Psych: A Swiss Army Knife for psychology

William Revelle

Department of Psychology
Northwestern University
Evanston, Illinois USA

July, 2023

Outline

The psych and psychTools packages

1. *psych* and *psychTools* have been developed to help further research in personality and individual differences.
2. Like a Swiss Army Knife, *psych* is not the best tool for anything, but it is a very helpful tool for many things.
3. *psych* is particularly aimed at the researcher in personality and individual differences.
4. Like core R *psych* helps researchers do open source science.
5. It is meant to be a relatively “light” package, in that it does not have many dependencies.
6. Unlike many other packages, the Help pages and Vignettes are fairly extensive (some would say wordy).

Installing the psych package ($\geq$ 2.3.6)

R code

```
#if you have not already done so, you first install the package
install.packages("psych",dependencies=TRUE)

library(psych)  #you need to do this every time you start R

#or automate the library(psych) call
#by creating and saving a function

.First <- function() {library(psych)}
quit() #with save option

#start R and psych will be automatically loaded
sessionInfo()  #will tell you what version you are using
```

```
Good morning Bill.
Are you ready to have some fun?


> sessionInfo()
 version 4.3.1 (2023-06-16)
Platform: aarch64-apple-darwin20 (64-bit)
Running under: macOS Ventura 13.4.1
```

To see the dependencies

R code

```
sessionInfo()
```

```
sessionInfo()
R version 4.3.1 (2023-06-16)
Platform: aarch64-apple-darwin20 (64-bit)
Running under: macOS Ventura 13.4.1

Matrix products: default
BLAS:   /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/lib/libRblas.0.dylib
LAPACK: /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/lib/libRlapack.dylib;

Random number generation:
 RNG:     Mersenne-Twister
 Normal:  Inversion
 Sample:  Rounding

locale:
[1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
time zone: America/Chicago
tzcode source: internal

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] psychTools_2.3.6 psych_2.3.6

loaded via a namespace (and not attached):
[1] compiler_4.3.1 tools_4.3.1    parallel_4.3.1 foreign_0.8-84 nlme_3.1-162   mnormt_2.1.1
```

Show all the functions in the psych package

`objects("package:psych")`

```
  [1] "%+%"                   "acs"                   "alpha"                 "alpha.ci"
  [5] "alpha2r"               "anova.psych"           "AUC"                   "autoR"
  [9] "bassAckward"           "bassAckward.diagram"   "Bechtoldt"             "Bechtoldt.
 [13] "Bechtoldt.2"           "bestItems"             "bestScales"            "bfi"
 [17] "bfi.keys"              "bi.bars"               "bifactor"              "bigCor"
 [21] "biplot.psych"          "biquartimin"           "biserial"              "block.rand
 [25] "bock.table"            "cattell"               "cd.validity"           "char2numer
 [29] "Chen"                  "chi2r"                 "circ.sim"              "circ.sim.p
 [33] "circ.simulation"       "circ.tests"            "circadian.cor"         "circadian.
 [37] "circadian.linear.cor"  "circadian.mean"        "circadian.phase"       "circadian.
 [41] "circadian.sd"          "circadian.stats"       "circular.cor"          "circular.m
 [45] "cluster.cor"           "cluster.fit"           "cluster.loadings"      "cluster.pl
 [49] "cluster2keys"          "cohen.d"               "cohen.d.by"            "cohen.d.ci
 [53] "cohen.kappa"           "cohen.profile"         "comorbidity"           "con2cat"
 [57] "congeneric.sim"        "congruence"            "cor.ci"                "cor.plot"
...
[441] "superMatrix"           "t2d"                   "t2r"                   "table2df"
[445] "table2matrix"          "tableF"                "Tal_Or"                "Tal.Or"
[449] "target.rot"            "TargetQ"               "TargetT"               "tenberge"
[453] "test.all"              "test.irt"              "test.psych"            "testRetest
[457] "tetrachoric"           "thurstone"             "Thurstone"             "Thurstone.
[461] "Thurstone.33G"         "Thurstone.9"           "topBottom"             "tr"
[465] "Tucker"                "unidim"                "validityItem"          "varimin"
[469] "vgQ.bimin"             "vgQ.targetQ"           "vgQ.varimin"           "violin"
[473] "violinBy"              "vss"                   "VSS"                   "VSS.parall
[477] "VSS.plot"              "VSS.scree"             "VSS.sim"               "VSS.simula
[481] "West"                  "winsor"                "winsor.mean"           "winsor.mea
[485] "winsor.sd"             "winsor.var"            "withinBetween"         "wkappa"
[489] "Yule"                  "Yule.inv"              "Yule2phi"              "Yule2phi.m
[493] "Yule2poly"             "Yule2poly.matrix"      "Yule2tetra"            "YuleBonett
[497] "YuleCor"
```

Objects in *psychTools*

```
 objects(package:psychTools)
 [1] "ability"                   "ability.keys"              "affect"
 [4] "all.income"                "Athenstaedt"               "Athenstaedt.dictionary"
 [7] "Athenstaedt.keys"          "bfi"                       "bfi.adjectives.dictionary"
[10] "bfi.adjectives.keys"       "bfi.dictionary"            "bfi.keys"
[13] "big5.100.adjectives"       "big5.adjectives.keys"      "blant"
[16] "blot"                      "burt"                      "cities"
[19] "city.location"             "colom"                     "colom.ed0"
[22] "colom.ed1"                 "colom.ed2"                 "colom.ed3"
[25] "combineMatrices"           "cor2latex"                 "cubits"
[28] "cushny"                    "Damian"                    "df2latex"
[31] "dfOrder"                   "eminence"                  "epi"
[34] "epi.bfi"                   "epi.dictionary"            "epi.keys"
[37] "epiR"                      "fa2latex"                  "fileCreate"
[40] "fileScan"                  "filesInfo"                 "filesList"
[43] "galton"                    "GERAS.dictionary"          "GERAS.items"
[46] "GERAS.keys"                "GERAS.scales"              "globalWarm"
[49] "heights"                   "holzinger.dictionary"      "holzinger.raw"
[52] "holzinger.swineford"       "ICC2latex"                 "income"
[55] "iqitems"                   "irt2latex"                 "msq"
[58] "msq.keys"                  "msqR"                      "neo"
[61] "omega2latex"               "peas"                      "Pollack"
[64] "read.clipboard"            "read.clipboard.csv"        "read.clipboard.fwf"
[67] "read.clipboard.lower"      "read.clipboard.tab"        "read.clipboard.upper"
[70] "read.file"                 "read.file.csv"             "read.https"
[73] "recode"                    "sai"                       "sai.dictionary"
[76] "sat.act"                   "Schutz"                    "selectBy"
[79] "Spengler"                  "Spengler.stat"             "spi"
[82] "spi.dictionary"            "spi.keys"                  "splitBy"
[85] "tai"                       "USAF"                      "veg"
[88] "vJoin"                     "write.file"                "write.file.csv"
[91] "zola"                      "zola.dictionary"           "zola.keys"
```

Vignettes and How To's

- How to's and Vignettes
    1. An introduction to the *psych* package: Part I:.
    2. Intro:Part II: Scale construction and psychometrics
    3. Installing Rand the *psych* package
    4. Using R and psych to find ω
    5. How To: Use *psych* for factor analysis and data reduction
    6. Using R to score personality scales
    7. Using *psych* for regression and mediation analysis
- User manual for *psych*
- User manual for *psychTools*
- Help files for *psych*
- Help files for *psychTools*

Get your data: using `read.file` or `read.clipboard`

From a website: define the file name

R code

```
fn <- "https://personality-project.org/r/datasets/glbwarm.sav"
fn  #show it to check
fn
[1] "https://personality-project.org/r/datasets/glbwarm.sav"
mydata <- read.file(fn)
```

From a local file: find the file using `read.file`

R code

```
> my.data <- read.file() #will open a search window, read the file
#depending upon the suffix, will read .sav, .csv, .txt,
           .rds, .rDa, etc.
```

From the clipboard: (first, go to the remote site, copy to the clipboard and then use the `read.clipboard` function).

R code

```
mydata <- read.clipboard()     #or
mydata <- read.clipboard.tab()  #if an excel file
my.data <- read.clipboard.csv() #if a tab delimited file
```

(This example data set can also be accessed directly in `glbwarm`.)

R code

```
dim(mydata)  #how many rows and columns?
headTail(mydata) #Show the top and bottom n rows and columns from c1 t
describe(mydata)  #basic descriptive statistics
```

```
 dim(mydata)  #how many rows and columns?
[1] 815   7
> headTail(mydata) #Show the top and bottom n rows and columns from c1 to c2
    govact posemot negemot ideology age sex partyid
1      3.6    3.67    4.67        6  61   0       2
2        5       2    2.33        2  55   0       1
3      6.6    2.33    3.67        1  85   1       1
4        1       5       5        1  59   0       1
...    ...     ...     ...      ... ... ...     ...
812    3.4       1       1        7  67   0       3
813    1.6    3.67    1.67        7  72   1       3
814    5.4    2.67    3.33        6  36   0       2
815    5.4    5.33       6        4  82   1       1
> describe(mydata)  #basic descriptive statistics
         vars   n  mean    sd median trimmed   mad min max range  skew kurtosis   se
govact      1 815  4.59  1.36   4.80    4.68  1.19   1   7     6 -0.63     0.22 0.05
posemot     2 815  3.13  1.35   3.00    3.11  1.48   1   6     5  0.09    -0.85 0.05
negemot     3 815  3.56  1.53   3.67    3.58  1.97   1   6     5 -0.15    -1.07 0.05
ideology    4 815  4.08  1.51   4.00    4.07  1.48   1   7     6  0.03    -0.43 0.05
age         5 815 49.54 16.33  51.00   49.66 19.27  17  87    70 -0.07    -1.03 0.57
sex         6 815  0.49  0.50   0.00    0.49  0.00   0   1     1  0.05    -2.00 0.02
partyid     7 815  1.88  0.87   2.00    1.85  1.48   1   3     2  0.23    -1.63 0.03
>
```

headTail of a bigger local file

R code

```
dim(msqR)
headTail(msqR,top=4,bottom=6,from=78, to=88)
```

```
[1] 6411   88
headTail(msqR,top=4,bottom=6,from=78, to=88)
     Lie Sociability Impulsivity gender TOD drug film time  id form study
1      3           7           1      2   9    2 <NA>    1   1    2  AGES
2      3           9           4      2   9    1 <NA>    1   2    2  AGES
3      4           3           1      1   9    1 <NA>    1   3    2  AGES
4      1          11           4      2   9    2 <NA>    1   4    2  AGES
...  ...         ...         ...    ... ...  ...  ...  ... ...  ...  <NA>
3941   2           9           6   <NA>   9    2    4    2 195    2  XRAY
3942   3           3           5   <NA>   9    1    2    2 196    2  XRAY
3943   2           7           2   <NA>   9    2    1    2 197    2  XRAY
3944   0          12           5   <NA>   9    1    4    2 198    2  XRAY
3945   3          11           3   <NA>   9    1    3    2 199    2  XRAY
3946   0           2           4   <NA>   9    2    4    2 200    2  XRAY
```

Notice that rowname although unique is not the line number

Descriptives by a grouping variable

R code

```
describeBy(mydata~sex)
```

```
 Descriptive statistics by group
sex: 0
         vars   n  mean    sd median trimmed   mad min max range  skew kurtosis   se
govact      1 417  4.72  1.16    4.8    4.77  1.19   1   7     6 -0.52     0.55 0.06
posemot     2 417  3.03  1.39    3.0    3.00  1.48   1   6     5  0.17    -0.98 0.07
negemot     3 417  3.73  1.45    4.0    3.79  1.48   1   6     5 -0.26    -0.83 0.07
ideology    4 417  3.89  1.44    4.0    3.87  1.48   1   7     6  0.05    -0.29 0.07
age         5 417 46.90 14.95   44.0   46.77 16.31  18  83    65  0.13    -0.90 0.73
sex         6 417  0.00  0.00    0.0    0.00  0.00   0   0     0   NaN      NaN 0.00
partyid     7 417  1.79  0.86    2.0    1.74  1.48   1   3     2  0.41    -1.52 0.04
-----------------------------------------------------------------------
sex: 1
         vars   n  mean    sd median trimmed   mad min max range  skew kurtosis   se
govact      1 398  4.45  1.53   4.60    4.55  1.48   1   7     6 -0.56    -0.31 0.08
posemot     2 398  3.23  1.30   3.33    3.23  1.48   1   6     5  0.04    -0.68 0.07
negemot     3 398  3.37  1.59   3.67    3.36  1.98   1   6     5 -0.01    -1.25 0.08
ideology    4 398  4.29  1.56   4.00    4.29  1.48   1   7     6 -0.05    -0.56 0.08
age         5 398 52.30 17.25  55.00   52.85 19.27  17  87    70 -0.33    -1.01 0.86
sex         6 398  1.00  0.00   1.00    1.00  0.00   1   1     0   NaN      NaN 0.00
partyid     7 398  1.98  0.87   2.00    1.98  1.48   1   3     2  0.04    -1.67 0.04
```

Correlations using `lowerCor`

`lowerCor` is a nice example of the power of R to nest functions. It is just a call to `cor` with the use="pairwise" option followed by a call to `lowerMat` which "prettifies" a correlation matrix.

R code

```
R<- lowerCor(mydata)   #returns R invisibly
```

```
lowerCor(mydata)
         govct posmt negmt idlgy age   sex   prtyd
govact    1.00
posemot   0.04  1.00
negemot   0.58  0.13  1.00
ideology -0.42 -0.03 -0.35  1.00
age      -0.10  0.04 -0.06  0.21  1.00
sex      -0.10  0.07 -0.12  0.13  0.17  1.00
partyid  -0.36 -0.04 -0.32  0.62  0.15  0.11  1.00

 R  #show R (if you want to use if for something else.  Note that it is not rounded.
              govact     posemot     negemot    ideology         age         sex     partyid
govact    1.00000000  0.04302895  0.57774582 -0.41831995 -0.09713873 -0.09861854 -0.36039647
posemot   0.04302895  1.00000000  0.12792202 -0.02937618  0.04235193  0.07429449 -0.03577099
negemot   0.57774582  0.12792202  1.00000000 -0.34878643 -0.05689493 -0.11735643 -0.32419141
ideology -0.41831995 -0.02937618 -0.34878643  1.00000000  0.21240565  0.13288895  0.61945381
age      -0.09713873  0.04235193 -0.05689493  0.21240565  1.00000000  0.16553039  0.15443184
sex      -0.09861854  0.07429449 -0.11735643  0.13288895  0.16553039  1.00000000  0.10875960
partyid  -0.36039647 -0.03577099 -0.32419141  0.61945381  0.15443184  0.10875960  1.00000000
```

Correlations using `corr.test`

R code

```
corr.test(mydata)
```

```
Call:corr.test(x = mydata)
Correlation matrix
         govact posemot negemot ideology   age   sex partyid
govact     1.00    0.04    0.58    -0.42 -0.10 -0.10   -0.36
posemot    0.04    1.00    0.13    -0.03  0.04  0.07   -0.04
negemot    0.58    0.13    1.00    -0.35 -0.06 -0.12   -0.32
ideology  -0.42   -0.03   -0.35     1.00  0.21  0.13    0.62
age       -0.10    0.04   -0.06     0.21  1.00  0.17    0.15
sex       -0.10    0.07   -0.12     0.13  0.17  1.00    0.11
partyid   -0.36   -0.04   -0.32     0.62  0.15  0.11    1.00
Sample Size
[1] 815
Probability values (Entries above the diagonal are adjusted* for multiple tests.)
         govact posemot negemot ideology  age  sex partyid
govact     0.00    0.88     0.0     0.00 0.04 0.04    0.00
posemot    0.22    0.00     0.0     0.88 0.88 0.20    0.88
negemot    0.00    0.00     0.0     0.00 0.52 0.01    0.00
ideology   0.00    0.40     0.0     0.00 0.00 0.00    0.00
age        0.01    0.23     0.1     0.00 0.00 0.00    0.00
sex        0.00    0.03     0.0     0.00 0.00 0.00    0.02
partyid    0.00    0.31     0.0     0.00 0.00 0.00    0.00

 To see confidence intervals of the correlations, print with the short=FALSE option'
```

*Adjustment using the Holm (1979) correction for multiple tests

long output from `corr.test` gives the normal theory CI

R code

```
print(corr.test(mydata),short=FALSE)
```

```
Confidence intervals based upon normal theory.  To get bootstrapped values, try cor.ci
            raw.lower raw.r raw.upper raw.p lower.adj upper.adj
govct-posmt     -0.03  0.04      0.11  0.22     -0.04      0.13
govct-negmt      0.53  0.58      0.62  0.00      0.50      0.64
govct-idlgy     -0.47 -0.42     -0.36  0.00     -0.50     -0.33
govct-age       -0.16 -0.10     -0.03  0.01     -0.19      0.00
govct-sex       -0.17 -0.10     -0.03  0.00     -0.19      0.00
govct-prtyd     -0.42 -0.36     -0.30  0.00     -0.45     -0.27
posmt-negmt      0.06  0.13      0.19  0.00      0.03      0.22
posmt-idlgy     -0.10 -0.03      0.04  0.40     -0.10      0.04
posmt-age       -0.03  0.04      0.11  0.23     -0.04      0.13
posmt-sex        0.01  0.07      0.14  0.03     -0.02      0.17
posmt-prtyd     -0.10 -0.04      0.03  0.31     -0.11      0.04
negmt-idlgy     -0.41 -0.35     -0.29  0.00     -0.44     -0.25
negmt-age       -0.13 -0.06      0.01  0.10     -0.15      0.03
negmt-sex       -0.18 -0.12     -0.05  0.00     -0.21     -0.02
negmt-prtyd     -0.38 -0.32     -0.26  0.00     -0.41     -0.23
idlgy-age        0.15  0.21      0.28  0.00      0.11      0.31
idlgy-sex        0.06  0.13      0.20  0.00      0.03      0.23
idlgy-prtyd      0.58  0.62      0.66  0.00      0.55      0.68
age-sex          0.10  0.17      0.23  0.00      0.06      0.26
age-prtyd        0.09  0.15      0.22  0.00      0.05      0.25
sex-prtyd        0.04  0.11      0.18  0.00      0.01      0.20
```

Adjusted cis are given with Holm (1979) adjustment

Correlations with “magic astericks”

R code

```
print(corr.test(mydata)$stars,quote=FALSE)
```

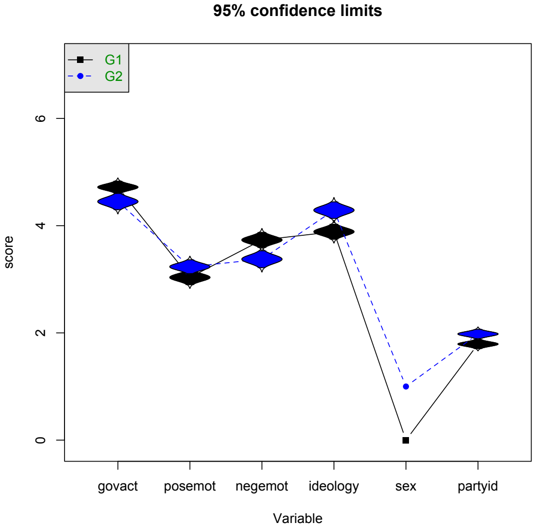
```
print(corr.test(mydata)$stars,quote=FALSE)
         govact   posemot negemot  ideology age     sex     partyid
govact   1***     0.04    0.58***  -0.42*** -0.1*   -0.1*   -0.36***
posemot  0.04     1***    0.13**   -0.03    0.04    0.07    -0.04
negemot  0.58***  0.13*** 1***     -0.35*** -0.06   -0.12** -0.32***
ideology -0.42*** -0.03   -0.35*** 1***     0.21*** 0.13**  0.62***
age      -0.1**   0.04    -0.06    0.21***  1***    0.17*** 0.15***
sex      -0.1**   0.07*   -0.12*** 0.13***  0.17*** 1***    0.11*
partyid  -0.36*** -0.04   -0.32*** 0.62***  0.15*** 0.11**  1***
```

Once again, the p values above the diagonal are adjusted using the Holm (1979) correction for multiple tests.

Graphical displays of data

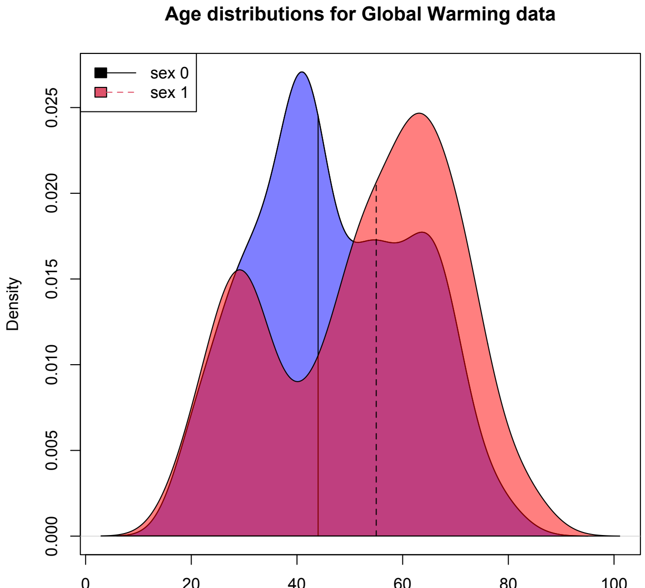
1. Can show basic means and ranges (`error.bars` and `error.bars.by`)
2. Can show correlations using `pairs.panels`, `corPlot`
3. `densityBy` for distribution information.
4. `scatterHist` to show bivariate plots with densities by group
5. `cohen.d` combines with `error.dots` to show effect sizes and confidence intervals

Showing group differences using `error.bars.by`



```
error.bars.by(mydata[-5],"sex",by.var=FALSE,ylab="score",xlab="Variable")
```

Global warming data set – age by sex



Age distributions for Global Warming data

Finding the effect of caffeine on mood

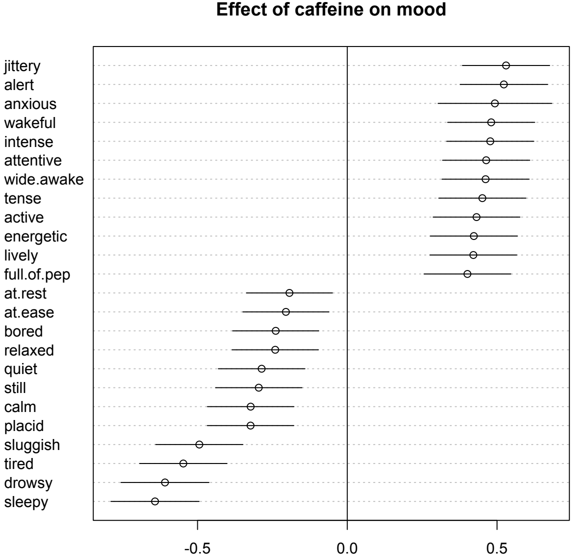
1. What is the effect of caffeine on motivational/emotional state?
2. Motivational State Questionnaire (Revelle & Anderson, 1998) was given to participants before and after caffeine/movie/stress manipulations
3. Data are pooled over 10 years of data ($>$ 50 studies) in the PMC lab and available as the msqR data set
4. Here we show how to select cases and find Cohen d (Cohen, 1988)

R code

```
table(msqR$time)
msq2 <- selectBy(msqR,"time=2") #just the time 2 data
msqd <- msq2[c(1:70,83)]   #just the mood and drug data
cd <- cohen.d(msqd~drug)  #find the Cohen d
error.dots(cd)  #show the top and bottom 10 items
```

```
 table(msqR$time)
   1    2    3    4 
3032 2086 1112  181 
> msq2 <- selectBy(msqR,"time=2") #just the time 2 data
> msqd <- msq2[c(1:70,83)]   #just the mood and drug data
> cd <- cohen.d(msqd~drug)  #find the Cohen d
> error.dots(cd)  #show the top and bottom 10 items
> summary(cd)
Multivariate (Mahalanobis) distance between groups 1.13
```

Cohen d for caffeine/placebo on msqR data



Showing sex differences in behavior

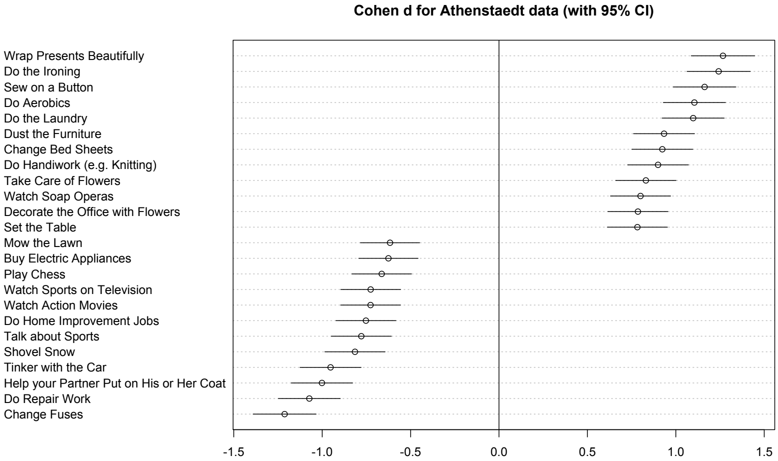
1. Athenstaedt (2003) examined Gender Role Self-Concept. She reports two independent dimensions of Male and Female behaviors.
2. While there are large gender/sex differences on both of these dimensions, the two represent independent factors!
3. Eagly & Revelle (2022) used these data to explore the power of aggregation when examining sex differences.
4. Included as an example of various graphical displays.

R code

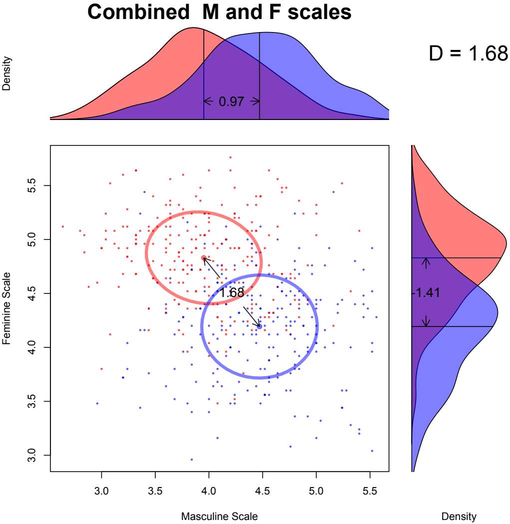
```
#show error dots and ci
cd <- cohen.d(Athenstaedt[2:75], group="gender",
          dictionary=Athenstaedt.dictionary)
error.dots(cd, main="Cohen d for Athenstaedt data (with 95% CI)")
abline(v=0)

#show scatter plots and density
scatterHist(Femininity ~ Masculinity + gender, data =Athenstaedt,
cex.point=.4,smooth=FALSE, correl=FALSE,d.arrow=TRUE,
   col=c("red","blue"), lwd=4,  cex.main=1.5,
    main="Scatter Plot and Density",cex.axis=2)
```

Male/female differences on GERAS items

Athenstaedt (2003)



Male/female differences on two scales



$r_{wg} = -.05, r = -.29$ (Athenstaedt, 2003; Eagly & Revelle, 2022)

Factor analysis, Cluster Analysis, Principal Components

1. Psychological data is typically of a high dimensionality.
2. One solution to this problem is the factor model which interprets observed manifest observed variables in terms of unobservable, latent variables. At the data level this is of course:
$$X_i = \Lambda \boldsymbol{x_k} + \Theta \boldsymbol{x}_u \tag{1}$$
3. At the level of covariances or correlations this is
$$\boldsymbol{C} \approx \boldsymbol{\Lambda\Lambda}' + \boldsymbol{\Theta}^2. \tag{2}$$
4. For a fixed number of factors and fixed values of $\boldsymbol{\Theta}^2$ this is solveable as a simple eigen value decomposition.
5. However, the problem of how many factors is difficult (there is no one right answer).

Factor analysis as an iterative procedure

1. An initial estimate of communalities ($1 - \mathbf{\Theta}^2$)
2. Find the eigen vectors ($\boldsymbol{F}$) of $\boldsymbol{R} - \mathbf{\Theta}^2$
3. Find the residuals of $\boldsymbol{R} - \boldsymbol{F}'\boldsymbol{F}$
    - ML for maximum likelihood
    - minres for minimum residual (default)
    - pa for principal factor
    - ...
4. Set the new communalities to diagonal of $\boldsymbol{F}'\boldsymbol{F}$
5. Iterate until communalities don't change or until sum of squared residuals is a minimum or until Maximum Likelihood estimate is minimized.
6. If rotation (orthogonal) or transformation (oblique) apply the chosen algorithm.
    - "none", "varimax", "quartimax", "BentlerT", "equamax", "varimin", "geominT" and "bifactor" are orthogonal rotations.
    - "Promax", "promax", "oblimin", "simplimax", "bentlerQ", "geominQ", "biquartimin" and "cluster" are oblique transformations

How many factors – no right answer, one wrong answer

1. Statistical
   - Extracting factors until the χ^2 of the residual matrix is not significant.
   - Extracting factors until the change in χ^2 from factor n to factor n+1 is not significant.
2. Rules of Thumb
   - Parallel: Extracting factors until the eigenvalues of the real data are less than the corresponding eigenvalues of a random data set of the same size (*parallel analysis*) `fa.parallel`
   - Plotting the magnitude of the successive eigenvalues and applying the *scree test*. `scree`
3. Interpretability
   - Extracting factors as long as they are interpretable.
   - Using the *Very Simple Structure* Criterion (`VSS`)
   - Using the Minimum Average Partial criterion (MAP).
4. Eigen Value of 1 rule (The worst rule)

`nfactors` applies many of these procedures

The number of factors problem is easy and hard

No best rule, one worst rule

"Solving the number of factors problem is easy, I do it everyday before breakfast. But knowing the right solution is harder." (attributed to Henry Kaiser by Horn & Engstrom (1979))

1. Parallel analysis (Extract factors until the eigen values are less than those of a random matrix).
   - Although a good rule for 100-500 subjects, this will not do as well with many ($>$ 1000) subjects.
2. Velicer (1976) Mininum Average Partial (MAP) is pretty good
3. For items, the Very Simple Structure (VSS) (Revelle & Rocklin, 1979) criterion is pretty good.
4. Multiple statistical tests, many have problems with sample size.
   - If you want few factors, run few subjects
   - If you want many factors, run many subjects
5. One worst rule is the eigen value of 1.0 rule.

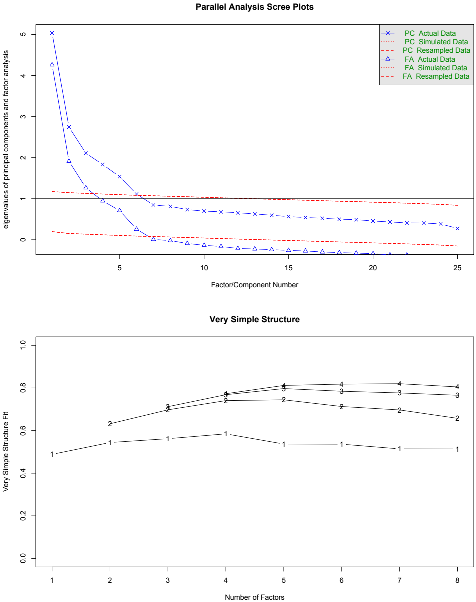
fa.parallel

R code

```
fa.parallel(bfi[1:25])
vss(bfi[1:25])
```

```
 fa.parallel(bfi[1:25])
Parallel analysis suggests that the number of factors =  6
       and the number of components =  6

Very Simple Structure
Call: vss(x = bfi[1:25])
VSS complexity 1 achieves a maximimum of 0.58  with  4  factors
VSS complexity 2 achieves a maximimum of 0.74  with  5  factors

The Velicer MAP achieves a minimum of 0.01  with  5  factors
BIC achieves a minimum of  -513.09  with  8  factors
Sample Size adjusted BIC achieves a minimum of  -106.39  with  8  factors
```

`fa.parallel` **and** `vss`



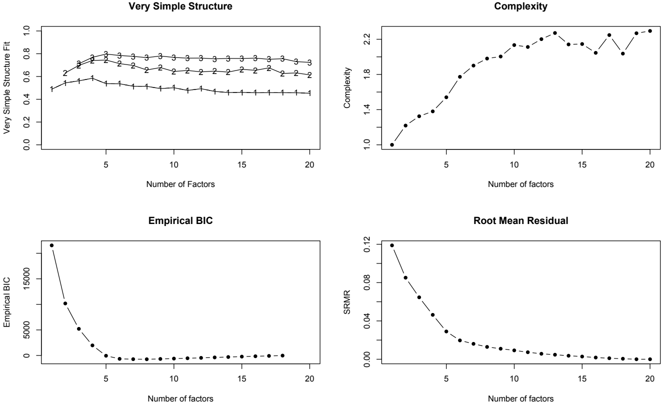
“The number of factors problem will break your heart”

R code

```
nfactors(bfi[1:25]
```

```
nfactors(bfi[1:25])

Number of factors
Call: vss(x = x, n = n, rotate = rotate, diagonal = diagonal, fm = fm,
    n.obs = n.obs, plot = FALSE, title = title, use = use, cor = cor)
VSS complexity 1 achieves a maximimum of 0.58  with  4  factors
VSS complexity 2 achieves a maximimum of 0.74  with  5  factors
The Velicer MAP achieves a minimum of 0.01  with  5  factors
Empirical BIC achieves a minimum of  -737.9  with  8  factors
Sample Size adjusted BIC achieves a minimum of  -205.18  with  12  factors

Statistics by number of factors
   vss1 vss2   map dof   chisq     prob sqresid  fit RMSEA  BIC SABIC complex  eChisq
1  0.49 0.00 0.024 275 1.2e+04  0.0e+00    25.9 0.49 0.123 9680 10554     1.0 2.4e+04
2  0.54 0.63 0.018 251 7.4e+03  0.0e+00    18.6 0.63 0.101 5370  6168     1.2 1.2e+04
3  0.56 0.70 0.017 228 5.1e+03  0.0e+00    14.6 0.71 0.087 3286  4010     1.3 7.0e+03
4  0.58 0.74 0.015 206 3.4e+03  0.0e+00    11.5 0.77 0.075 1787  2441     1.4 3.6e+03
5  0.54 0.74 0.015 185 1.8e+03 4.3e-264     9.4 0.81 0.056  341   928     1.5 1.4e+03
6  0.54 0.71 0.016 165 1.0e+03 1.8e-125     8.3 0.84 0.043 -277   247     1.8 6.5e+02
7  0.51 0.70 0.019 146 7.1e+02  1.2e-74     7.9 0.85 0.037 -451    13     1.9 4.3e+02
8  0.51 0.66 0.022 128 5.0e+02  7.1e-46     7.4 0.85 0.032 -513  -106     2.0 2.8e+02
9  0.49 0.68 0.027 111 3.8e+02  8.2e-31     7.1 0.86 0.029 -503  -150     2.0 2.0e+02
10 0.50 0.64 0.032  95 2.9e+02  4.5e-21     6.7 0.87 0.027 -468  -166     2.1 1.4e+02
11 0.48 0.65 0.039  80 1.8e+02  2.2e-09     6.4 0.87 0.021 -457  -203     2.1 9.0e+01
12 0.49 0.64 0.047  66 1.1e+02  6.9e-04     6.3 0.88 0.015 -415  -205     2.2 5.5e+01
13 0.47 0.65 0.057  53 7.6e+01  2.0e-02     6.2 0.88 0.012 -345  -176     2.3 3.7e+01
14 0.46 0.64 0.066  41 5.4e+01  7.9e-02     5.6 0.89 0.011 -271  -141     2.1 2.2e+01
```

How many factors? What ever you want



fa (from the help page)

Exploratory Factor analysis using MinRes (minimum residual) as well as EFA by Principal Axis, Weighted Least Squares or Maximum Likelihood

Among the many ways to do latent variable exploratory factor analysis (EFA), one of the better is to use Ordinary Least Squares (OLS) to find the minimum residual (minres) solution. This produces solutions very similar to maximum likelihood even for badly behaved matrices. A variation on minres is to do weighted least squares (WLS). Perhaps the most conventional technique is principal axes (PAF). An eigen value decomposition of a correlation matrix is done and then the communalities for each variable are estimated by the first n factors. These communalities are entered onto the diagonal and the procedure is repeated until the sum(diag(r)) does not vary. Yet another estimate procedure is maximum likelihood. For well behaved matrices, maximum likelihood factor analysis (either in the fa or in the factanal function) is probably preferred. Bootstrapped confidence intervals of the loadings and interfactor correlations are found by fa with n.iter > 1.

factor the bfi data set, extract 5 factors

R code

```
f5 <- fa(bfi[1:25],5)
```

```
Factor Analysis using method =  minres
Call: fa(r = bfi[1:25], nfactors = 5)
Standardized loadings (pattern matrix) based upon correlation matrix
     MR2   MR1   MR3   MR5   MR4   h2   u2 com
A1  0.21  0.17  0.07 -0.41 -0.06 0.19 0.81 2.0
A2 -0.02  0.00  0.08  0.64  0.03 0.45 0.55 1.0
A3 -0.03  0.12  0.02  0.66  0.03 0.52 0.48 1.1
A4 -0.06  0.06  0.19  0.43 -0.15 0.28 0.72 1.7
A5 -0.11  0.23  0.01  0.53  0.04 0.46 0.54 1.5
C1  0.07 -0.03  0.55 -0.02  0.15 0.33 0.67 1.2
C2  0.15 -0.09  0.67  0.08  0.04 0.45 0.55 1.2
C3  0.03 -0.06  0.57  0.09 -0.07 0.32 0.68 1.1
C4  0.17  0.00 -0.61  0.04 -0.05 0.45 0.55 1.2
C5  0.19 -0.14 -0.55  0.02  0.09 0.43 0.57 1.4
E1 -0.06 -0.56  0.11 -0.08 -0.10 0.35 0.65 1.2
E2  0.10 -0.68 -0.02 -0.05 -0.06 0.54 0.46 1.1
E3  0.08  0.42  0.00  0.25  0.28 0.44 0.56 2.6
E4  0.01  0.59  0.02  0.29 -0.08 0.53 0.47 1.5
E5  0.15  0.42  0.27  0.05  0.21 0.40 0.60 2.6
N1  0.81  0.10  0.00 -0.11 -0.05 0.65 0.35 1.1
N2  0.78  0.04  0.01 -0.09  0.01 0.60 0.40 1.0
N3  0.71 -0.10 -0.04  0.08  0.02 0.55 0.45 1.1
N4  0.47 -0.39 -0.14  0.09  0.08 0.49 0.51 2.3
N5  0.49 -0.20  0.00  0.21 -0.15 0.35 0.65 2.0
O1  0.02  0.10  0.07  0.02  0.51 0.31 0.69 1.1
O2  0.19  0.06 -0.08  0.16 -0.46 0.26 0.74 1.7
O3  0.03  0.15  0.02  0.08  0.61 0.46 0.54 1.2
O4  0.13 -0.32 -0.02  0.17  0.37 0.25 0.75 2.7
O5  0.13  0.10 -0.03  0.04 -0.54 0.30 0.70 1.2
```

```
                       MR2  MR1  MR3  MR5  MR4
SS loadings           2.57 2.20 2.03 1.99 1.59
Proportion Var        0.10 0.09 0.08 0.08 0.06
Cumulative Var        0.10 0.19 0.27 0.35 0.41
Proportion Explained  0.25 0.21 0.20 0.19 0.15
Cumulative Proportion 0.25 0.46 0.66 0.85 1.00

 With factor correlations of 
      MR2   MR1   MR3   MR5   MR4
MR2  1.00 -0.21 -0.19 -0.04 -0.01
MR1 -0.21  1.00  0.23  0.33  0.17
MR3 -0.19  0.23  1.00  0.20  0.19
MR5 -0.04  0.33  0.20  1.00  0.19
MR4 -0.01  0.17  0.19  0.19  1.00

Mean item complexity =  1.5
```

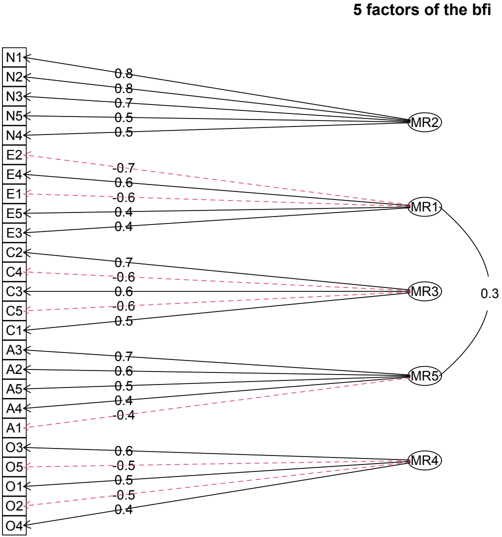
More important output

R code

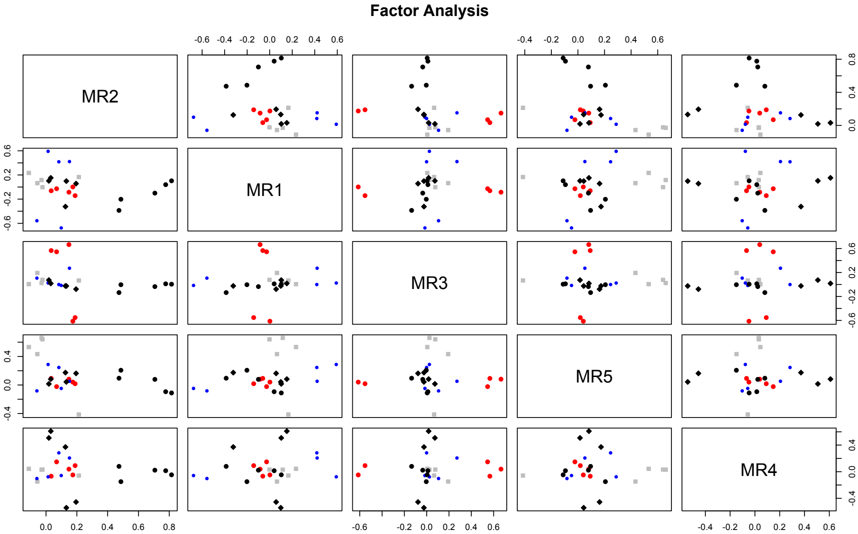
```
f5
diagram(f5, main="5 factors of the bfi")
plot(f5) #an alternative way to show the results
biplot(f5 )  #show a biplot
```

```
Mean item complexity =  1.5
Test of the hypothesis that 5 factors are sufficient.

df null model =  300  with the objective function =  7.23 with Chi Square =  20163.79
df of  the model are 185  and the objective function was  0.65

The root mean square of the residuals (RMSR) is  0.03
The df corrected root mean square of the residuals is  0.04

The harmonic n.obs is  2762 with the empirical chi square  1392.16  with prob <  5.6e-184
The total n.obs was  2800  with Likelihood Chi Square =  1808.94  with prob <  4.3e-264

Tucker Lewis Index of factoring reliability =  0.867
RMSEA index =  0.056  and the 90 % confidence intervals are  0.054 0.058
BIC =  340.53
Fit based upon off diagonal values = 0.98
Measures of factor score adequacy
                                                  MR2  MR1  MR3  MR5  MR4
Correlation of (regression) scores with factors   0.92 0.89 0.88 0.88 0.84
Multiple R square of scores with factors          0.85 0.79 0.77 0.77 0.71
Minimum correlation of possible factor scores     0.70 0.59 0.54 0.54 0.42
>
```

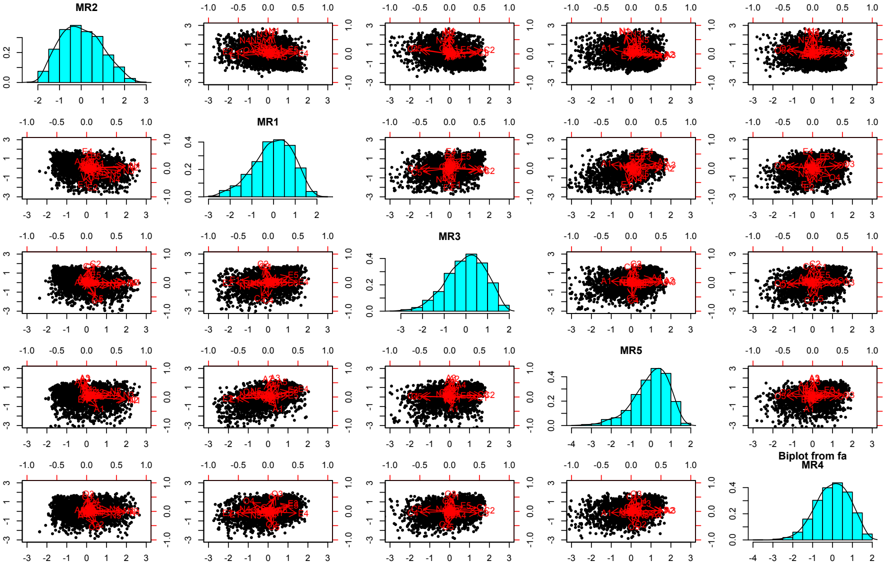
Use the `diagram` to show the structure



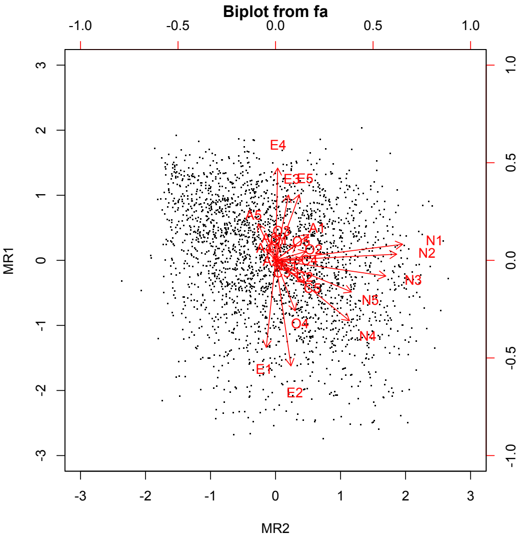
Use the `plot` to show the loadings in a different fashion



Use the `biplot` to show the loadings in a different fashion



Use the `biplot` with the choose option for just some factors



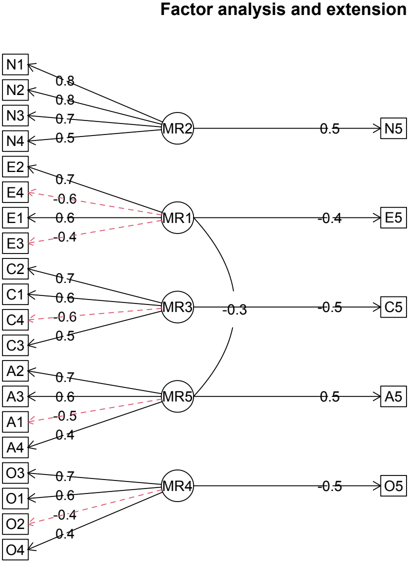
Factor Extension

1. What if you have more variables to fit into a factor model?
2. Dwyer (1937); Mosier (1938) and then elaborated by Gorsuch (1997); Horn (1973)
3. Specify the data set, the original variables, and the extension variables.

R code

```
f5.e <- fa.extend(bfi,nf=5,ov=c(1:4,6:9,11:14,16:19, 21:24),
       ev=c(20,15,10,5,25))  #notice the ordering for better graphic
diagram(f5.e,e.cut=.3)
```

Factor extension of the bfi



Cluster analysis as an alternative to factor analysis

The ICLUST algorithm was developed for the construction of internally consistent scales from items Revelle (1979)

1. Form the proximity (correlation) matrix
2. Find the most similar pair of items
3. Combine them into a new scale,
4. Recalculate the correlation matrix
5. Repeat steps 2-4 until various criteria are met

alpha	Coefficient α of the composite fails to increase (rarely happens)
beta	Coefficient *beta* (the worst split half reliability) fails to increase

`iclust` Is meant for forming scales from items and is a useful guide to the structure of a test.

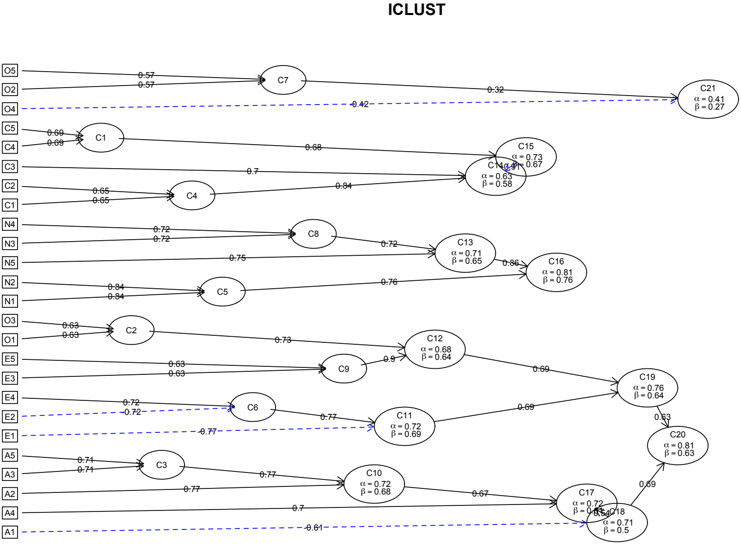
`iclust` Revelle (1979)

R code

```
ic <- iclust(bfi[1:25])
 summary(ic)
```

```
ICLUST (Item Cluster Analysis)Call: iclust(r.mat = bfi[1:25])
ICLUST

Purified Alpha:
 C20  C16  C15  C21
0.80 0.81 0.73 0.61

 Guttman Lambda6*
 C20  C16  C15  C21
0.82 0.81 0.72 0.61

Original Beta:
 C20  C16  C15  C21
0.63 0.76 0.67 0.27

Cluster size:
C20 C16 C15 C21
 10   5   5   5

Purified scale intercorrelations
 reliabilities on diagonal
 correlations corrected for attenuation above diagonal:
      C20    C16   C15   C21
C20  0.80 -0.291 -0.40 -0.33
C16 -0.24  0.815  0.29  0.11
C15 -0.30  0.221  0.73  0.30
C21 -0.23  0.074  0.20  0.61
```

Hierarchical cluster analysis of items using `iclust`



Multiple levels of factors

1. Hierarchical/higher order models (Jensen & Weng, 1994)
    - Simulated data to match Jensen & Weng (1994)
    - Ability data from the ICAR Condon & Revelle (2014)
    - Size data from the United States Airforce
    - Hierarchical solution of the SAPA Personality Inventory (spi) (Condon, 2018)
2. Bifactor models (Holzinger & Swineford, 1937; Reise, 2012; Rodriguez, Reise & Haviland, 2016)
    - Schmid & Leiman (1957) introduced a transformation of a higher order model into a bifactor model.
    - Constraints on the factor loadings given the structure.
3. Comparing factors at different levels of n factors using the `bassAckward` algorithm (Goldberg, 2006; Waller, 2007)
    - Two levels of factors from the SAPA Personality Inventory (spi) (Condon, 2018)
4. The SAPA Personality Inventory (spi) (Condon, 2018) data set has 135 items plus 10 criteria variables for 4,000 participants.
    - It was developed from 696 IPIP items to represent 200 broad and narrow public domain measures of personality.

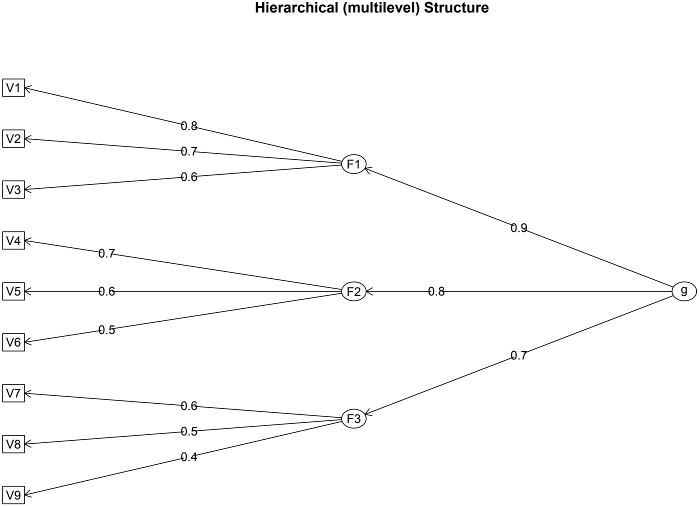
Simulating 9 variables from Jensen & Weng (1994)

R code

```
jensen <- sim.hierarchical()  #the default values are Jensen-Weng
f3 <- fa(jensen,3)
om<- omega(jensen)
diagram(om, sl=FALSE); diagram(om)  #default is to do Schmid-Leiman
```

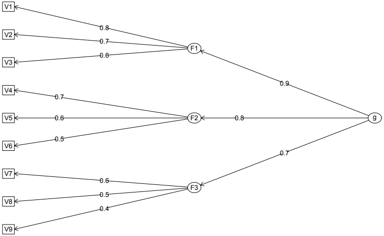
```
Factor Analysis using method =  minres
Call: fa(r = jensen, nfactors = 3)
Standardized loadings (pattern matrix) based upon correlation matrix
   MR1 MR3 MR2   h2   u2 com
V1 0.8 0.0 0.0 0.64 0.36   1
V2 0.7 0.0 0.0 0.49 0.51   1
V3 0.6 0.0 0.0 0.36 0.64   1
V4 0.0 0.7 0.0 0.49 0.51   1
V5 0.0 0.6 0.0 0.36 0.64   1
V6 0.0 0.5 0.0 0.25 0.75   1
V7 0.0 0.0 0.6 0.36 0.64   1
V8 0.0 0.0 0.5 0.25 0.75   1
V9 0.0 0.0 0.4 0.16 0.84   1

                       MR1  MR3  MR2
SS loadings           1.49 1.10 0.77
Proportion Var        0.17 0.12 0.09
Cumulative Var        0.17 0.29 0.37
Proportion Explained  0.44 0.33 0.23
Cumulative Proportion 0.44 0.77 1.00

 With factor correlations of 
     MR1  MR3  MR2
MR1 1.00 0.72 0.63
MR3 0.72 1.00 0.56
MR2 0.63 0.56 1.00
```

A higher order factor representation

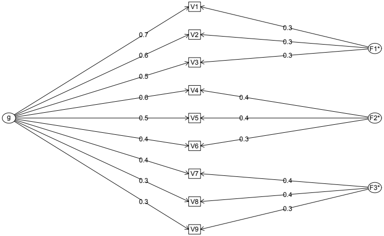


Schmid & Leiman (1957) transformation to a bifactor model

Hierarchical (multilevel) Structure



Omega with Schmid Leiman Transformation



Unfortunately, the `bifactor` rotation does not capture the right structure

R code

```
f4 <- fa(jensen,4, rotate="bifactor")
```

```
Factor Analysis using method =  minres
Call: fa(r = jensen, nfactors = 4, rotate = "bifactor")
Standardized loadings (pattern matrix) based upon correlation matrix
    MR1   MR3   MR2   MR4   h2   u2 com
V1 0.79 -0.03 -0.09  0.02 0.63 0.37 1.0
V2 0.70 -0.05 -0.09 -0.06 0.51 0.49 1.1
V3 0.60 -0.03 -0.07  0.12 0.38 0.62 1.1
V4 0.53  0.45  0.01  0.00 0.49 0.51 2.0
V5 0.46  0.39  0.01  0.00 0.36 0.64 2.0
V6 0.38  0.32  0.00  0.00 0.25 0.75 2.0
V7 0.43  0.01  0.42  0.00 0.36 0.64 2.0
V8 0.36  0.01  0.35  0.00 0.25 0.75 2.0
V9 0.29  0.01  0.28  0.00 0.16 0.84 2.0

                       MR1  MR3  MR2  MR4
SS loadings           2.50 0.47 0.39 0.02
Proportion Var        0.28 0.05 0.04 0.00
Cumulative Var        0.28 0.33 0.37 0.38
Proportion Explained  0.74 0.14 0.12 0.01
Cumulative Proportion 0.74 0.88 0.99 1.00
```

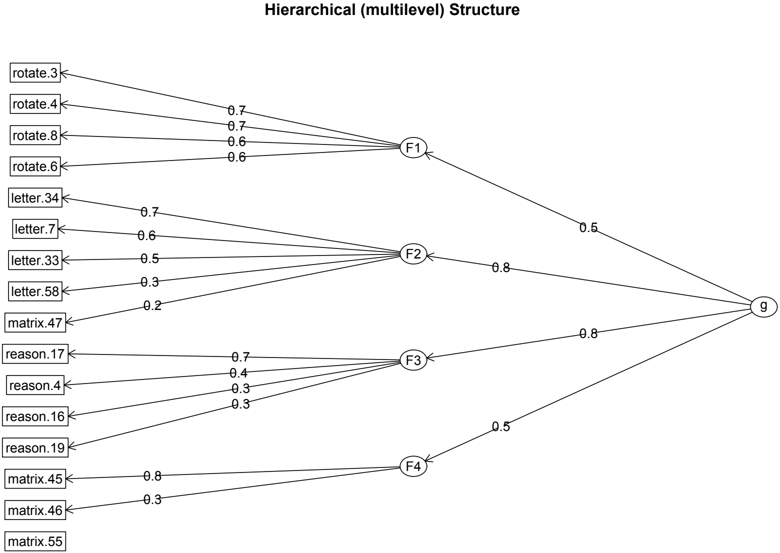
Another case: the ICAR 16

R code

```
om.icar <- omega(icar,4)
```

```
Schmid Leiman Factor loadings greater than  0.2
             g   F1*   F2*   F3*   F4*   h2   u2   p2
reason.4  0.50              0.28       0.35 0.65 0.74
reason.16 0.42              0.21       0.23 0.77 0.76
reason.17 0.55              0.46       0.51 0.49 0.59
reason.19 0.44              0.21       0.25 0.75 0.78
letter.7  0.51        0.35             0.39 0.61 0.68
letter.33 0.46        0.31             0.31 0.69 0.69
letter.34 0.53        0.39             0.43 0.57 0.65
letter.58 0.47        0.20             0.28 0.72 0.78
matrix.45 0.40                    0.64 0.57 0.43 0.28
matrix.46 0.40                    0.26 0.24 0.76 0.65
matrix.47 0.43                         0.23 0.77 0.79
matrix.55 0.29                         0.13 0.87 0.66
rotate.3  0.36  0.60                   0.50 0.50 0.26
rotate.4  0.41  0.60                   0.53 0.47 0.32
rotate.6  0.40  0.49                   0.41 0.59 0.39
rotate.8  0.33  0.54                   0.41 0.59 0.27

With Sums of squares  of:
   g  F1*  F2*  F3*  F4*
3.05 1.31 0.47 0.40 0.53
```

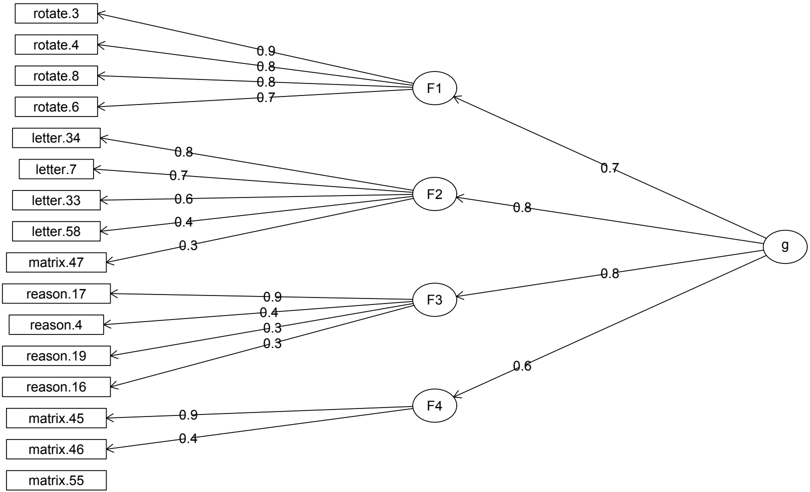
`omega` of the `ability` items



Pearson, polychoric and tetrachoric correlations

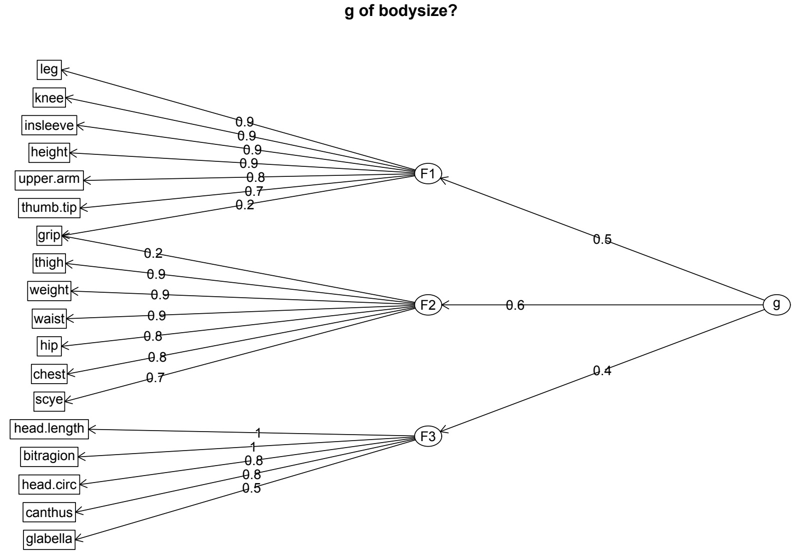
1. As we know, Pearson correlations are appropriate for continuous data.
2. But with categorical or dichotomous data, the correlations are attenuated.
3. The tetrachoric correlation estimates what the Pearson would be with continuous data.
    - Tetrachoric and polychoric correlations are thus estimates of the *latent* correlation assuming bivariate normality.
    - Appropriate for determining the structure of correlations, inappropriate for estimates of reliability.
4. The `fa`, `omega` functions have an option to find the tetrachoric/polychoric correlations before factoring.
5. Particularly appropriate for dichotomous variables (e.g. the ICAR example, `ability`)

`omega` of the `ability` items using tetrachoric correlations

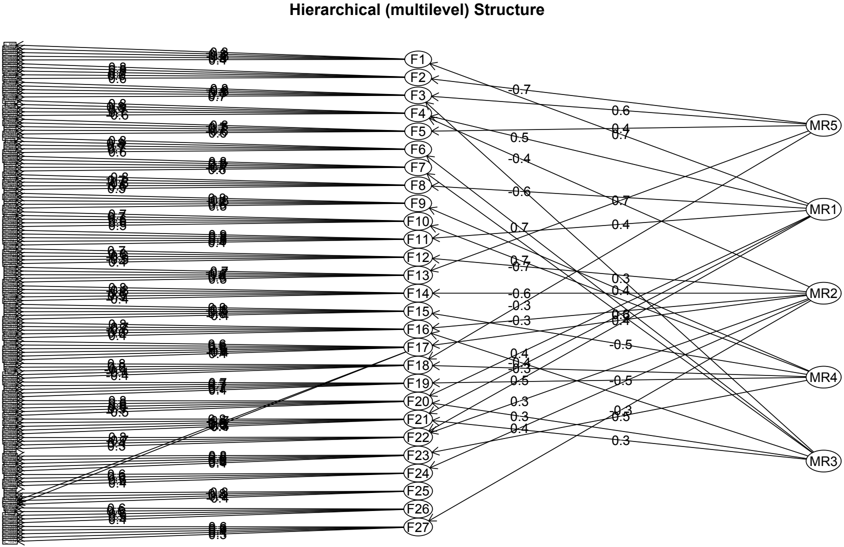


Hierarchical/multilevel Structure using tetrachoric correlations

Is there a general factor of body size? The USAF data set



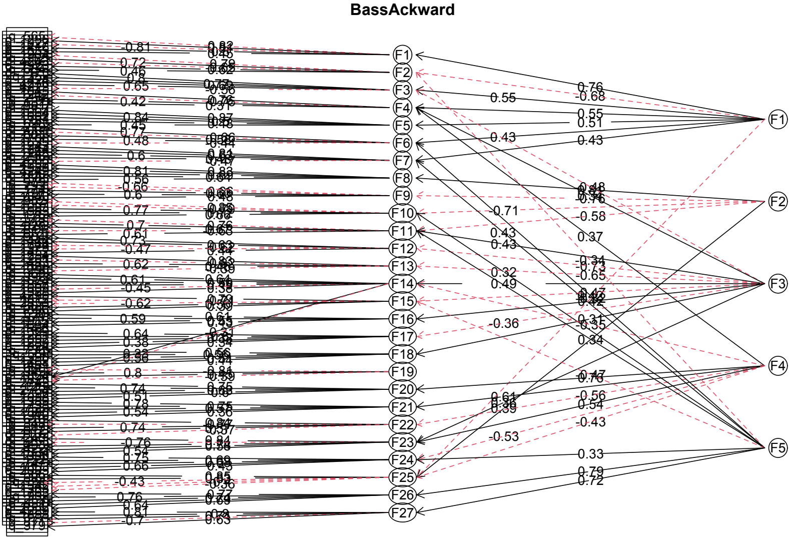
`fa.hierarchical` solution of the `spi` of the spi items



The “Bass-Ackwards” algorithm

1. Goldberg (2006) described a hierarchical factor structure organization from the “top down”.
   - The original idea was to do successive factor analyses from 1 to nf factors organized by factor score correlations from one level to the next.
2. Waller (2007) discussed a simple way of doing this for components without finding the scores.
3. Using the factor correlations (from Gorsuch, 1983) to organize factors hierarchically results may be organized at many different levels.
4. The algorithm may be applied to principal components (pca) or to true factor analysis.
5. Implemented as `bassAckward`.
6. The solutions should not be confused with a hierarchical solution where the higher order factors are factors of the lower order factors.

`bassAckward` solution of the spi items for 5 and 27 factors



Scale Construction

1. *psych* was specifically designed for the problem of reading and describing sets of items and then forming unit weighed scales from these items.
2. The advantage of scales formed from unit weighted items rather than factor weights is that they are more robust to sample variation (Widaman & Revelle, 2022).
3. Although there are functions to combine a set of items into just one scale `alpha` the more typical problem is form multiple scales e.g., `scoreItems`.
4. `fastScore` will scale scores without any accompanying statistics, but the more typical case is to use . pfunscoreItems.
5. To find scales based upon Item Response Theory, use `scoreIrt`

Example data sets

1. The `sai` represents 3,032 participants on 20 state anxiety items with 1229 participants who took it twice, 1047 with three measures, and 70 with four measures.
2. The `epi` represents 3570 participants for the 57 items of the Eysenck Personality Inventory from the early 1990s at the PMC lab.
3. An additional data set (`epiR`) has test and retest information for 474 participants.
4. The Motivational State Questionnaire `msqR` (Revelle & Anderson, 1998) contains 75 mood items for 3032 unique participants . 2753 took it at least twice, 446 three times, and 181 four times.

Scoring scales

1. For one set of items for one scale use `alpha`
   - Will warn if item x total correlations are negative and encourage
   - Using the check.keys option to reverse score negatively keyed items
2. More typical is to specify a `keys.list` of multiple keys each with multiple items.
   - Negatively keyed items are reversed scored by subtracting from the maximum possible item score - minimum possible item score
   - Scale scores are expressed as the *mean* item response, although *sum* scores is also an option,
   - Missing items scores can be *imputed* by means, medians, or ignored.
3. Most scoring functions return scores as well as statistics for the scales.
4. `scoreFast` and `scoreVeryFast` just return the scores.

Example keys list

R code

```
sai.keys <- list(sai = c("tense","regretful" , "upset", "worrying", "anxious", "nervous" ,
"jittery" , "high.strung", "worried" , "rattled","-calm",
"-secure","-at.ease","-rested","-comfortable", "-confident" ,"-relaxed" , "-content" ,
"-joyful", "-pleasant"  ) ,
sai.p = c("calm","at.ease","rested","comfortable", "confident", "secure" ,"relaxed" ,
       "content" , "joyful", "pleasant" ),
sai.n = c( "tense" , "anxious", "nervous" , "jittery" , "rattled",     "high.strung",
       "upset", "worrying","worried","regretful" )
       )
sai.keys
```

```
$sai
 [1] "tense"        "regretful"    "upset"        "worrying"     "anxious"      "nervous"
 [7] "jittery"      "high.strung"  "worried"      "rattled"      "-calm"        "-secure"
[13] "-at.ease"     "-rested"      "-comfortable" "-confident"   "-relaxed"     "-content"
[19] "-joyful"      "-pleasant"

$sai.p
 [1] "calm"        "at.ease"     "rested"      "comfortable" "confident"   "secure"
    "relaxed"     "content"     "joyful"      "pleasant"

$sai.n
 [1] "tense"       "anxious"     "nervous"     "jittery"     "rattled"     "high.strung"
    "upset"       "worrying"    "worried"     "regretful"
```

Some keys.list are part of the data set

R code

```
epi.keys
```

```
epi.keys
$E
 [1] "V1"   "V3"   "V8"   "V10"  "V13"  "V17"  "V22"  "V25"  "V27"  "V39"
[11] "V44"  "V46"  "V49"  "V53"  "V56"  "-V5"  "-V15" "-V20" "-V29" "-V32"
[21] "-V34" "-V37" "-V41" "-V51"

$N
 [1] "V2"  "V4"  "V7"  "V9"  "V11" "V14" "V16" "V19" "V21" "V23" "V26" "V28"
[13] "V31" "V33" "V35" "V38" "V40" "V43" "V45" "V47" "V50" "V52" "V55" "V57"

$L
[1] "V6"   "V24"  "V36"  "-V12" "-V18" "-V30" "-V42" "-V48" "-V54"

$Imp
[1] "V1"   "V3"   "V8"   "V10"  "V13"  "V22"  "V39"  "-V5"  "-V41"

$Soc
 [1] "V17"  "V25"  "V27"  "V44"  "V46"  "V53"  "-V11" "-V15" "-V20" "-V29"
[11] "-V32" "-V37" "-V51"
```

Dictionaries

1. Referring to item numbers is not convenient for discussing results.
2. Thus, it is possible to create a `dictionary` of the items.
3. A dictionary can be prepared outside of R by forming a spreadsheet including at least one column labeled “content” and with rownames for the item number or name. Other columns can specify the item source, or anything interesting.

```
headTail(epi.dictionary)
```

```
                                                    Content
V1                         Do you often long for excitement?
V2    Do you often need understanding friends to cheer you up?
V3                                 Are you usually carefree?
V4          Do you find it very hard to take no for an answer?
...                                                      <NA>
V54 Do you sometimes talk about things you know nothing about?
V55                             Do you worry about your health?
V56                         Do you like playing pranks on others?
V57                         Do you suffer from sleeplessness?
```

Using a keys list and a dictionary to show content

R code

```
lookupFromKeys(epi.keys, epi.dictionary, n=2)
```

```
$E
                             Content
V1 Do you often long for excitement?
V3          Are you usually carefree?

$N
                                                       Content
V2 Do you often need understanding friends to cheer you up?
V4        Do you find it very hard to take no for an answer?

$L

V6  If you say you will do something do you always keep your promise,
                 no matter how inconvenient it might be to do so?
V24       Are all your habits good and desirable ones?

$Imp
                             Content
V1 Do you often long for excitement?
V3          Are you usually carefree?

$Soc
                                                                             Content
V17      Do you like going out a lot?
V25 Can you usually let yourself go and enjoy yourself a lot at a lively party?
```

Or show the items for just one scale (as a way of checking the keys)

R code

```
lookupFromKeys(epi.keys, epi.dictionary)$Imp
```

```
lookupFromKeys(epi.keys, epi.dictionary)$Imp
                                                                   Content
V1                                          Do you often long for excitement?
V3                                                  Are you usually carefree?
V8    Do you generally do and say things quickly without stopping to think?
V10                                   Would you do almost anything for a dare?
V13                          Do you often do things on the spur of the moment?
V22                                When people shout at you do you shout back?
V39              Do you like doing things in which you have to act quickly?
V5-                Do you stop and think things over before doing anything?
V41-                             Are you slow and unhurried in the way you move?
```

Using `scoreItems` on the `epi` dataset

R code

```
scales <- scoreItems(epi.keys, epi) # produces raw scores and stats
overlap <- scoreOverlap(epi.keys, epi) #finds the correlarti
```

```
Scale intercorrelations corrected for attenuation
 raw correlations below the diagonal, (unstandardized) alpha on the diagonal
 corrected correlations above the diagonal:
        E      N     L    Imp   Soc
E    0.73 -0.228 -0.40  1.211  1.19
N   -0.17  0.793 -0.28  0.025 -0.33
L   -0.23 -0.165  0.44 -0.339 -0.31
Imp  0.71  0.015 -0.16  0.478  0.56     <- note that the imp and Soc scales
Soc  0.86 -0.250 -0.18  0.330  0.73    <- overlapping items with the E scale
> overlap <- scoreOverlap(epi.keys, epi)
>
> summary(overlap)
Call: scoreOverlap(keys = epi.keys, r = epi)

Scale intercorrelations adjusted for item overlap
Scale intercorrelations corrected for attenuation
 raw correlations (corrected for overlap) below the diagonal, (standardized) alpha on the dia
 corrected (for overlap and reliability) correlations above the diagonal:
        E     N     L   Imp   Soc
E    0.73 -0.23 -0.38 0.799  0.94
N   -0.18  0.80 -0.28 0.049 -0.31
L   -0.22 -0.17  0.45 -0.311 -0.30
Imp  0.47  0.03 -0.14 0.474  0.54
Soc  0.68 -0.24 -0.17 0.320  0.73
```

But what if we have overlapping scales?

1. Sometimes we are interested in how higher order scales relate to lower order scales.
2. The problem is, the items overlap.
3. Some people solve this problem by dropping the overlapping items. But this changes the meaning of the scales.
4. A fairly straight foward procedure is estimate the overlapping variances with the best estimate of shared (common) variance, similar to what is done when finding coefficient α.
5. Need to do this on the correlation matrix of the items, not the raw data.
6. See ?scoreOverlap

A small part of the output for scoreItems

R code

```
names(scales)
```

```
names(scales)
 [1] "scores"         "missing"        "alpha"          "av.r"
 [5] "sn"             "n.items"        "item.cor"       "cor"
 [9] "corrected"      "G6"             "item.corrected" "response.freq"
[13] "raw"            "ase"            "med.r"          "keys"
[17] "MIMS"           "MIMT"           "Call"


dim(scales$scores)
[1] 3570    5
```

By default, `scoreItems` imputes item medians for missing data

R code

```
describe(scales$scores)
scales <- scoreItems(epi.keys, epi,impute="none")
describe(scales$scores)
```

```
describe(scales$scores)
    vars    n mean   sd median trimmed  mad  min max range  skew kurtosis se
E      1 3570 1.46 0.17   1.46    1.46 0.19 1.04   2  0.96  0.26    -0.28  0
N      2 3570 1.54 0.19   1.54    1.55 0.19 1.00   2  1.00 -0.06    -0.42  0
L      3 3570 1.73 0.18   1.78    1.74 0.16 1.00   2  1.00 -0.57    -0.14  0
Imp    4 3570 1.51 0.20   1.56    1.51 0.16 1.00   2  1.00 -0.10    -0.57  0
Soc    5 3570 1.45 0.21   1.38    1.44 0.23 1.00   2  1.00  0.37    -0.46  0
> scales <- scoreItems(epi.keys, epi,impute="none")
> describe(scales$scores)
    vars    n mean   sd median trimmed  mad min max range  skew kurtosis se
E      1 3516 1.46 0.17   1.46    1.46 0.19   1   2     1  0.20    -0.34  0
N      2 3514 1.54 0.19   1.54    1.55 0.19   1   2     1 -0.06    -0.47  0
L      3 3510 1.73 0.18   1.78    1.74 0.16   1   2     1 -0.56    -0.10  0
Imp    4 3516 1.52 0.20   1.56    1.52 0.16   1   2     1 -0.16    -0.51  0
Soc    5 3509 1.45 0.22   1.46    1.44 0.23   1   2     1  0.34    -0.52  0
```

The structure of scales can be found from correlation matrices

1. The typical use of scoring scales is from raw data.
2. But for those of us interested in large data matrices with lots of missing data, it is convenient to score from the correlation matrix level.
3. If we just care about the correlations of composite scales, the correlations are adequate.
4. Given a N x n data matrix of deviation scores, for N subjects on n items, ${}_N\boldsymbol{X}_n$, the covariance matrix of the n items, ${}_n\boldsymbol{C}_n$, is just
$${}_n\boldsymbol{C}_n = \boldsymbol{XX'} * (N-1)^{-1},$$
with variances, σ_i^2, on the diagonal of $\boldsymbol{C}$ and item by item covariances, σ_{ij}, off the diagonal.
5. The covariances of scales are just
$${}_k\boldsymbol{Cs}_k = {}_k\boldsymbol{K'}_n \, {}_n\boldsymbol{C}_n \, {}_n\boldsymbol{K}_k. \tag{3}$$

Scales from correlations

R code

```
R <- lowerCor(epi, show=FALSE)
fromCors <- scoreItems(epi.keys, R)
summary(fromCors)
```

```
Call: scoreOverlap(keys = epi.keys, r = R)

Scale intercorrelations adjusted for item overlap
Scale intercorrelations corrected for attenuation
 raw correlations (corrected for overlap) below the diagonal, (standardized) alpha on the dia
 corrected (for overlap and reliability) correlations above the diagonal:
        E     N     L    Imp   Soc
E    0.73 -0.23 -0.38  0.799  0.94
N   -0.18  0.80 -0.28  0.049 -0.31
L   -0.22 -0.17  0.45 -0.311 -0.30
Imp  0.47  0.03 -0.14  0.474  0.54
Soc  0.68 -0.24 -0.17  0.320  0.73
```

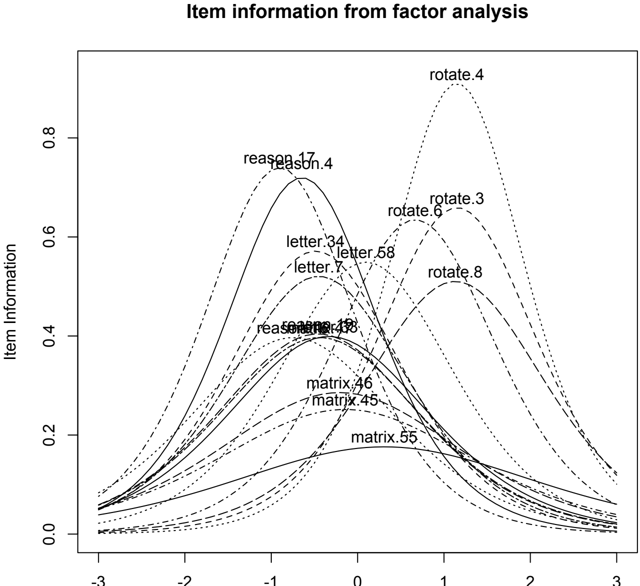
Item Response Theory from factor analysis

R code

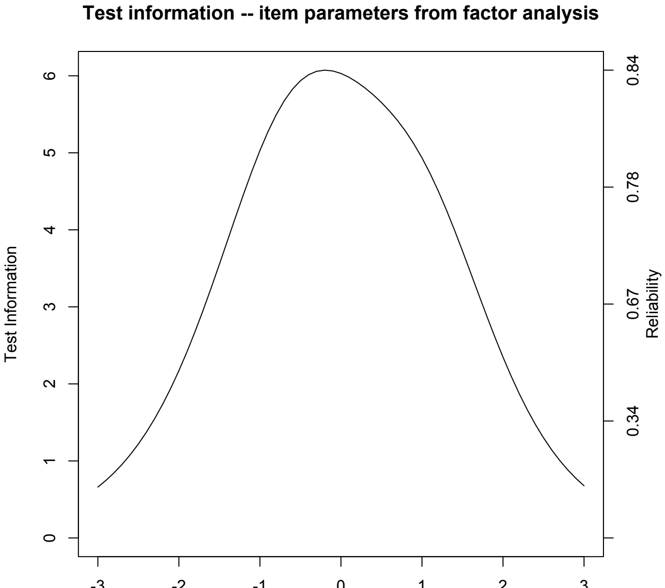
```
ab.irt <- irt.fa(ability)
plot(ab.irt)
plot(ab.irt,type="test")
```

```
Item Response Theory using factor analysis with Call: irt.fa(x = ability)

Test of the hypothesis that 1 factor is sufficient.
The degrees of freedom for the model is 104  and the objective function was  1.92
The number of observations was  1525  with Chi Square =  2906.16  with prob <  0

The root mean square of the residuals (RMSA) is  0.08
The df corrected root mean square of the residuals is  0.09

Tucker Lewis Index of factoring reliability =  0.722
RMSEA index =  0.133  and the 10 % confidence intervals are  0.129 0.137
BIC =  2143.86
```

Item Information for one factor of ability items



Test Information for one factor of ability items



Multiple types of reliability

1. Internal consistency estimates
   - α, λ_6 , use the `alpha` or `scoreItems` functions
   - $\omega_{hierarchical}$ and ω_{total} use the `omega` function
2. IntraClass coefficients
   - ICC
3. Rater agreement use `kappa` function
4. Test Retest reliability

For the next examples we will use a built in data set

1. `bfi` consists of 25 personality items measuring 5 factors as well as some demographics.
2. The data were collected as part of the SAPA project and have 2,800 subjects.
3. For help on this data set, ?bfi
4. To see all of the *psych* data sets: `data(package="psych")`

First, we intentionally mis-specify the data

R code

```
alpha(bfi[1:5]) #score the first five items
```

```
Some items ( A1 ) were negatively correlated with the total scale a
probably should be reversed.
To do this, run the function again with the 'check.keys=TRUE' optic
Reliability analysis
Call: alpha(x = bfi[1:5])

  raw_alpha std.alpha G6(smc) average_r  S/N   ase mean   sd
      0.43      0.46    0.53      0.15 0.85 0.016  4.2 0.74

 lower alpha upper     95% confidence boundaries
0.4 0.43 0.46

 Reliability if an item is dropped:
   raw_alpha std.alpha G6(smc) average_r  S/N alpha se
A1      0.72      0.73    0.67     0.398 2.64    0.0087
A2      0.28      0.30    0.39     0.097 0.43    0.0219
A3      0.18      0.21    0.31     0.061 0.26    0.0249
A4      0.25      0.31    0.44     0.099 0.44    0.0229
A5      0.21      0.24    0.36     0.072 0.31    0.0238

 Item statistics
      n raw.r std.r r.cor r.drop mean  sd
```

Try it again. Turn on automatic reversals. Get the scores

R code

```
scores <- alpha(bfi[1:5], check.keys =TRUE)
```

```
 alpha(bfi[1:5], check.keys =TRUE)

Reliability analysis
Call: alpha(x = bfi[1:5], check.keys = TRUE)

  raw_alpha std.alpha G6(smc) average_r S/N   ase mean  sd
       0.7      0.71     0.68      0.33 2.5 0.009  4.7 0.9

 lower alpha upper     95% confidence boundaries
0.69 0.7 0.72

 Reliability if an item is dropped:
    raw_alpha std.alpha G6(smc) average_r S/N alpha se
A1-      0.72      0.73    0.67      0.40 2.6   0.0087
A2       0.62      0.63    0.58      0.29 1.7   0.0119
A3       0.60      0.61    0.56      0.28 1.6   0.0124
A4       0.69      0.69    0.65      0.36 2.3   0.0098
A5       0.64      0.66    0.61      0.32 1.9   0.0111
...

Warning message:
In alpha(bfi[1:5], check.keys = TRUE) :
  Some items were negatively correlated with total scale and were
```

R functions will return objects without necessarily telling you

1. The basic logic of R is that you can do lots of calculations, but you might not want all the output.
2. The output is there, to be processed by other functions if you want, but you probably don't want to see all of it unless you ask.,
3. Thus, `alpha` returns the scores based upon the scales you asked for, but doesn't show them, because they are so many,
4. The `str` command tells you the structure of an object. The `names` will just list the names of the objects.

names and str of alpha output

R code

```
names(scores)
str(scores)
```

```
   names(scores)
    [1] "total"            "alpha.drop"       "item.stats"       "response.fre
     "scores"            "nvar"             "boot.ci"
    [9] "boot"             "Unidim"           "Fit"              "call"

        $ total          :'data.frame':         1 obs. of  8 variables:
     ..$ raw_alpha: num 0.703
     ..$ std.alpha: num 0.713
     ..$ G6(smc)  : num 0.683
     ..$ average_r: num 0.332
     ..$ S/N      : num 2.48
     ..$ ase      : num 0.00895
     ..$ mean     : num 4.65
     ..$ sd       : num 0.898
    $ alpha.drop    :'data.frame':         5 obs. of  6 variables:
     ..$ raw_alpha: num [1:5] 0.719 0.617 0.6 0.686 0.643
     ..$ std.alpha: num [1:5] 0.726 0.626 0.613 0.694 0.656
     ..$ G6(smc)  : num [1:5] 0.673 0.579 0.558 0.65 0.605
     ..$ average_r: num [1:5] 0.398 0.295 0.284 0.361 0.322
     ..$ S/N      : num [1:5] 2.64 1.67 1.58 2.26 1.9
     ..$ alpha se : num [1:5] 0.00873 0.0119 0.01244 0.00983 0.01115
    $ item.stats    :'data.frame':         5 obs. of  7 variables:
```

One of the objects of `alpha` is the scores object

R code

```
describe(scores$scores)
```

But, since there are scores for all subjects, but just one score, this is not very interesting.

```
 describe(scores$scores)
   vars    n mean  sd median trimmed  mad min max range  skew kurto
X1    1 2800 4.65 0.9    4.8    4.73 0.89   1   6     5 -0.76
>
```

Note that alpha has the option of doing cumulative scores (adding up items, or scoring in the unit of the items (the default).

R code

```
scores <- alpha(bfi[1:5],check.keys=TRUE,cumulative=TRUE)
#set the cumulative option to be true
describe(scores$scores)
```

```
describe(scores$scores)
   vars    n  mean   sd median trimmed  mad min max range  skew kur
X1    1 2800 23.08 4.54     24   23.43 4.45   5  30    25 -0.73
```

Better yet, score multiple scales at one time

R code

```
sc.bfi <- scoreItems(bfi.keys,bfi, impute="none")
sc.bfi  #lots of ouput
scores.bfi <- sc.bfi$scores
describe(scores.bfi)
```

```
describe(scores.bfi)
              vars    n mean   sd median trimmed  mad min max range  skew kurtosis   se
agree            1 2800 4.65 0.90    4.8    4.73 0.89 1.0   6   5.0 -0.76     0.40 0.02
conscientious    2 2800 4.27 0.95    4.4    4.30 0.89 1.0   6   5.0 -0.40    -0.19 0.02
extraversion     3 2800 4.15 1.06    4.2    4.20 1.19 1.0   6   5.0 -0.48    -0.21 0.02
neuroticism      4 2800 3.16 1.20    3.0    3.13 1.48 1.0   6   5.0  0.21    -0.67 0.02
openness         5 2800 4.59 0.81    4.6    4.62 0.89 1.2   6   4.8 -0.34    -0.29 0.02
```

R code

```
sc.bfi  #lots of ouput
```

```
sc.bfi  #lots of ouput
Call: scoreItems(keys = bfi.keys, items = bfi, impute = "none")

(Standardized) Alpha:
      agree conscientious extraversion neuroticism openness
alpha   0.7          0.73         0.76        0.81      0.6

Standard errors of unstandardized Alpha:
      agree conscientious extraversion neuroticism openness
ASE   0.014         0.014        0.013       0.011    0.017

Standardized Alpha of observed scales:
     agree conscientious extraversion neuroticism openness
[1,]   0.7          0.73         0.76        0.81      0.6

Average item correlation:
          agree conscientious extraversion neuroticism openness
average.r  0.32          0.35         0.39        0.47     0.23

Median item correlation:
        agree conscientious  extraversion   neuroticism      openness
         0.34          0.34          0.38          0.41          0.23

 Guttman 6* reliability:
         agree conscientious extraversion neuroticism openness
Lambda.6   0.7          0.72         0.77        0.81     0.61

Signal/Noise based upon av.r :
             agree conscientious extraversion neuroticism openness
Signal/Noise   2.4           2.7          3.2         4.4      1.5

Scale intercorrelations corrected for attenuation
 raw correlations below the diagonal, alpha on the diagonal
```

Reliability is not just *alpha*. In fact it is more than internal consistency

See Revelle & Condon (2019) for a review.

1. Internal consistency
    - Some measures (e.g. α = KR20 = λ_3) were developed for desk calcuiators.
    - They just used item variances and total scale variances
    - Other internal consistency measures examined the factorial structure of the test (e.g. $\omega_h, \omega_t, \beta$)
2. Other measures include split-half reliabilities (there many split halves)
3. Perhaps better yet (but less common) are test-retest estimates.

α, $\omega_{hierarchical}$ and β as alternative measures of internal consistency

1. α as the mean split half reliability
    - `alpha` to find α
    - `splitHalf` to find all (if $n \leq 16$) or 10,000 random possible split half reliabilities ($n > 16$)
2. $\omega_{hierarchical}$ and ω_{total} as factor based reliabilities
    - $\omega_{hierarchical}$ estimates general factor saturation
    - Found using `omega` and `omegaSem`
3. β as worst split half reliability as an alternative estimate of the general factor saturation.
    - Found using a hierarchical clustering algorithm (`iclust`).
    - `iclust` is also useful for scale construction.

Using the omega function

R code

```
omega(ability,4)
```

```
Omega
Call: omega(m = ability, nfactors = 4)
Alpha:                 0.83
G.6:                   0.84
Omega Hierarchical:    0.65
Omega H asymptotic:    0.76
Omega Total            0.86

Schmid Leiman Factor loadings greater than  0.2
             g   F1*   F2*   F3*   F4*   h2   u2   p2
reason.4  0.50              0.27       0.34 0.66 0.73
reason.16 0.42              0.21       0.23 0.77 0.76
reason.17 0.55              0.47       0.52 0.48 0.57
reason.19 0.44              0.21       0.25 0.75 0.77
letter.7  0.52        0.35             0.39 0.61 0.69
letter.33 0.46        0.30             0.31 0.69 0.70
letter.34 0.54        0.38             0.43 0.57 0.67
letter.58 0.47        0.20             0.28 0.72 0.78
matrix.45 0.40                    0.66 0.59 0.41 0.27
matrix.46 0.40                    0.26 0.24 0.76 0.65
matrix.47 0.42                         0.23 0.77 0.79
matrix.55 0.28                         0.12 0.88 0.65
rotate.3  0.36  0.61                   0.50 0.50 0.26
rotate.4  0.41  0.61                   0.54 0.46 0.31
rotate.6  0.40  0.49                   0.41 0.59 0.39
rotate.8  0.32  0.53                   0.40 0.60 0.26

With eigenvalues of:
   g  F1*  F2*  F3*  F4*
3.04 1.32 0.46 0.42 0.55
```

α from `alpha` and all split halves found using `splitHalf`

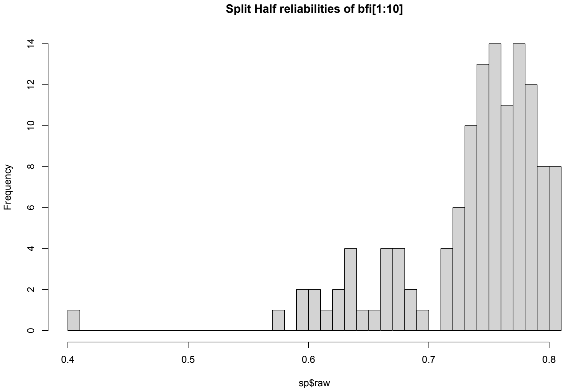
Find α and all split half reliabilities of 5 Agreeableness items and 5 Conscientiousness items from the `bfi` data set included in *psych*.

R code

```
alpha(bfi[1:10]) #find alpha, let it automatically reverse items
sp <- splitHalf(bfi[1:10],key=c(1,9,10), raw=TRUE) #reverse 3 items
hists(sp$raw, breaks=51, main="Split Half reliabilities of bfi[1:10]"
```

```
Reliability analysis
Call: alpha(x = bfi[1:10])
  raw_alpha std.alpha G6(smc) average_r S/N  ase mean   sd
      0.73      0.74    0.76      0.22 2.8 0.01  4.5 0.73
 lower alpha upper     95% confidence boundaries
0.71 0.73 0.75

Split half reliabilities
Call: splitHalf(r = bfi[1:10], key = c(1, 9, 10))

Maximum split half reliability (lambda 4) =  0.81
Guttman lambda 6                          =  0.76
Average split half reliability            =  0.73
Guttman lambda 3 (alpha)                  =  0.74
Minimum split half reliability  (beta)    =  0.41
```

All possible spit halves of 5 agreeableness and 5 conscientiousness items. Note the one worst one! This is not one construct.



test-retest data sets

Discussed in part by Revelle & Condon (2019)

epiR 474 participants took the Eysenck Personality Inventory twice

msqR 3032 unique participants took the MSQ at least once, 2753 at least twice, 446 three times, and 181 four times.

1. The obvious approach: score time 1 and time 2 and correlate them.
2. A somewhat more elegant approach, use `testRetest`

The obvious approach

R code

```
epi1 <- selectBy(epiR, "time=1")
epi2 <- selectBy(epiR, "time=2")
scales1 <- scoreItems(epi.keys,epi1)
scales2 <- scoreItems(epi.keys,epi2)
summary(scales1)
summary(scales2)
```

```
summary(scales1)
Call: scoreItems(keys = epi.keys, items = epi1)
Scale intercorrelations corrected for attenuation
 raw correlations below the diagonal, (unstandardized) alpha on the diagonal
 corrected correlations above the diagonal:
        E       N     L   Imp   Soc
E    0.77 -0.2083 -0.36  1.219  1.14
N   -0.16  0.8135 -0.39 -0.015 -0.28
L   -0.19 -0.2185  0.39 -0.432 -0.15
Imp  0.77 -0.0099 -0.19  0.519  0.66
Soc  0.87 -0.2216 -0.08  0.417  0.76
> summary(scales2)
Call: scoreItems(keys = epi.keys, items = epi2)
Scale intercorrelations corrected for attenuation
 raw correlations below the diagonal, (unstandardized) alpha on the diagonal
 corrected correlations above the diagonal:
        E      N     L   Imp   Soc
E    0.74 -0.271 -0.46  1.209  1.18
N   -0.21  0.796 -0.30 -0.074 -0.32
L   -0.25 -0.167  0.40 -0.593 -0.26
Imp  0.73 -0.046 -0.26  0.488  0.62
Soc  0.88 -0.251 -0.14  0.379  0.75
```

Correlate time 1 and time 2 scale scores

R code

```
cor2(epi1.scales$scores, epi2.scales$scores)
```

```
cor2(epi1.scales$scores, epi2.scales$scores)
        E     N     L   Imp   Soc
E    0.81 -0.16 -0.21  0.58  0.73
N   -0.14  0.80 -0.18  0.00 -0.18
L   -0.22 -0.16  0.65 -0.22 -0.11
Imp  0.60 -0.02 -0.21  0.70  0.38
Soc  0.73 -0.19 -0.11  0.35  0.81
```

Note that the test retest of extraversion (.81) and impulsivity (.70) are much higher than α at time 1 or time 2.

test-retest

R code

```
epi.test <- testRetest(epiR,keys=epi.keys$E)
```

```
Call: testRetest(t1 = epiR, keys = epi.keys$E)

Number of subjects =  474  Number of items =  24
 Correlation of scale scores over time 0.82
 Alpha reliability statistics for time 1 and time 2
       raw G3 std G3   G6 av.r  S/N   se lower upper var.r
Time 1   0.77   0.77 0.80 0.12 3.34 0.11  0.35  0.98  0.01
Time 2   0.75   0.74 0.79 0.11 2.90 0.13  0.30  0.98  0.01

 Mean between person, across item reliability =  0.54
 Mean within person, across item reliability =  0.6
with standard deviation of  0.2

 Mean within person, across item d2 =  0.18
R1F  =  0.87 Reliability of average of all items for one  time (Random time effects)
RkF  =  0.93 Reliability of average of all items and both times (Fixed time effects)
R1R  =  0.83 Generalizability of a single time point across all items (Random time effects)
Rc   =  0.29 Generalizability of change (fixed time points, fixed items)
Multilevel components of variance
             variance Percent
ID               0.02    0.09
Time             0.00    0.00
Items            0.05    0.20
ID x time        0.00    0.01
ID x items       0.09    0.35
time x items     0.00    0.00
Residual         0.09    0.36
Total            0.25    1.00
```

Do this again, for the Impulsivity scale

R code

```
imp.test <- testRetest(epiR,keys=epi.keys$Imp)
```

```
Test Retest reliability
Call: testRetest(t1 = epiR, keys = epi.keys$Imp)

Number of subjects =  474  Number of items =  9
 Correlation of scale scores over time 0.7
 Alpha reliability statistics for time 1 and time 2
       raw G3 std G3   G6 av.r  S/N   se lower upper var.r
Time 1   0.51   0.52 0.52 0.11 1.08 0.52  0.25  0.99  0.01
Time 2   0.51   0.52 0.52 0.11 1.07 0.51  0.25  0.99  0.01

 Mean between person, across item reliability =  0.52
 Mean within person, across item reliability =  0.58
with standard deviation of  0.3

 Mean within person, across item d2 =  0.2
R1F  =  0.73 Reliability of average of all items for one  time (Random time effects)
RkF  =  0.85 Reliability of average of all items and both times (Fixed time effects)
R1R  =  0.71 Generalizability of a single time point across all items (Random time effects)
Rc   =  0.12 Generalizability of change (fixed time points, fixed items)
Multilevel components of variance
             variance Percent
ID                 0.02    0.08
Time               0.00    0.00
Items              0.04    0.18
ID x time          0.00    0.01
ID x items         0.09    0.35
time x items       0.00    0.00
Residual           0.10    0.39
Total              0.25    1.00
```

Item stats for the Impulsivity scale

R code

```
> imp.test$item.stats
```

```
imp.test$item.stats
          rii         PC1        PC2    mean1    mean2 keys keys.orig
V1  0.4683082  0.21367412  0.1802059 1.297872 1.241379    1        V1
V3  0.5301015  0.50814071  0.4553856 1.397436 1.440171    1        V3
V8  0.4897216  0.57143300  0.6285440 1.674518 1.647436    1        V8
V10 0.5172569  0.50630391  0.4594627 1.854390 1.899573    1       V10
V13 0.5483092  0.69578254  0.6868002 1.411638 1.422747    1       V13
V22 0.5618986  0.26139796  0.2449332 1.428879 1.442765    1       V22
V39 0.5664763  0.58260278  0.5204840 1.447084 1.432258    1       V39
V5  0.3345742  0.54082190  0.6009108 1.182403 1.181237    1       -V5
V41 0.6196989 -0.05112183 -0.1655940 1.701944 1.762313   -1      -V41
```

The mean test retest correlation of the items (.52) is much higher than the mean communality of those items (.14).

Regression from correlation matrices

1. Although typically done from the raw data, multiple regression can also be done from the correlation matrix.
2. `lmCor` will take either raw data or correlation matrices as input and then do and show the multiple regressions.
3. Consider how the big 5 from the `spi` data set can predict 10 criteria.

Predicting 10 criteria from the Big 5 using the `spi` dataset

R code

```
spi.scales <- scoreItems(spi.keys[1:5],spi) #score the scales
spi.scores.crit <- cbind(spi.scales$scores, spi[1:10])
spi.reg <- lmCor(x=1:5, y=6:15,  data =spi.scores.crit, plot=FALSE)
spi.reg.r <- rbind(spi.reg$coefficients, R = spi.reg$R)
ord <- order(spi.reg$R)   #sort the resulting regressions
df2latex(spi.reg.r[ord],big=.2) #convert to a LaTex table
```

Table: 10 criteria and 5 predictors from the `spi`

A table from the psych package in R

Variable	p1edu	p2edu	ER	wllns	smoke	edctn	exer	age	sex	helth
(Intercept)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Agree	0.02	0.01	-0.03	0.06	-0.08	0.12	-0.01	0.16	0.16	0.01
Consc	-0.03	-0.05	0.02	0.11	-0.08	0.06	0.16	0.13	0.10	0.17
Neuro	-0.04	-0.03	0.13	0.03	0.06	-0.15	-0.12	-0.14	**0.29**	**-0.27**
Extra	0.05	0.06	0.05	0.09	0.08	-0.09	0.09	-0.11	0.09	0.14
Open	0.06	0.06	-0.01	0.00	0.09	0.14	0.07	0.12	-0.12	0.01
R	0.10	0.11	0.13	0.17	0.18	**0.26**	**0.27**	**0.31**	**0.36**	**0.41**

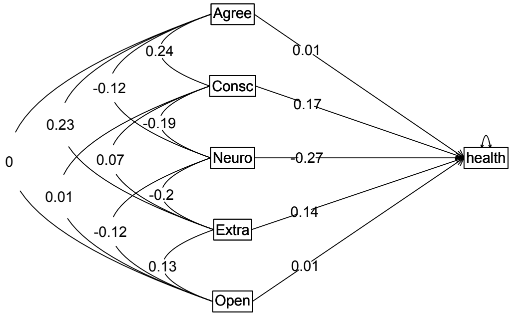
Show just one prediction model

R code

```
lmCor(health ~ Agree + Consc + Neuro + Extra + Open,
        data=spi.scores.crit)
```

```
multiple Regression from raw data

 DV =  health
            slope   se      t        p lower.ci upper.ci  VIF Vy.x
(Intercept)  0.00 0.01   0.00 1.0e+00    -0.03     0.03 1.00 0.00
Agree        0.01 0.02   0.41 6.8e-01    -0.02     0.04 1.12 0.00
Consc        0.17 0.02  11.34 2.3e-29     0.14     0.20 1.09 0.04
Neuro       -0.27 0.02 -18.00 9.7e-70    -0.30    -0.24 1.09 0.09
Extra        0.14 0.02   9.45 5.9e-21     0.11     0.17 1.10 0.03
Open         0.01 0.01   0.86 3.9e-01    -0.02     0.04 1.03 0.00

Residual Standard Error =  0.91  with  3994  degrees of freedom

 Multiple Regression
          R   R2  Ruw R2uw Shrunken R2 SE of R2 overall F df1  df2         p
health 0.41 0.16 0.35 0.12        0.16     0.01    156.95   5 3994 1.18e-152
>
```

Regression Models



We have known since the 1930's the need to Cross validate

1. A lovely paper by Cureton (1950) makes this point in three pages.
2. `crossValidation` will apply a regression model from one sample to a second sample.

R code

```
N <- NROW(spi.scores.crit)
ss <- sample(N,N/2)
model <- lmCor(health ~ Agree + Consc + Neuro + Extra + Open,
        data=spi.scores.crit[ss,])
  cv <- crossValidation(model, spi.scores.crit[-ss, ] )
```

```
          Cross Validation
Call:crossValidation(model = model, data = spi.scores.crit[-ss, ])
Validities from raw items or from the correlation matrix
Number of unique predictors used =  5
       items  mat
health  0.42 0.42
Correlations based upon item based regressions
[1] 0.42
Correlations based upon correlation matrix based regressions
[1] 0.42
```

Cross validation of multiple models

R code

```
ss <- sample(NROW(spi),NROW(spi)/2)
spi.reg <- lmCor(x=1:5, y=6:15,  data =spi.scores.crit[ss,],
            plot=FALSE)
cv.reg <- crossValidation(spi.reg, data=spi.scores.crit[-ss,])
cv.reg
```

```
spi.reg <- lmCor(x=1:5, y=6:15,  data =spi.scores.crit[ss,], plot=FALSE)
> cv.reg <- crossValidation(spi.reg, data=spi.scores.crit[-ss,])
> cv.reg
Cross Validation
Call:crossValidation(model = spi.reg, data = spi.scores.crit[-ss,   ])

Validities from raw items or from the correlation matrix
Number of unique predictors used =  5
          items  mat
age        0.33 0.33
sex        0.38 0.38
health     0.39 0.40
p1edu      0.11 0.12
p2edu      0.12 0.12
education  0.26 0.26
wellness   0.15 0.15
exer       0.26 0.26
smoke      0.17 0.17
ER         0.13 0.13
```

Scale construction through “Machine Learning”

1. Supervised Learning was called item analysis in 1930
2. Need to cross validate is not a new concept.
    - Now called K-fold cross validation
        - With N = 2, this is the traditional cross validation of half the sample being derivation, half being validation.
        - Typical is 10 fold which means sample 90% validate on remaining 10% and repeat for successive slices.
    - Boot strap resampling
        - This samples N subjects (with replacement) from the N subjects.
        - This leads to 62.3 % of the subjects being ‘in the bag’ and 36.8% ‘out of bag”
        - fit the in bag subjects, validate on out of bag subjects.

k.fold folds=10

R code

```
bs <- bestScales(spi[11:145], criteria = spi[1:10],folds=10,
       dictionary=spi.dictionary[c(2,6)])
```

```
Call = bestScales(x = spi[11:145], criteria = spi[1:10], folds = 10, 
    dictionary = spi.dictionary[c(2, 6)])
          derivation.mean derivation.sd validation.m validation.sd final.valid final.wtd N.wt
age                  0.36        0.0055        0.352         0.042        0.35      0.36    1
sex                  0.35        0.0072        0.351         0.054        0.35      0.35    1
health               0.44        0.0054        0.435         0.046        0.43      0.44    1
p1edu                0.13        0.0164        0.128         0.042        0.12      0.19    1
p2edu                0.11        0.0117        0.087         0.039          NA      0.19    1
education            0.30        0.0100        0.290         0.071        0.30      0.31    1
wellness             0.24        0.0052        0.221         0.045        0.23      0.24    1
exer                 0.31        0.0088        0.289         0.038        0.30      0.32    1
smoke                0.27        0.0079        0.260         0.063        0.27      0.28    1
ER                   0.16        0.0163        0.139         0.060        0.16      0.16    1
```

two example results

R code

```
bs
```

```
Criterion =  age
       Freq mean.r sd.r                                    item                     L27
q_4296   10  -0.23 0.00                     Tell a lot of lies.                 Honesty
q_501    10  -0.21 0.01                     Cheat to get ahead.                 Honesty
q_4249   10  -0.21 0.01     Would call myself a nervous person.                 Anxiety
q_1024   10  -0.21 0.01              Hang around doing nothing.           EasyGoingness
q_1452   10  -0.20 0.01                      Neglect my duties.                Industry
q_803    10   0.20 0.01                  Express myself easily. EmotionalExpressiveness
q_1081   10  -0.20 0.00 Have difficulty expressing my feelings. EmotionalExpressiveness
q_808     9  -0.19 0.01                     Fear for the worst.                 Anxiety

Criterion =  health
       Freq mean.r sd.r                                           item         L27
q_820    10   0.35 0.00                  Feel comfortable with myself.   WellBeing
q_2765   10   0.34 0.01                         Am happy with my life.   WellBeing
q_811    10  -0.34 0.01 Feel a sense of worthlessness or hopelessness.   WellBeing
q_578    10  -0.34 0.01                                Dislike myself.   WellBeing
q_1371   10   0.31 0.01                                     Love life.   WellBeing
q_56     10   0.28 0.01                Am able to control my cravings. SelfControl
q_1505   10  -0.27 0.00                                  Panic easily.     Anxiety
q_4249   10  -0.26 0.00            Would call myself a nervous person.     Anxiety
q_808    10  -0.26 0.00                            Fear for the worst.     Anxiety
```

Try 100 bootstrap resamplings

R code

```
bs <- bestScales(spi[11:145], criteria = spi[1:10],n.iter=100,
          dictionary=spi.dictionary[c(2,6)])
```

```
Call = bestScales(x = spi[11:145], criteria = spi[1:10], n.iter = 100,
    dictionary = spi.dictionary[c(2, 6)])
          derivation.mean derivation.sd validation.m validation.sd final.valid final.wtd N.wt
age                  0.37         0.018        0.354         0.026        0.33      0.36 1
sex                  0.36         0.014        0.353         0.021        0.35      0.35 1
health               0.44         0.014        0.430         0.022        0.43      0.44 1
p1edu                0.16         0.032        0.116         0.027          NA      0.19 1
p2edu                0.15         0.032        0.085         0.028          NA      0.19 1
education            0.32         0.018        0.279         0.027        0.18      0.30 1
wellness             0.24         0.016        0.219         0.023        0.23      0.24 1
exer                 0.32         0.016        0.300         0.027        0.30      0.31 1
smoke                0.28         0.019        0.258         0.027        0.28      0.28 1
ER                   0.17         0.024        0.128         0.024          NA      0.16 1
```

With items and the number of bootstrap samples they were included

R code

```
bs
```

```
Criterion =  age
       Freq mean.r sd.r                                   item           L27
q_4296  100  -0.23 0.01                  Tell a lot of lies.         Honesty
q_501    99  -0.21 0.01                  Cheat to get ahead.         Honesty
q_4249   98  -0.21 0.02 Would call myself a nervous person.         Anxiety
q_1024   92  -0.21 0.02         Hang around doing nothing. EasyGoingness

Criterion =  health
       Freq mean.r sd.r                                               item          L27
q_820   100   0.35 0.01                     Feel comfortable with myself.   WellBeing
q_2765  100   0.35 0.01                            Am happy with my life.   WellBeing
q_811   100  -0.34 0.01 Feel a sense of worthlessness or hopelessness.   WellBeing
q_578   100  -0.34 0.02                                   Dislike myself.   WellBeing
q_1371  100   0.31 0.02                                        Love life.   WellBeing
q_56     97   0.28 0.02                  Am able to control my cravings. SelfControl
q_1505  100  -0.27 0.01                                    Panic easily.     Anxiety
q_4249   91  -0.26 0.01              Would call myself a nervous person.     Anxiety
```

Making up data: Be sure to call this simulation!

1. Multiple simulation functions
2. See ?sim for a list

sim	Create a factor simplex
sim.simplex	A data simplex
sim.hierarchical	A hierarchical factor structure
simCor	Create data from a specified correlation matrix
sim.parallel	To compare alternative factor solutions
sim.structural	Simulate complex structural models

See the help pages for details.
Read the vignettes.

Results returned

1. Most (but not all simulation functions return:
    - A raw data set
    - The model used for simulation
    - ‘True scores’ used to create the data

Miscellaneous functions that are useful

vJoin	Merge two files by rownames and column names
scrub	Clean up data
df2latex	One of several functions to create LaTeXtables.
read.file	and `read.clipboard` for convenient input

1. An introduction to the *psych* package: Part I:.
2. Intro:Part II: Scale construction and psychometrics
3. Installing Rand the *psych* package
4. Using R and psych to find ω
5. How To: Use *psych* for factor analysis and data reduction
6. Using R to score personality scales
7. Using *psych* for regression and mediation analysis

Athenstaedt, U. (2003). On the content and structure of the gender role self-concept: Including gender-stereotypical behaviors in addition to traits. *Psychology of Women Quarterly*, *27*(4), 309–318.

Cohen, J. (1988). *Statistical power analysis for the behavioral sciences* (2nd ed ed.). Hillsdale, N.J.: L. Erlbaum Associates.

Condon, D. M. (2018). *The SAPA Personality Inventory: An empirically-derived, hierarchically-organized self-report personality assessment model*. PsyArXiv /sc4p9/.

Condon, D. M. & Revelle, W. (2014). The International Cognitive Ability Resource: Development and initial validation of a public-domain measure. *Intelligence*, *43*, 52–64.

Cureton, E. E. (1950). Validity, reliability, and baloney. *Educational and Psychological Measurement*, *10*(1), 94–96.

Dwyer, P. S. (1937). The determination of the factor loadings of a given test from the known factor loadings of other tests. *Psychometrika*, *2*(3), 173–178.

Eagly, A. H. & Revelle, W. (2022). Understanding the Magnitude of Psychological Differences Between Women and Men Requires Seeing the Forest and the Trees. *Perspectives on Psychological Science*, *17*(5), 1339–1358.

Goldberg, L. R. (2006). Doing it all bass-ackwards: The development of hierarchical factor structures from the top down. *Journal of Research in Personality*, *40*(4), 347 – 358.

Gorsuch, R. L. (1983). *Factor analysis* (2nd ed.). Hillsdale, N.J.: L. Erlbaum Associates.

Gorsuch, R. L. (1997). New procedure for extension analysis in exploratory factor analysis. *Educational and Psychological Measurement*, *57*, 725–740.

Holm, S. (1979). A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, *6*(2), pp. 65–70.

Holzinger, K. & Swineford, F. (1937). The bi-factor method. *Psychometrika*, *2*(1), 41–54.

Horn, J. L. (1973). On extension analysis and its relation to correlations between variables and factor scores. *Multivariate Behavioral Research*, *8*(4), 477 – 489.

Horn, J. L. & Engstrom, R. (1979). Cattell's scree test in relation to Bartlett's chi-square test and other observations on the number of factors problem. *Multivariate Behavioral Research*, *14*(3), 283–300.

Jensen, A. R. & Weng, L.-J. (1994). What is a good g? *Intelligence*, *18*(3), 231–258.

Mosier, C. (1938). A note on Dwyer: The determination of the factor loadings of a given test. *Psychometrika*, *3*(4), 297–299.

Reise, S. P. (2012). The rediscovery of bifactor measurement models. *Multivariate Behavioral Research*, *47*(5), 667–696.

Revelle, W. (1979). Hierarchical cluster-analysis and the internal structure of tests. *Multivariate Behavioral Research*, *14*(1), 57–74.

Revelle, W. & Anderson, K. J. (1998). Personality, motivation and cognitive performance: Final report to the army research institute on contract MDA 903-93-K-0008. Technical report, Northwestern University, Evanston, Illinois, USA.

Revelle, W. & Condon, D. M. (2019). Reliability: from alpha to omega. *Psychological Assessment*, *31*(12), 1395–1411.

Revelle, W. & Rocklin, T. (1979). Very Simple Structure - alternative procedure for estimating the optimal number of interpretable factors. *Multivariate Behavioral Research*, *14*(4), 403–414.

Rodriguez, A., Reise, S. P., & Haviland, M. G. (2016). Evaluating bifactor models: Calculating and interpreting statistical indices. *Psychological methods*, *21*(2), 137–150.

Schmid, J. J. & Leiman, J. M. (1957). The development of hierarchical factor solutions. *Psychometrika*, *22*(1), 83–90.

Velicer, W. (1976). Determining the number of components from the matrix of partial correlations. *Psychometrika*, *41*(3), 321–327.

Waller, N. (2007). A general method for computing hierarchical component structures by Goldberg's Bass-Ackwards method. *Journal of Research in Personality*, *41*(4), 745 – 752.

Widaman, K. F. & Revelle, W. (2022). Thinking thrice about sum scores, and then some more about measurement and analysis. *Behavior Research Methods*, 1–19.