Correlations between the IPIP100 and ICAR scales in a large online sample

David Condon, Lorien Elleman, William Revelle
Northwestern University

This Sweave document is used to demonstrate the methods used to organize the data collected from the survey at 'test.personality-project.org' between April 4, 2006 and August 18, 2010. The output of this document - 'Kahuna06to10.rdata' - includes large data frames of the scale-level correlations, the item-level descriptive statistics, the proportion of participants by race/ethnicity, and sample size information. All these steps were completed on October 30, 2013 using the following:

```
[1] "R version 3.0.2 (2013-09-25)"
[1] "psych"
[1] "1.3.10.12"
```

Explanations and examples are given for each of the steps below, though readers of the pdf version should note that the Rnw version of this file contains many lines of `R` code which are not passed to the pdf.

Step 1: Load the Data

The commands in this section pull in data from the SAPA databases, remove participants who took the survey more than once with the same RID, drop participants who claim to be under 14 or over 90 years of age, and drop those participants who tell us that they've taken the survey before. Also note that one item was removed as we can't be certain that it was storing the data correctly.

Step 2: Merge Many Into Few (Data Sets) and Clean Them Up

The first part of this step puts together the data from several different tables. Note that the tables are dated (and merged chronologically).

---

contact: David Condon at davidcondon@u.northwestern.edu
Draft version of October 30, 2013

```
> # First merge the concomitantly administered B5 and IQ sets
> tempiqOrderedUnscrubbed1 <- merge(B5OrderedUnscrubbed1, iqOrderedUnscrubbed1[,
c((which(colnames(iqOrderedUnscrubbed1)=="RID")):ncol(iqOrderedUnscrubbed1))],
by.x ="RID", by.y = "RID", all = TRUE)
> tempiqOrderedUnscrubbed2 <- merge(B5OrderedUnscrubbed2, iqOrderedUnscrubbed2[,
c((which(colnames(iqOrderedUnscrubbed2)=="RID")):ncol(iqOrderedUnscrubbed2))],
by.x ="RID", by.y = "RID", all = TRUE)
> tempiqOrderedUnscrubbed3 <- merge(B5OrderedUnscrubbed3, iqOrderedUnscrubbed3[,
c((which(colnames(iqOrderedUnscrubbed3)=="RID")):ncol(iqOrderedUnscrubbed3))],
by.x ="RID", by.y = "RID", all = TRUE)
> # Now merge the tables across different administration periods
>
> TAOrderedUnscrubbed.merge1 <- merge(tempiqOrderedUnscrubbed1,
tempiqOrderedUnscrubbed2, all = TRUE, sort = FALSE)
> dim(TAOrderedUnscrubbed.merge1)
[1] 65826   141
> TAOrderedUnscrubbed.merge2 <- merge(TAOrderedUnscrubbed.merge1,
tempiqOrderedUnscrubbed3, all = TRUE, sort = FALSE)
> dim(TAOrderedUnscrubbed.merge2)
[1] 116558    141
> # Give it a slightly easier name
> TAOrderedUnscrubbed <- TAOrderedUnscrubbed.merge2
> # Clear out the memory in workspace before going on...
> rm(list=setdiff(ls(), "TAOrderedUnscrubbed"))
> #load("/sscc/home/d/dmc174/batch/DataEagre/UpdatedFunctions.rdata")
> load("/Users/DC/DCstuff/lab/SAPA/data/Eagre/UpdatedFunctions.rdata")
> gc()
           used  (Mb) gc trigger   (Mb)  max used   (Mb)
Ncells   612630  32.8    2023709  108.1   2361559  126.2
Vcells 22682276 173.1  145195293 1107.8 181478750 1384.6
```

Step 3: Scrub the Data

Now we are going to 'scrub' the data in many ways. Before doing so, it's important to inspect the descriptive statistics to identify weirdness (use 'describe()'). Based on this we know we need to do some re-ordering and scrubbing some strange values for a few of the IQ variables. Also, we drop B5 variable 'q_55' because of an error with data storage (iq_55 was somehow being stored as iq_10055 and q_55).

```
> # First, re-order the variables (two B5 variables are at the end before re-ordering due to administration error).
> # Also, drop q_55, RID, no_code.
>
> TAOrderedUnscrubbedFinal <- subset(TAOrderedUnscrubbed, select=c(gender, age,
race, education, q_76, q_108, q_124, q_128, q_132, q_140, q_146, q_150, q_177, q_194,
q_195, q_200, q_217, q_240, q_241, q_248, q_254, q_262, q_316, q_403, q_422, q_492,
q_493, q_497, q_530, q_609, q_619, q_626, q_690, q_698, q_712, q_815, q_819, q_838,
q_844, q_890, q_901, q_904, q_931, q_952, q_960, q_962, q_974, q_979, q_986, q_995,
q_1020, q_1041, q_1050, q_1053, q_1058, q_1083, q_1088, q_1090, q_1099, q_1114, q_1162,
q_1163, q_1180, q_1205, q_1206, q_1254, q_1255, q_1290, q_1333, q_1364, q_1374, q_1385,
q_1388, q_1392, q_1397, q_1410, q_1419, q_1422, q_1452, q_1479, q_1480, q_1483, q_1505,
q_1507, q_1585, q_1677, q_1683, q_1696, q_1705, q_1738, q_1742, q_1763, q_1768, q_1775,
q_1792, q_1803, q_1832, q_1861, q_1893, q_1913, q_1949, q_1964, q_1989, iq_10001, iq_10003,
iq_10004, iq_10005, iq_10006, iq_10007, iq_10009, iq_10011, iq_10013, iq_10014, iq_10016,
iq_10017, iq_10018, iq_10019, iq_10023, iq_10026, iq_10031, iq_10032, iq_10033, iq_10034,
iq_10035, iq_10036, iq_10039, iq_10042, iq_10043, iq_10044, iq_10045, iq_10046, iq_10047,
iq_10048, iq_10050, iq_10053, iq_10054, iq_10055, iq_10056))
> # Get an object which is the descriptive statistics
> # Before scrubbing the weird IQ items, let's identify the magnitude of the problem.
> UnscrubbedDescription <- describe(TAOrderedUnscrubbedFinal)
> # Table the responses for IQ items that look weird.
```

```
> temp <- rownames(subset(UnscrubbedDescription, max == 7))
> tableOfRespsWeirdItems <- t(apply(TAOrderedUnscrubbedFinal[,temp], 2, FUN=table))
> tableOfRespsWeirdItems
           0    1     2     3     4     5    6 7
iq_10004 1357 1310  2593  2938 19620   812  481 1
iq_10033 1477 3756  3903 16493   951  1885  401 1
iq_10035 1133 2751 13514 10441   523   240  589 1
iq_10036 1129 1386  3585 12138  1625  3053 6238 1
iq_10039  972  506   785 25771   169   223  728 1
iq_10044  737 2785  1514 19551   529  3820  284 1
iq_10055 1618  989  5690  4494  9111  2713 4430 2
iq_10056 1398 4779  2999  2744  3618 11212 2404 1
> # This is a minor issue.  We'll just drop those participants (only 9 of them).
>
> # Scrub the weird items with responses of 7 and race and education
> TAOrderedScrubbed <- scrub(TAOrderedUnscrubbedFinal,
where=c(rownames(subset(UnscrubbedDescription, max == 7))), isvalue=c(7))
> #Confirm successful scrub.
> #describe(TAOrderedScrubbed)
>
> # Here we are dropping the rows where participants skipped all of the IQ items.
This is not entirely necessary but we do it anyway (not many Ps are lost).
>
> good.iq <- rowSums(TAOrderedScrubbed[,c((which(colnames(TAOrderedScrubbed)=="iq_10001")):
(which(colnames(TAOrderedScrubbed)=="iq_10056")))],na.rm=TRUE)
> TAOrderedScrubbed <- TAOrderedScrubbed[good.iq>0,]
> dim(TAOrderedScrubbed)        # Participants who did not answer any IQ items removed.
[1] 114366    138
> #Create a subset--TA.front--and scrub for values of 0 for disposition questions.
> TA.front <- TAOrderedScrubbed[,c(1:((which(colnames(TAOrderedScrubbed)=="iq_10001"))-1))]
> TA.front <- scrub(TA.front, where = c((which(colnames(TA.front)=="education")+1): ncol(TA.front)), isvalue = 0)
> TA.front <- scrub(TA.front, where = c((which(colnames(TA.front)=="race"))), isvalue = 14)
> #Create the other subset, iq.data, and then score it (converts to 0s and 1s).
> iq.data <- TAOrderedScrubbed[,c((which(colnames(TAOrderedScrubbed)=="iq_10001")):ncol(TAOrderedScrubbed))]
> iq.correct <- c(4, 4, 4, 2, 3, 6, 4, 1, 5, 2, 4, 4, 5, 6, 5, 5, 1, 1,
3, 4, 2, 3, 3, 5, 4, 3, 5, 2, 2, 4, 5, 3, 1, 4, 5)
> iq.matrix <- score.multiple.choice(iq.correct,iq.data,score=FALSE)
> taFinal <- data.matrix(data.frame(TA.front,iq.matrix))
> rm(list=setdiff(ls(), "taFinal"))
> #load("/sscc/home/d/dmc174/batch/DataEagre/UpdatedFunctions.rdata")
> load("/Users/DC/DCstuff/lab/SAPA/data/Eagre/UpdatedFunctions.rdata")
> gc()
           used  (Mb) gc trigger  (Mb)  max used   (Mb)
Ncells   735665  39.3    2023709 108.1   2361559  126.2
Vcells 22318478 170.3  116156234 886.3 181478750 1384.6
```

Step 4: Get the descriptives and race table

Getting the means and standard deviations by item is easy enough. Describing the proportion of participants by race/ethnicity is a bit more tricky. For one thing, many participants did not provide these data (it is unclear whether this is because they were not prompted or because they opted out... presumably the former). For the cases where we do have data, we have to put labels on the values.

```
> MeansSDs <-describe(taFinal)[,3:4]
> raceTable <- table(taFinal[,"race"])
> race.text <- c("African American", "Chinese", "Japanese", "Korean", "Philipino", "Indian/Pakistani",
"Other Asian", "Latino", "Mexican", "Puerto Rican", "Native American", "Pacific Islander",
"White/Caucasian", "Other")
> names(raceTable) <- race.text
```

```
> totalN <- describe(taFinal)[1,2]
> raceN <- sum(raceTable)
> # proportion responding to race
> raceResp <- raceN/totalN
> # proportion not responding to race
> raceNotResp <- 1-raceResp
> # Show the values
> totalN
[1] 114263
> raceN
[1] 83299
> raceResp
[1] 0.7290111
> raceNotResp
[1] 0.2709889
> # Race by proportion of those responding
> raceProportions <- round(raceTable/raceN,4)
> raceProportions
African American          Chinese         Japanese           Korean        Philipino
          0.0765           0.0143           0.0032           0.0064           0.0078
Indian/Pakistani      Other Asian           Latino          Mexican     Puerto Rican
          0.0060           0.0072           0.0264           0.0271           0.0067
Native American Pacific Islander  White/Caucasian            Other
         0.0089           0.0040           0.7670           0.0386
```

Step 5: Get the Correlations

Now get the covariance/correlation matrix and clean it up. There are two possible methods for doing this. Pearson's is faster but the tetrachoric/polychoric is preferred. So we do the latter.

```
> taFinal <- subset(taFinal, select = -c(race))
> ta.cov <- mixed.cor(taFinal)
```

```
> ta.cor <- ta.cov$rho
> # Now we want to use the inter-item correlations to generate the inter-scale correlations.  We need a key.
> Keylist <- list(gender="gender", age="age", education="education",
+ agreeableness=c("-q_140", "-q_146", "q_150", "-q_195", "-q_200", "q_217", "-q_838", "q_844", "q_1041",
"q_1053", "q_1162", "-q_1163", "q_1206", "q_1364", "q_1385", "q_1419", "q_1705", "q_1763", "q_1792",
"q_1832"),
+ conscientiousness=c("q_76", "q_124", "q_530", "q_619", "-q_626", "-q_904", "q_931", "q_962", "-q_1254",
"-q_1255", "q_1290", "q_1333", "q_1374", "-q_1397", "q_1422", "-q_1452", "-q_1483", "q_1507", "-q_1696",
"-q_1949"),
+ extraversion=c("-q_241", "q_254", "q_262", "-q_403", "-q_690", "q_698", "-q_712", "q_815", "q_819",
"-q_901", "-q_1114", "-q_1180", "q_1205", "q_1410", "-q_1480", "q_1742", "q_1768", "q_1803", "-q_1913"),
+ stability=c("-q_108", "q_177", "q_248", "-q_497", "-q_890", "-q_952", "-q_960", "-q_974", "-q_979", "-q_986",
"-q_995", "-q_1020", "-q_1099", "-q_1479", "-q_1505", "q_1585", "q_1677", "q_1683", "-q_1775", "-q_1989"),
+ intellect=c("q_128", "q_132", "-q_194", "q_240", "-q_316", "q_422", "q_492", "q_493", "-q_609", "q_1050",
"q_1058", "-q_1083", "-q_1088", "q_1090", "q_1388", "q_1392", "q_1738", "-q_1861", "q_1893", "-q_1964"),
+ ICAR60=c("iq_10001",  "iq_10003", "iq_10004",  "iq_10005",  "iq_10006",  "iq_10007",  "iq_10009",
"iq_10011",  "iq_10013",  "iq_10014",  "iq_10016",  "iq_10017",  "iq_10018",  "iq_10019",  "iq_10023",
"iq_10026",  "iq_10031",  "iq_10032",  "iq_10033",  "iq_10034",  "iq_10035",  "iq_10036",  "iq_10039",
"iq_10042",  "iq_10043",  "iq_10044",  "iq_10045",  "iq_10046",  "iq_10047",  "iq_10048",  "iq_10050",
"iq_10053",  "iq_10054",  "iq_10055",  "iq_10056"),
+ LNiq= c("iq_10001",  "iq_10003", "iq_10005",  "iq_10006",  "iq_10007", "iq_10033",  "iq_10034",  "iq_10035"),
+ MRiq= c("iq_10043",  "iq_10044",  "iq_10045",  "iq_10046",  "iq_10047",  "iq_10048",  "iq_10050",
"iq_10053",  "iq_10054",  "iq_10055",  "iq_10056"),
+ VRiq= c("iq_10004", "iq_10009",  "iq_10011",  "iq_10013",  "iq_10014",  "iq_10016",  "iq_10017",
"iq_10018",  "iq_10019",  "iq_10023",  "iq_10026",  "iq_10031",  "iq_10032",  "iq_10036",  "iq_10039",
```

```
"iq_10042"))
> master.key <- make.keys(taFinal, Keylist)
> allscores <- cluster.cor(master.key, ta.cor)
> all.scores <- round((allscores$cor),2) # This is the uncorrected correlation matrix
> all.scores.corrected <- round((allscores$corrected),2) # Uncorrected lower left, corrected upper right,
reliabilities on the diagonal.
```

Step 6: Get Sample Size Information

Providing an answer to the eventual question regarding sample size is quite complicated. In this section we derive the effective sample size for each of the scale-level correlations by using the bootstrapped confidence intervals of the correlations. While the range of effective Ns is quite large, the mean and median are roughly equivalent at about 70,500.
The last step is to save the relevant objects for distribution and later use.

```
> taFinal.CIs <- cor.ci(taFinal, master.key, n.iter=100, poly=FALSE)
> effectiveN <- (1/taFinal.CIs$sds)^2
> effectiveNsummary <- round(describe(effectiveN)[c(3:5,8,9)],0)
> effectiveNsummary
   mean    sd median   min    max
1 70864 28255  70505 12680 130371
> save(raceProportions, raceNotResp, raceResp, raceN, totalN, all.scores, all.scores.corrected,
MeansSDs, taFinal.CIs, effectiveN, effectiveNsummary,
file="/Users/DC/DCstuff/lab/MPSP/PandCAmeta/Kahuna06to10.rdata")
```