", "NA2", "NA3", "NA4",
  "NA5", "NB1", "NB2", "NB3", "AAS1", "AAS2", "AAS3",
  "AU1", "AU2", "AU3", "PF1", "PF2", "PF3", "AL1",
  "AL2", "R_AL3", "AL4", "AL5", "AS1", "AS2", "AS3",
  "APP1", "R_APP2", "APP3", "UAA1", "UAA2", "R_UAA3",
  "R_AE1", "AE2", "AE3", "R_AE4")
#Changing columns of English ASA question items in dataset
```

```
colnames(dataset2)[8:53] <- c("UE1", "UE2", "UE3", "UT1", "UT2", "UT3", "UAL1",
"UAL2", "UAL3", "UAL4", "UAL5", "UAL6", "AA1",
"AA2", "AA3", "R_AC1", "R_AC2", "R_AC3", "R_AC4",
"AI1", "AI2", "R_AI3", "AI4", "AT1", "AT2",
"R_AT3", "SP1", "SP2", "SP3", "IIS1", "IIS2",
"IIS3", "IIS4", "AEI1", "AEI2", "R_AEI3", "AEI4",
"R_AEI5", "UEP1", "UEP2", "UEP3", "UEP4", "UAI1",
"UAI2", "UAI3", "UAI4")
#Changing columns of English ASA question items in dataset2
```

```
dataset <- as.data.frame(lapply(dataset[1:102],
function(y) as.numeric(gsub('[a-zA-Z]', '', y))))
dataset2 <- as.data.frame(lapply(dataset2[1:106],
function(y) as.numeric(gsub('[a-zA-Z]', '', y))))
#Transform the data to numeric representation
```

```
i <- grep("AC_",colnames(dataset))
# Find the column number of attention control questions
Atten <- c(-3,3,3,-3,0,3,-3,-3,3,3,-3,0,3,-3)
# Correct answers for the 14 attention control questions
x <- NULL # Row number of participant who failed the attention check
for (j in (1:nrow(dataset))){
# Find participants who failed attention check in 'dataset'
count <- 0
# The number of incorrectly answered attention questions of each participant
for (k in 1:14){
if (as.numeric(dataset[[i[k]]][j])!=Atten[k])
# Check whether each participant's
# attention question answers are consistent with the correct answers
count <- count+1
}
if (count>0)
# Row number of the participant who failed more than zero
# attention control questions were added to 'x'
x <- append(x,j)
# Participants who failed more than two attention control questions
}

m <- length(x) # The number of participants who failed attention check
if (m!=0)
dataset <- dataset[-x,]
# Participants who failed attention checks in "dataset" were excluded
```

```
p <- grep("AC_",colnames(dataset2))
# Find the column number of attention control questions
Atten2 <- c(0,-3,3,2,3,-3,1, 3,0,-3,3,2,3,-3)
# Correct answers for the 14 attention control questions
x2 <- NULL # Row number of participant who failed the attention check
for (j in (1:nrow(dataset2))){
# Find participants who failed attention check in 'dataset'
count <- 0
# The number of incorrectly answered attention questions of each participant
for (k in 1:14){
```

```

if (as.numeric(dataset2[[p[k]]][j])!=Atten2[k])
  # Check whether each participant's
  # attention question answers are consistent with the correct answers
  count <- count+1
}
if (count>0)
  # Row number of the participant who failed more than zero
  # attention control questions were added to 'x'
  x2 <- append(x2,j)
  # Participants who failed more than two attention control questions
}

m2 <- length(x2) # The number of participants who failed attention check
if (m2!=0)
  dataset2 <- dataset2[-x2,]
# Participants who failed attention checks in "dataset2"were excluded

```

```

d1 <- as.data.frame(select(dataset, HLA1:R_AE4, D_HLA1:D_R_AE4))
# Select scores of 44 English items and corresponding Dutch translations

```

```

d2 <- as.data.frame(select(dataset2, UE1:UAI4, D_UE1:D_UAI4))
# Select scores of 46 English items and corresponding Dutch translations

```

```

#Since some rows may have been removed due to failed attention checks,
#therefore, NA rows may have to be appended to d1 or d2 to ensure that they are
#equal length, allowing d1 and d2 to later be combined
if(count(d2)>count(d1)) {
  diff <-count(d2)-count(d1)
  while(diff>0) {
    #fill d1 with NA rows until d1 is as long as d2
    d1[nrow(d1) + 1 , ] <- NA
    diff <- diff-1
  }
}
if(count(d1)>count(d2)) {
  diff <-count(d1)-count(d2)
  while(diff>0) {
    #fill d2 with NA data until d2 is as long as d1
    d1[nrow(d2) + 1 , ] <- NA
    diff <- diff-1
  }
}
d_total <- cbind(select(d1,HLA1:R_AE4), select(d2,UE1:UAI4),
  select(d1,D_HLA1:D_R_AE4), select(d2,D_UE1:D_UAI4))
# Combine evaluation scores of 44 items and 46 items

```

```

for (i in grep("R_",colnames(d_total))){
  # Find column number of reversing-scoring items and translations
  d_total[[i]][ ] <- d_total[[i]][ ]*(-1)
  # Reverse scores of reverse-scoring items and translations
}

```

Analysis and results

ICC values were computed for all 90 questionnaire items, and a random intercept model was applied to the dataset. This model consists of a fixed intercept (~ 1) and incorporates participant as a random intercept, denoted by $\text{random} = \sim 1|\text{id}$. In this context, 'id' refers to the unique participant code assigned to the 30 bilingual participants whose scores were used to determine the ICC values.

We calculated ICC as: $\rho_I = \frac{\tau^2}{\tau^2 + \sigma^2}$ where σ^2 is the variance within the score of individual, and τ^2 is the variance between participants [Finch2019multilevel]. The *getICC* function is used to calculate the ICC value.

```
getICC <-function(model)
  # Function for ICC value calculation using multilevel linear model
{
  vc.model <- VarCorr(model)
  # Estimated variances and correlations between the random-effects terms
  sigma_var <-as.numeric(vc.model[2,1])
  # Variance within the groups
  tau_var <- as.numeric(vc.model[1,1])
  # Variance between the groups
  icc <- tau_var/(tau_var + sigma_var)
  # Calculate ICC value
  return(icc)
}
```

The *getLME* function was defined to execute a multilevel model and then derive the corresponding ICC value for said model using the *getICC* function. This function requires the input of scores for both languages and the participant's ID number.

```
getLME <-function(s_1,s_2)
  # Function for a linear mixed-effects model
{
  id<-rownames(s_2)
  # Row names that represent the ID number of each participant
  Score_Dutch<- data.frame(id, s_1, language= 1)
  # Transform Dutch scores from wide format to long format and label as 1
  Score_English<- data.frame(id, s_2, language= 2)
  # Transform English scores from wide format to long format and label as 2
  Score_total <- rbind(Score_Dutch, Score_English)
  # Combine Dutch and English scores in the long format
  m0 <- lme(score ~ 1, data = Score_total, random = ~1|id, method = "ML")
  # Linear mixed-effects model with a fixed intercept and
  # a random intercept of participant's ID number
  return(getICC(m0))
}
```

After defining the *getLME* function, ICC values were calculated for each of the 90 questionnaire items

```
l_ICC <- data.frame(ItemID = double(), Item = character(), icc = double())
# Initialize output of ICC values of 44 items
n <- ncol(d_total)

# Numbers of columns in d1
Dutch_column_offset <- ncol(d_total)/2
```

```

# Offset, the first column with scores of the Dutch version of ASAQ item
d0 <- data.frame(d_total)

for (i in 1:90)
  # Go step by step to 90 items of the ASA questionnaire, whereby i is
  #the ASA questionnaire item number
  {
    score_Dutch <- na.omit(data.frame(score=d0[,i + Dutch_column_offset ]))
    # Select scores of Dutch version of ASAQ item i
    score_English <- na.omit(data.frame(score=d0[,i ]))
    # Select scores of English version of ASAQ items i
    iccScore <- getLME(score_Dutch, score_English)
    l_ICC <- rbind(l_ICC, data.frame (i, icc = iccScore))
    # Calculated ICC and add it to the list of ICC values,
    # with ID number of the ASA questionnaire item
  }
l_ICC$Item = colnames(select(d0,HLA1:UAI4))
pander(l_ICC, caption = "ICC values for 90 items")

```

Table 1: ICC values for 90 items For the assessment of the correlation between the English and Dutch questionnaire items, we followed Cicchetti’s categorization of ICC values [cicchetti1994guidelines], classifying questionnaire items into poor, fair, good, and excellent.

i	icc	Item
1	0.6725	HLA1
2	0.4052	HLA2
3	0.6586	HLA3
4	0.7368	HLA4
5	0.6111	HLB1
6	0.3975	HLB2
7	0.7799	HLB3
8	0.5148	HLB4
9	0.6236	HLB5
10	0.6341	NA1
11	7.738e-09	NA2
12	0.5364	NA3
13	0.6065	NA4
14	0.273	NA5
15	0.6246	NB1
16	0.5097	NB2
17	0.3704	NB3
18	0.6291	AAS1
19	0.398	AAS2
20	0.477	AAS3
21	0.7509	AU1
22	0.7518	AU2
23	0.5493	AU3
24	0.8442	PF1
25	0.6643	PF2

i	icc	Item
26	0.6491	PF3
27	0.3865	AL1
28	0.8252	AL2
29	0.7139	R_AL3
30	0.7753	AL4
31	0.9249	AL5
32	0.5586	AS1
33	0.6752	AS2
34	0.3432	AS3
35	0.7241	APP1
36	0.8079	R_APP2
37	0.633	APP3
38	0.6341	UAA1
39	0.7357	UAA2
40	0.3762	R_UAA3
41	0.5848	R_AE1
42	0.5433	AE2
43	0.7098	AE3
44	0.7142	R_AE4
45	0.6029	UE1
46	0.5247	UE2
47	0.696	UE3
48	0.6442	UT1
49	0.2196	UT2
50	0.7971	UT3
51	0.7106	UAL1
52	0.4402	UAL2
53	0.6086	UAL3
54	0.7246	UAL4
55	0.5993	UAL5
56	0.6044	UAL6
57	0.7631	AA1
58	0.4828	AA2
59	0.5531	AA3
60	0.7388	R_AC1
61	0.9036	R_AC2
62	0.7605	R_AC3
63	0.8253	R_AC4
64	0.8023	AI1
65	0.8135	AI2
66	0.7973	R_AI3
67	0.7078	AI4
68	0.3702	AT1
69	0.7193	AT2
70	0.6371	R_AT3
71	0.519	SP1
72	0.7326	SP2
73	0.7511	SP3
74	0.5918	IIS1
75	0.5755	IIS2
76	0.4384	IIS3
77	0.2571	IIS4

i	icc	Item
78	0.4491	AEI1
79	0.7272	AEI2
80	0.897	R_AEI3
81	0.7595	AEI4
82	0.8307	R_AEI5
83	0.3082	UEP1
84	0.6197	UEP2
85	0.7303	UEP3
86	0.6515	UEP4
87	0.7071	UAI1
88	0.4861	UAI2
89	0.4256	UAI3
90	0.3478	UAI4

```

poor <- data.frame(ItemID = double(), Item = character(), icc = double())
fair <- data.frame(ItemID = double(), Item = character(), icc = double())
good <- data.frame(ItemID = double(), Item = character(), icc = double())
excellent <- data.frame(ItemID = double(), Item = character(), icc = double())
#Create categorizations of ICC values, ranging from poor to excellent

for(i in 1:90){
  if(l_ICC$icc[i]>=0.75) {
    #If the ICC value is greater than 0.75, it is excellent
    excellent <- rbind(excellent, data.frame (i ,l_ICC$Item[i] ,
                                              icc = l_ICC$icc[i]))
  } else if(l_ICC$icc[i]>=0.60) {
    #If the ICC value is between 0.60 and 0.75, it is good
    good <- rbind(good, data.frame (i, l_ICC$Item[i], icc = l_ICC$icc[i]))
  } else if(l_ICC$icc[i]>=0.4) {
    #If the ICC value is between 0.4 and 0.6, it is fair
    fair <- rbind(fair, data.frame (i, l_ICC$Item[i], icc = l_ICC$icc[i]))
  } else {
    #If the ICC value is below 0.4, it is poor
    poor <- rbind(poor, data.frame (i, l_ICC$Item[i], icc = l_ICC$icc[i]))
  }
}

pander(poor, caption = "ICC values for poor items")

```

Table 2: ICC values for poor items

i	l_ICC.Item.i.	icc
6	HLB2	0.3975
11	NA2	7.738e-09
14	NA5	0.273
17	NB3	0.3704
19	AAS2	0.398
27	AL1	0.3865
34	AS3	0.3432

i	l_ICC.Item.i.	icc
40	R_UAA3	0.3762
49	UT2	0.2196
68	AT1	0.3702
77	IIS4	0.2571
83	UEP1	0.3082
90	UAI4	0.3478

```
pander(fair, caption = "ICC values for fair items")
```

Table 3: ICC values for fair items

i	l_ICC.Item.i.	icc
2	HLA2	0.4052
8	HLB4	0.5148
12	NA3	0.5364
16	NB2	0.5097
20	AAS3	0.477
23	AU3	0.5493
32	AS1	0.5586
41	R_AE1	0.5848
42	AE2	0.5433
46	UE2	0.5247
52	UAL2	0.4402
55	UAL5	0.5993
58	AA2	0.4828
59	AA3	0.5531
71	SP1	0.519
74	IIS1	0.5918
75	IIS2	0.5755
76	IIS3	0.4384
78	AEI1	0.4491
88	UAI2	0.4861
89	UAI3	0.4256

```
pander(good, caption = "ICC values for good items")
```

Table 4: ICC values for good items

i	l_ICC.Item.i.	icc
1	HLA1	0.6725
3	HLA3	0.6586
4	HLA4	0.7368
5	HLB1	0.6111
9	HLB5	0.6236
10	NA1	0.6341
13	NA4	0.6065
15	NB1	0.6246
18	AAS1	0.6291

i	l_ICC.Item.i.	icc
25	PF2	0.6643
26	PF3	0.6491
29	R_AL3	0.7139
33	AS2	0.6752
35	APP1	0.7241
37	APP3	0.633
38	UAA1	0.6341
39	UAA2	0.7357
43	AE3	0.7098
44	R_AE4	0.7142
45	UE1	0.6029
47	UE3	0.696
48	UT1	0.6442
51	UAL1	0.7106
53	UAL3	0.6086
54	UAL4	0.7246
56	UAL6	0.6044
60	R_AC1	0.7388
67	AI4	0.7078
69	AT2	0.7193
70	R_AT3	0.6371
72	SP2	0.7326
79	AEI2	0.7272
84	UEP2	0.6197
85	UEP3	0.7303
86	UEP4	0.6515
87	UAI1	0.7071

```
pander(excellent, caption = "ICC values for excellent items")
```

Table 5: ICC values for excellent items

i	l_ICC.Item.i.	icc
7	HLB3	0.7799
21	AU1	0.7509
22	AU2	0.7518
24	PF1	0.8442
28	AL2	0.8252
30	AL4	0.7753
31	AL5	0.9249
36	R_APP2	0.8079
50	UT3	0.7971
57	AA1	0.7631
61	R_AC2	0.9036
62	R_AC3	0.7605
63	R_AC4	0.8253
64	AI1	0.8023
65	AI2	0.8135
66	R_AI3	0.7973
73	SP3	0.7511

i	l_ICC.Item.i.	icc
80	R_AEI3	0.897
81	AEI4	0.7595
82	R_AEI5	0.8307

```
cat("Mean of 90 ICC values:", mean(l_ICC$icc), "\n
Standard deviation of 90 ICC values:", sd(l_ICC$icc))
```

```
## Mean of 90 ICC values: 0.6131859
##
## Standard deviation of 90 ICC values: 0.1716844
```

```
retranslations2 <- rbind(poor, fair)
retranslations2
```

```
##      i l_ICC.Item.i.      icc
## 1   6      HLB2 3.974896e-01
## 2  11      NA2 7.737791e-09
## 3  14      NA5 2.729955e-01
## 4  17      NB3 3.704218e-01
## 5  19      AAS2 3.980479e-01
## 6  27      AL1 3.864688e-01
## 7  34      AS3 3.431580e-01
## 8  40      R_UAA3 3.762496e-01
## 9  49      UT2 2.196373e-01
## 10 68      AT1 3.702409e-01
## 11 77      IIS4 2.570690e-01
## 12 83      UEP1 3.082087e-01
## 13 90      UAI4 3.478262e-01
## 14  2      HLA2 4.052138e-01
## 15  8      HLB4 5.147715e-01
## 16 12      NA3 5.364460e-01
## 17 16      NB2 5.097312e-01
## 18 20      AAS3 4.769843e-01
## 19 23      AU3 5.493124e-01
## 20 32      AS1 5.585574e-01
## 21 41      R_AE1 5.848160e-01
## 22 42      AE2 5.433129e-01
## 23 46      UE2 5.247076e-01
## 24 52      UAL2 4.401624e-01
## 25 55      UAL5 5.992643e-01
## 26 58      AA2 4.827585e-01
## 27 59      AA3 5.531281e-01
## 28 71      SP1 5.189791e-01
## 29 74      IIS1 5.918367e-01
## 30 75      IIS2 5.754961e-01
## 31 76      IIS3 4.384237e-01
## 32 78      AEI1 4.491019e-01
## 33 88      UAI2 4.860559e-01
## 34 89      UAI3 4.256030e-01
```

#Display all ASA question items which need to be re translated.