psych: a general purpose toolkit for personality and psychological research

Presented at the Southwestern Psychological Association meeting, San Antonio, Texas April, 2017

William Revelle

Department of Psychology
Northwestern University

slides: `http://personality-project.org/r/tutorials/swpa/swpa.pdf`

April 1, 2017

Outline

Installing the psych package ($\geq$ 1.7.3)

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()
R version 3.3.3 Patched (2017-03-06 r72351)
Platform: x86_64-apple-darwin13.4.0 (64-bit)
Running under: macOS Sierra 10.12.3

locale:
[1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
```

Show all the functions in the psych package

```
objects("package:psych")
```

```
 [1] "%+%"                    "ability"                "affect"                 "all.income"
  [6] "anova.psych"           "autoR"                  "Bechtoldt"              "Bechtoldt.1
...
[81] "cushny"                "d2r"                    "densityBy"              "describe"
 [86] "describeBy"           "describeData"           "df2latex"               "dfOrder"
...
[111] "error.bars"           "error.bars.by"          "error.bars.tab"         "error.cross
[116] "errorCircles"         "esem"                   "esem.diagram"           "fa"
...
[131] "fa.plot"              "fa.poly"                "fa.rgraph"              "fa.sort"
[136] "fa2irt"               "fa2latex"               "faBy"                   "fac"
...
[176] "ICC"                  "ICC2latex"              "iclust"                 "ICLUST"
[181] "iclust.diagram"       "ICLUST.graph"           "ICLUST.rgraph"          "iclust.sort
...
[201] "irt.fa"               "irt.item.diff.rasch"    "irt.person.rasch"       "irt.respons
[206] "irt.stats.like"       "irt.tau"                "irt2latex"              "isCorrelati
[211] "item.lookup"          "item.sim"               "kaiser"                 "keys.lookup
...
[251] "omega"                "omega.diagram"          "omega.graph"            "omega2latex
[256] "omegah"               "omegaSem"               "outlier"                "p.rep"
...
[301] "read.clipboard"       "read.clipboard.csv"     "read.clipboard.fwf"     "read.clipbo
[306] "read.clipboard.upper" "read.file"              "read.file.csv"          "read.https'
...
[326] "score.alpha"          "score.irt"              "score.irt.2"            "score.irt.p
[331] "score.multiple.choice" "scoreFast"             "scoreIrt"               "scoreIrt.1p
[336] "scoreItems"           "scoreOverlap"           "scree"                  "scrub"
...
[356] "sim.multi"            "sim.multilevel"         "sim.npl"                "sim.npn"
...
```

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

From a website: define the file name

R code

```
fn <- "http://personality-project.org/r/datasets/Maps.mixx.msq1.epi.bf.txt"
fn  #show it to check
[1] "http://personality-project.org/r/datasets/Maps.mixx.msq1.epi.bf.txt"
mydata <- read.file(fn,header=TRUE)
```

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, et
```

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
```

Checking the data using `describe`

```
> dim(mydata)
[1] 231  86
> describe(mydata)
            vars   n   mean    sd median trimmed   mad min max range  skew kurtosis   se
id             1 231  66.82 45.13     58   64.14 50.41   1 160   159  0.45    -0.96 2.97
delighted      2 231   0.82  1.05      1    0.68  1.48   0   9     9  2.46    14.15 0.07
sociable       3 231   1.32  0.96      1    1.28  1.48   0   3     3  0.06    -1.03 0.06
jittery        4 231   0.55  0.78      0    0.39  0.00   0   3     3  1.37     1.23 0.05
hostile        5 231   0.35  0.85      0    0.17  0.00   0   9     9  5.34    45.21 0.06
sluggish       6 231   1.21  0.96      1    1.14  1.48   0   3     3  0.47    -0.70 0.06
depressed      7 231   0.56  0.83      0    0.39  0.00   0   3     3  1.45     1.37 0.05
...
ashamed       71 231   0.32  1.15      0    0.06  0.00   0   9     9  5.92    40.25 0.08
anxious       72 231   0.75  1.26      0    0.53  0.00   0   9     9  3.85    21.39 0.08
idle          73 231   0.98  1.15      1    0.83  1.48   0   9     9  3.11    18.20 0.08
epiE          74 231  13.33  4.14     14   13.49  4.45   1  22    21 -0.33    -0.06 0.27
epiS          75 231   7.58  2.69      8    7.77  2.97   0  13    13 -0.57    -0.02 0.18
epiImp        76 231   4.37  1.88      4    4.36  1.48   0   9     9  0.06    -0.62 0.12
epilie        77 231   2.38  1.50      2    2.27  1.48   0   7     7  0.66     0.24 0.10
...
traitanx      85 231  39.01  9.52     38   38.36  8.90  22  71    49  0.67     0.47 0.63
stateanx      86 231  39.85 11.48     38   38.92 10.38  21  79    58  0.72    -0.01 0.76
```

Cleaning the data using `scrub`

We want to change 9s in variables 2 - 73 into NA

```
> cleaned <- scrub(mydata,where=2:73,isvalue=9,newvalue=NA)
> describe(cleaned)
           vars   n   mean    sd median trimmed   mad min max range  skew kurtosis   se
id            1 231  66.82 45.13   58.0   64.14 50.41   1 160   159  0.45    -0.96 2.97
delighted     2 230   0.78  0.90    1.0    0.67  1.48   0   3     3  0.79    -0.52 0.06
sociable      3 231   1.32  0.96    1.0    1.28  1.48   0   3     3  0.06    -1.03 0.06
jittery       4 231   0.55  0.78    0.0    0.39  0.00   0   3     3  1.37     1.23 0.05
hostile       5 230   0.31  0.63    0.0    0.17  0.00   0   3     3  2.12     4.19 0.04
sluggish      6 231   1.21  0.96    1.0    1.14  1.48   0   3     3  0.47    -0.70 0.06
depressed     7 231   0.56  0.83    0.0    0.39  0.00   0   3     3  1.45     1.37 0.05
...
ashamed      71 228   0.21  0.57    0.0    0.05  0.00   0   3     3  3.00     8.95 0.04
anxious      72 228   0.64  0.84    0.0    0.51  0.00   0   3     3  1.11     0.32 0.06
idle         73 229   0.91  0.88    1.0    0.82  1.48   0   3     3  0.64    -0.42 0.06
epiE         74 231  13.33  4.14   14.0   13.49  4.45   1  22    21 -0.33    -0.06 0.27
epiS         75 231   7.58  2.69    8.0    7.77  2.97   0  13    13 -0.57    -0.02 0.18
epiImp       76 231   4.37  1.88    4.0    4.36  1.48   0   9     9  0.06    -0.62 0.12
epilie       77 231   2.38  1.50    2.0    2.27  1.48   0   7     7  0.66     0.24 0.10
...
traitanx     85 231  39.01  9.52   38.0   38.36  8.90  22  71    49  0.67     0.47 0.63
stateanx     86 231  39.85 11.48   38.0   38.92 10.38  21  79    58  0.72    -0.01 0.76
```

Multiple ways to graphically display data

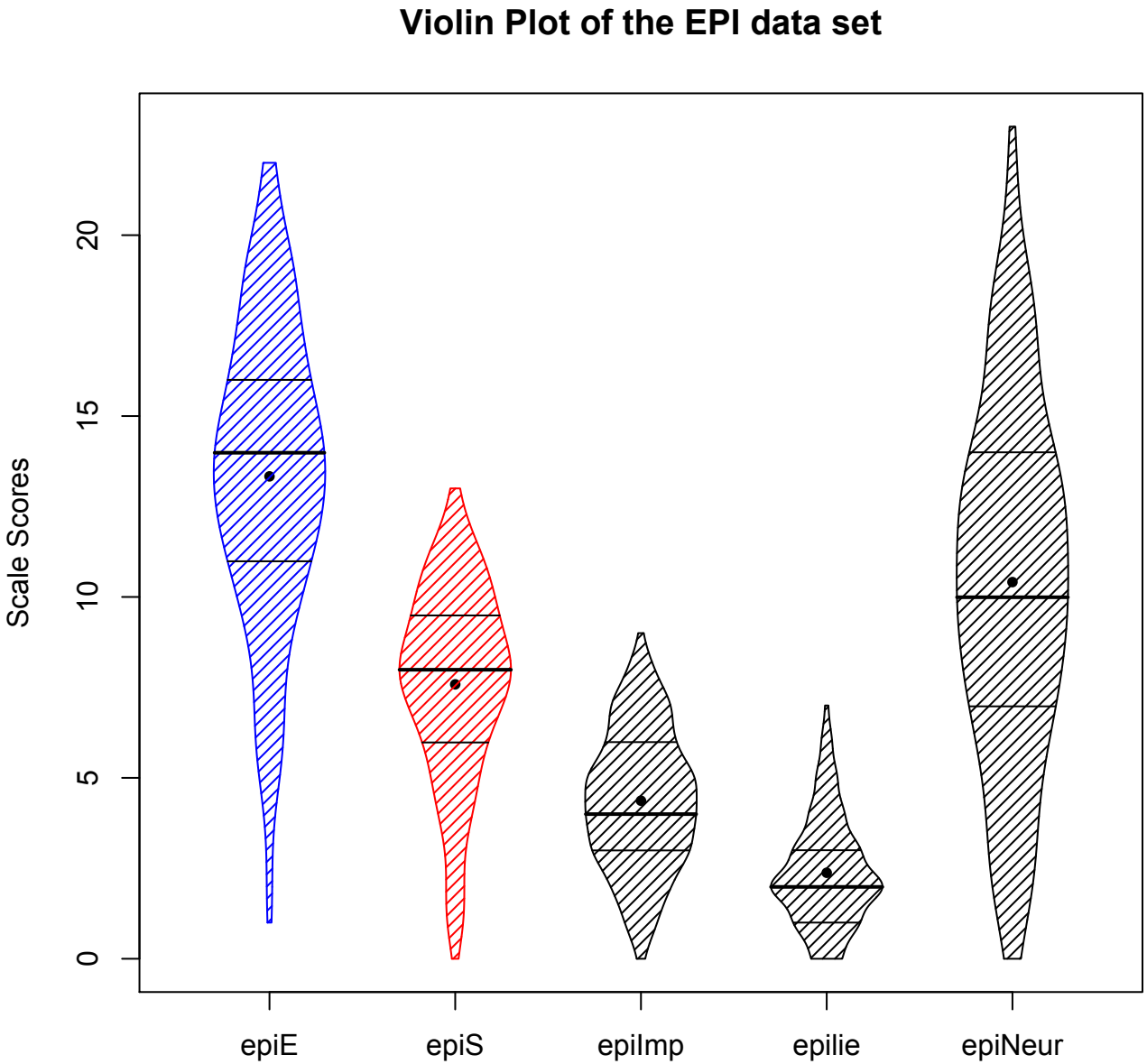
1. box.plots (Core R)
2. Violin plots (`violinBy` in *psych*)
3. Scatter Plot Matrix (SPLOM) plots (`pairs.panels` in *psych*)

First, lets just make a smaller data.frame and then issue two different graphic commands.

R code

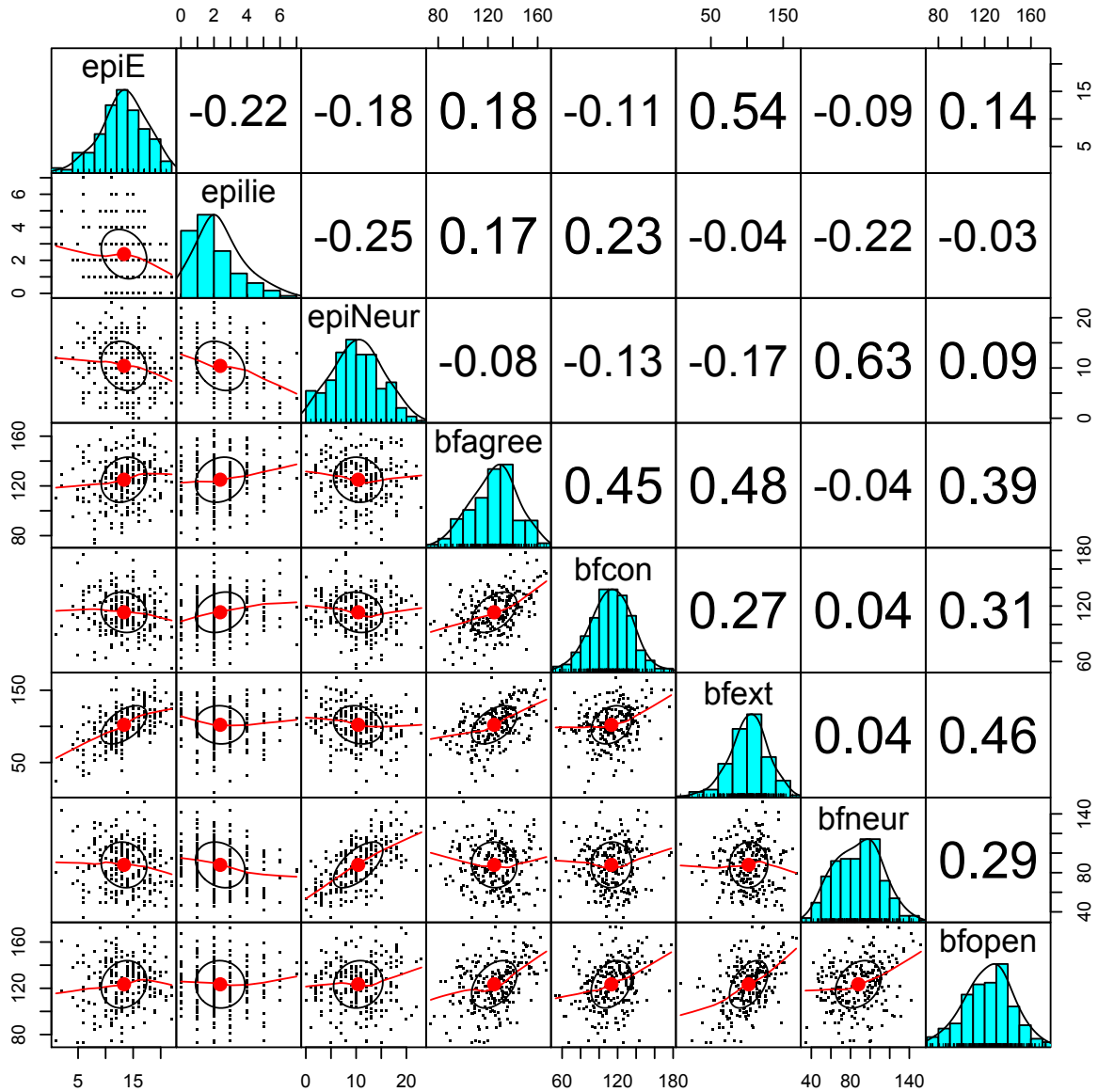
```
my.scales <- cleaned[74:86]
violinBy(my.scales[1:5])
pairs.panels(my.scales[c(1,4,5:10)],gap=0,pch=".")
```

Violin Plot `violinBy(my.scales[1:5])`



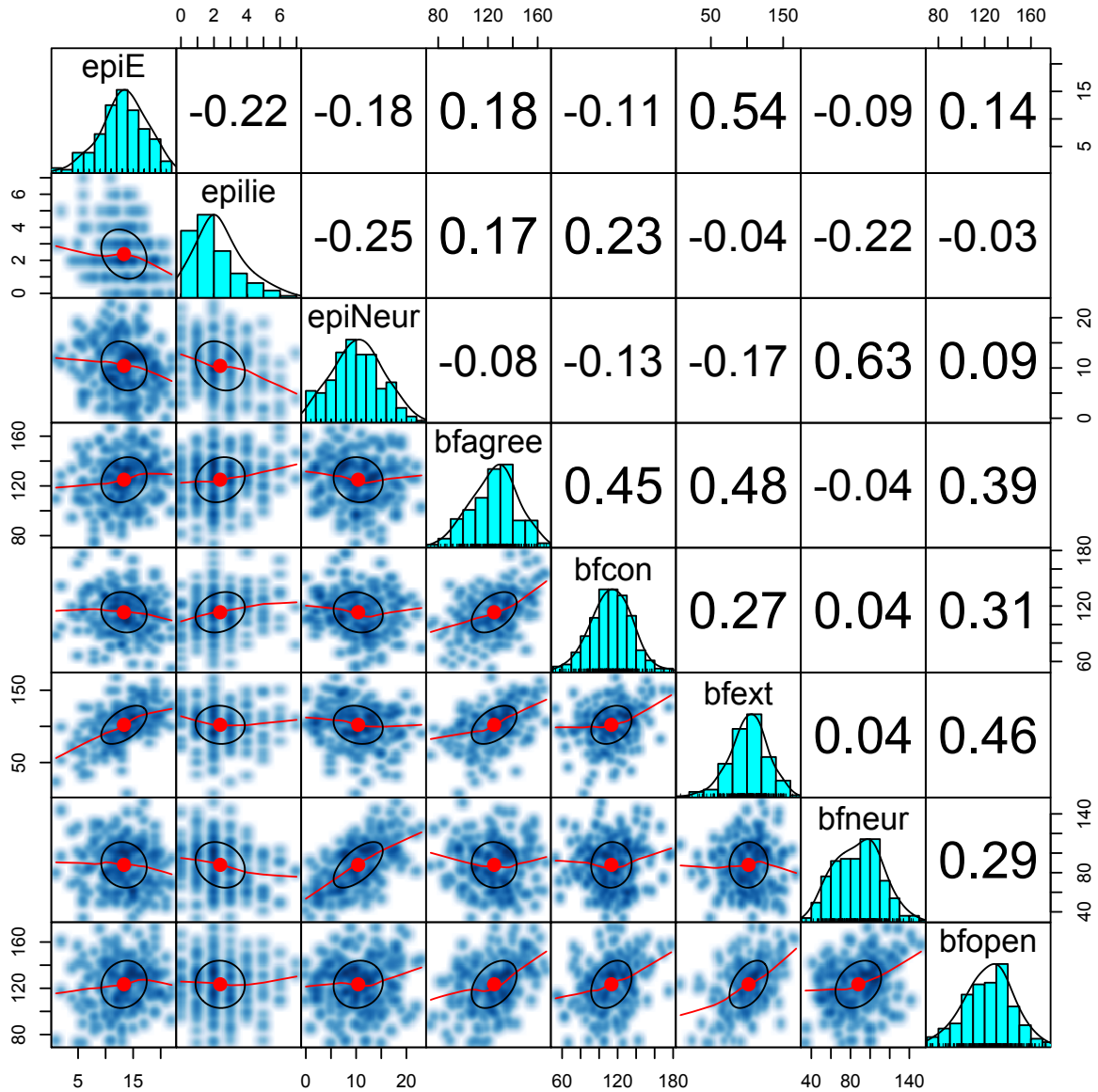
Scatter Plot of Matrices (SPLOM) of select variables

```
pairs.panels(my.scales[c(1,4,5:10)],gap=0,pch=".")
```



Scatter Plot of Matrices (SPLOM) of select variables

```
pairs.panels(my.scales[c(1,4,5:10)],gap=0,pch=".",smoother=TRU
```



Show a table of correlations

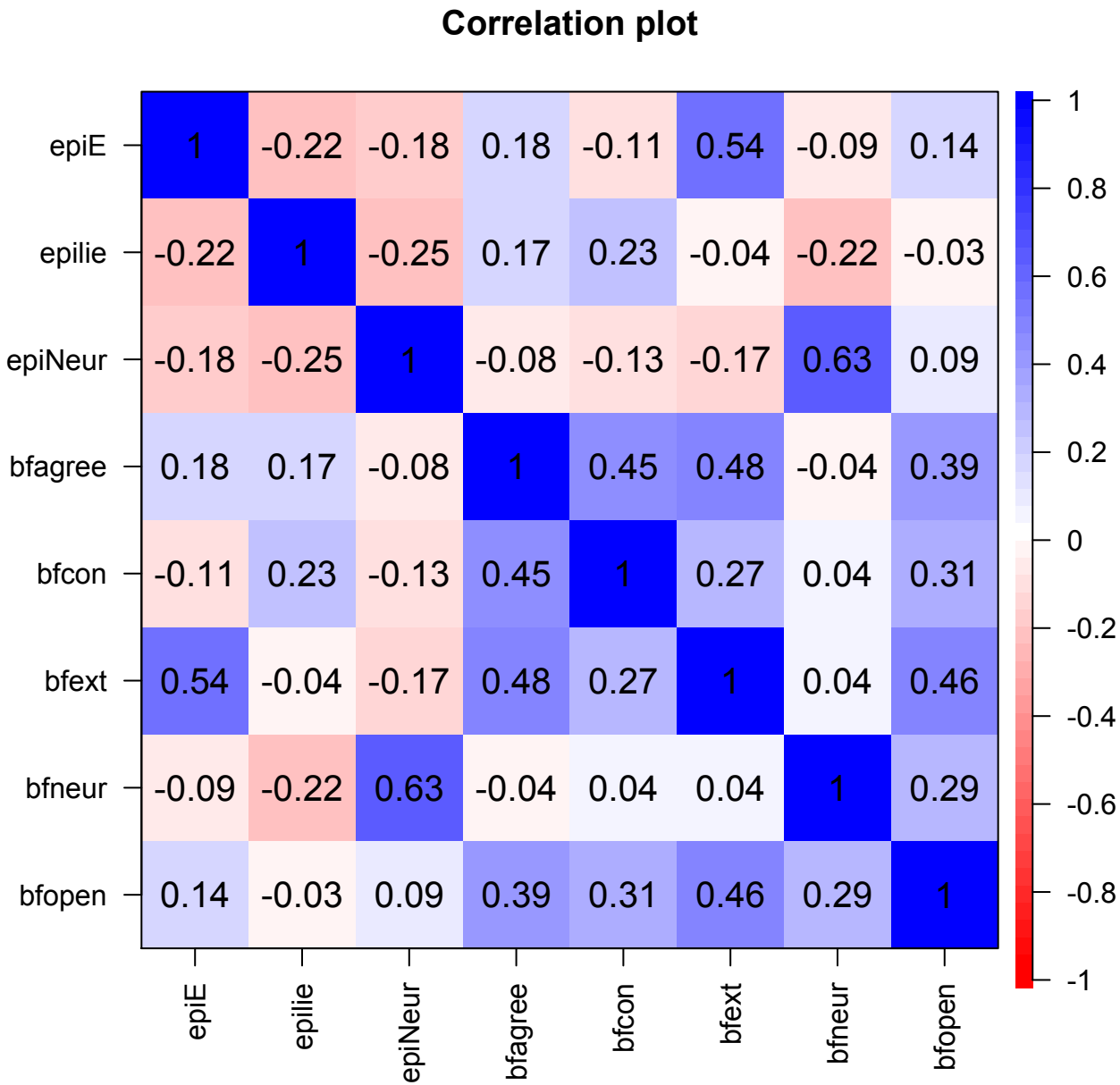
R code

```
R <- lowerCor(my.scales[c(1,4,5:10)])
cor.plot(R,numbers=TRUE)
```

```
          epiE  epili epiNr bfagr bfcon bfext bfner bfopn
epiE      1.00
epilie   -0.22  1.00
epiNeur  -0.18 -0.25  1.00
bfagree   0.18  0.17 -0.08  1.00
bfcon    -0.11  0.23 -0.13  0.45  1.00
bfext     0.54 -0.04 -0.17  0.48  0.27  1.00
bfneur   -0.09 -0.22  0.63 -0.04  0.04  0.04  1.00
bfopen    0.14 -0.03  0.09  0.39  0.31  0.46  0.29  1.00
```

Automatically calls the `cor` and `round` functions with default parameters and then does a pretty print out using `lowerMat`. Invisibly returns the full (square) matrix of unrounded values.

A simple heat map using `cor.plot`



Testing and displaying the "significance" of a set of correlations

1. Normal theory `corr.test`
   - raw probabilities as well as with a Holm adjusted for multiple correlations
2. Display these with `cor.plot`
3. Boot strapped confidence intervals based significance using `cor.ci`
   - Graphic displays correlations scaled by "significance"
   - Graphic displays of probability of correlation using `plot.cor.upperLowerCi`

Code for the next slides

R code

```
corr.test(my.scales[c(1,4,5:10)])
ci <- cor.ci(my.scales[c(1,4,5:10)])
cor.plot.upperLowerCi(ci). #temporarily unavailable -- patch coming i
```

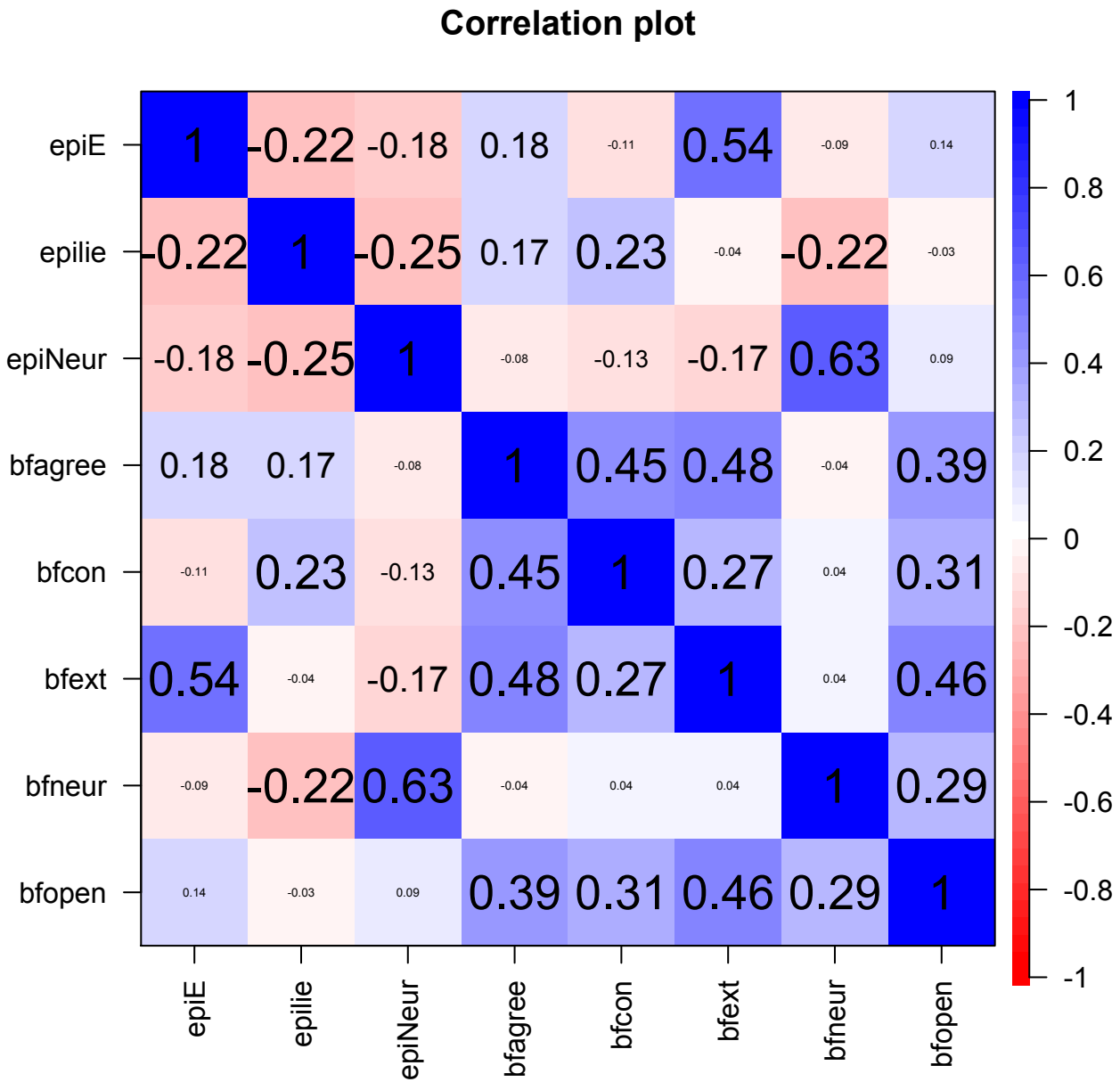
Normal theory test of correlations using `corr.test`

R code

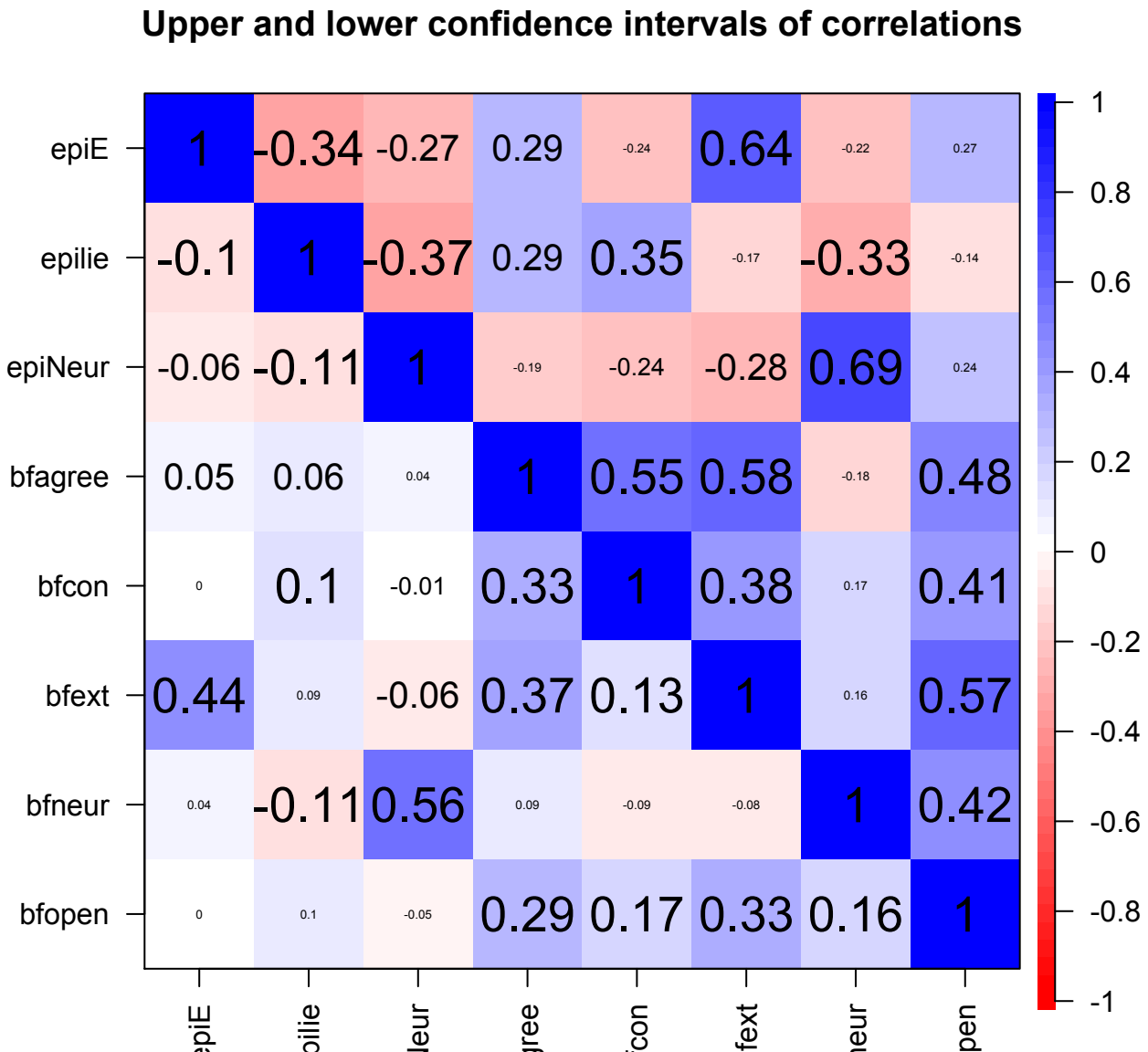
```
> corr.test(my.scales[c(1,4,5:10)])
```

```
Correlation matrix
         epiE epilie epiNeur bfagree bfcon bfext bfneur bfopen
epiE     1.00  -0.22   -0.18    0.18 -0.11  0.54  -0.09   0.14
epilie  -0.22   1.00   -0.25    0.17  0.23 -0.04  -0.22  -0.03
epiNeur -0.18  -0.25    1.00   -0.08 -0.13 -0.17   0.63   0.09
bfagree  0.18   0.17   -0.08    1.00  0.45  0.48  -0.04   0.39
bfcon   -0.11   0.23   -0.13    0.45  1.00  0.27   0.04   0.31
bfext    0.54  -0.04   -0.17    0.48  0.27  1.00   0.04   0.46
bfneur  -0.09  -0.22    0.63   -0.04  0.04  0.04   1.00   0.29
bfopen   0.14  -0.03    0.09    0.39  0.31  0.46   0.29   1.00
Sample Size
[1] 231
Probability values (Entries above the diagonal are adjusted for multiple tests.)
        epiE epilie epiNeur bfagree bfcon bfext bfneur bfopen
epiE    0.00   0.01    0.11    0.11  0.75  0.00   1.00    0.4
epilie  0.00   0.00    0.00    0.12  0.01  1.00   0.01    1.0
epiNeur 0.01   0.00    0.00    1.00  0.43  0.12   0.00    1.0
bfagree 0.01   0.01    0.21    0.00  0.00  0.00   1.00    0.0
bfcon   0.08   0.00    0.04    0.00  0.00  0.00   1.00    0.0
bfext   0.00   0.50    0.01    0.00  0.00  0.00   1.00    0.0
bfneur  0.15   0.00    0.00    0.50  0.50  0.57   0.00    0.0
bfopen  0.04   0.70    0.19    0.00  0.00  0.00   0.00    0.0

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

Heat map scaled by "significance" using `cor.ci`



Heat map scaled by "significance" show upper and lower bounds–patched in 1.7.4



Multiple types of reliability

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

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 misspecify the data

R code

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

```
Some items ( A1 ) were negatively correlated with the total scale and
probably should be reversed.
To do this, run the function again with the 'check.keys=TRUE' option
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
A1 2784 0.066 0.024 -0.39  -0.31  2.4 1.4     <-- need to rekey this item
A2 2773 0.630 0.666  0.58   0.37  4.8 1.2
A3 2774 0.724 0.742  0.72   0.48  4.6 1.3
A4 2781 0.686 0.661  0.50   0.37  4.7 1.5
A5 2784 0.700 0.719  0.64   0.45  4.6 1.3

Non missing response frequency for each item
      1    2    3    4    5    6 miss
```

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 automatically reversed.
 This is indicated by a negative sign for the variable name.
```

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.freq" "keys"
  "scores"        "nvar"          "boot.ci"
 [9] "boot"          "Unidim"        "Fit"           "call"          "title"

    $ 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:
  ..$ n     : num [1:5] 2784 2773 2774 2781 2784
  ..$ raw.r : num [1:5] 0.581 0.728 0.76 0.654 0.687
  ..$ std.r : num [1:5] 0.566 0.748 0.767 0.631 0.699
  ..$ r.cor : num [1:5] 0.376 0.667 0.709 0.471 0.596
  ..$ r.drop: num [1:5] 0.308 0.564 0.587 0.394 0.489
  ..$ mean  : num [1:5] 4.59 4.8 4.6 4.7 4.56
  ..$ sd    : num [1:5] 1.41 1.17 1.3 1.48 1.26
```

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

R code

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

But, since there 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 ku
X1    1 2800 23.08 4.54     24   23.43 4.45   5  30    25 -0.73
```

Perhaps a more useful case: scoring multiple scales using `scoreItems`

First, define the scoring keys, and then call `scoreitems` Use the `msq` data set

R code

```
keys <- make.keys(msq[1:75],list(
+ EA = c("active", "energetic", "vigorous", "wakeful", "wide.awake", "full.of.pep",
+        "lively", "-sleepy", "-tired", "-drowsy"),
+ TA =c("intense", "jittery", "fearful", "tense", "clutched.up", "-quiet", "-still",
+        "-placid", "-calm", "-at.rest") ,
+ PA =c("active", "excited", "strong", "inspired", "determined", "attentive",
+           "interested", "enthusiastic", "proud", "alert"),
+ NAf =c("jittery", "nervous", "scared", "afraid", "guilty", "ashamed", "distressed",
+          "upset", "hostile", "irritable" ))
+ )
> msq.scores <- scoreItems(keys,msq[1:75])
> msq.scores
```

```
  all: scoreItems(keys = keys, items = msq[1:75])

(Unstandardized) Alpha:
        EA   TA   PA  NAf
alpha 0.93 0.75 0.92 0.83

Standard errors of unstandardized Alpha:
         EA     TA     PA    NAf
ASE   0.004 0.0082 0.0044 0.0064

Average item correlation:
            EA   TA   PA  NAf
average.r 0.58 0.23 0.52 0.33
```

Score multiple scales (continued)

```
Standard errors of unstandardized Alpha:
         EA     TA     PA    NAf
ASE   0.004 0.0082 0.0044 0.0064

Average item correlation:
            EA   TA   PA  NAf
average.r 0.58 0.23 0.52 0.33

 Guttman 6* reliability:
           EA   TA   PA  NAf
Lambda.6 0.96 0.82 0.93 0.88

Signal/Noise based upon av.r :
             EA TA PA NAf
Signal/Noise 14  3 11 4.9

Scale intercorrelations corrected for attenuation
 raw correlations below the diagonal, alpha on the diagonal
 corrected correlations above the diagonal:
        EA   TA    PA    NAf
EA   0.932 0.29 0.870 -0.069
TA   0.238 0.75 0.226  0.710
PA   0.804 0.19 0.915  0.044
NAf -0.061 0.56 0.039  0.831
```

More detailed item statistics

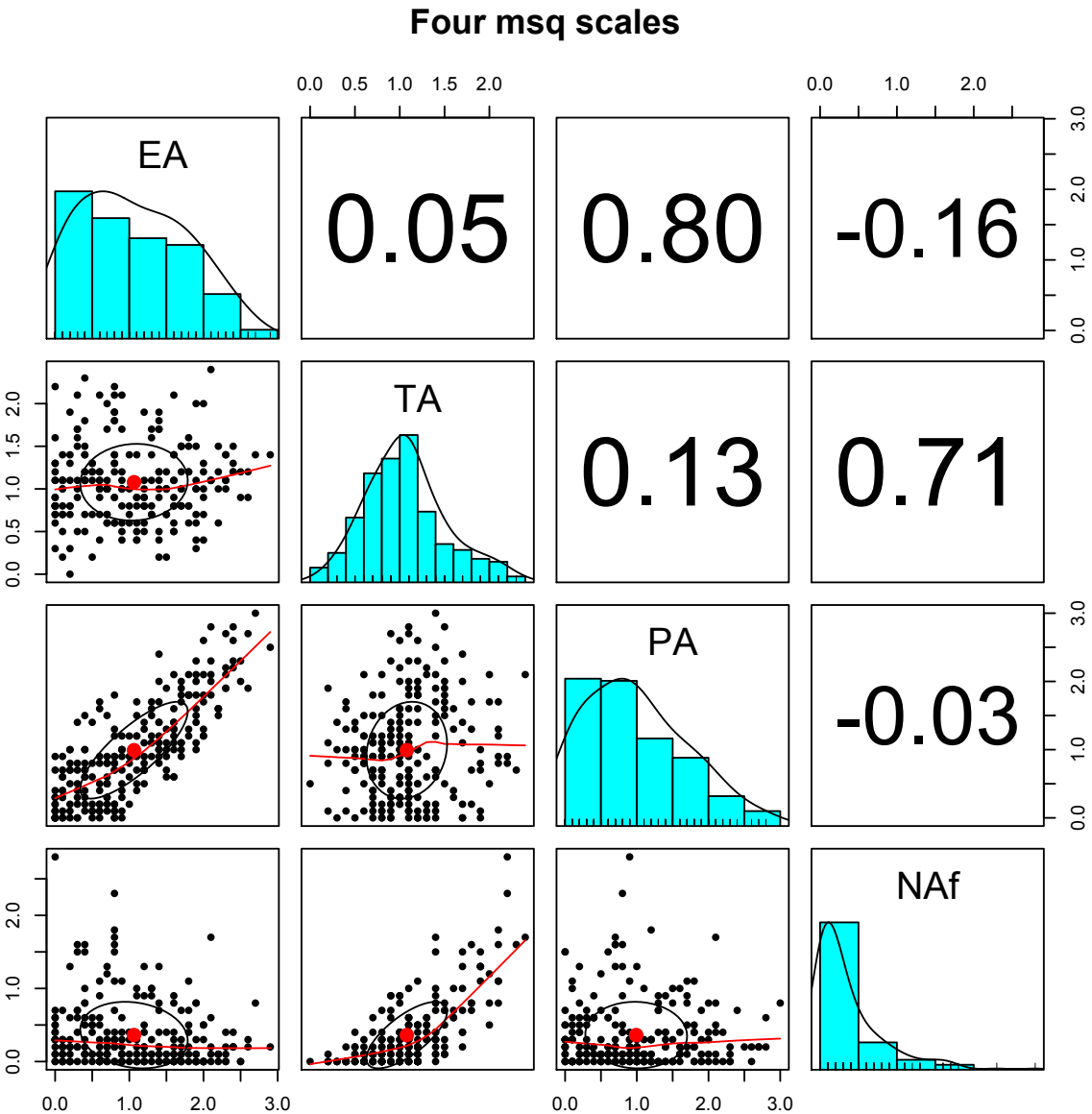
R code

```
print(msq.scores,short=FALSE)
```

```
Item by scale correlations:
 corrected for item overlap and scale reliability
                EA    TA    PA   NAf
delighted     0.56  0.04  0.67 -0.17
sociable      0.57  0.06  0.64 -0.15
jittery       0.15  0.52  0.23  0.46
hostile      -0.21  0.37 -0.13  0.58
sluggish     -0.65 -0.02 -0.40  0.21
depressed    -0.30  0.44 -0.26  0.67
satisfied     0.54 -0.23  0.62 -0.35
relaxed       0.35 -0.56  0.40 -0.51
warmhearted   0.47 -0.09  0.66 -0.19
blue         -0.23  0.40 -0.19  0.64
intense       0.25  0.42  0.44  0.47
strong        0.55  0.00  0.69 -0.03
scared       -0.05  0.61  0.08  0.75
enthusiastic  0.69  0.16  0.83 -0.07
proud         0.57  0.05  0.73 -0.11
sad          -0.22  0.46 -0.18  0.73
active        0.71  0.18  0.83 -0.05
full.of.pep   0.80  0.05  0.80 -0.15
unhappy      -0.30  0.44 -0.26  0.70
lively        0.78  0.04  0.81 -0.13
```

```
Non missing response frequency for each item
                0    1    2    3 miss
delighted    0.50 0.27 0.19 0.04 0.00
sociable     0.24 0.30 0.35 0.11 0.00
jittery      0.60 0.28 0.09 0.03 0.00
hostile      0.77 0.17 0.05 0.01 0.00
sluggish     0.24 0.44 0.19 0.13 0.00
depressed    0.61 0.26 0.08 0.05 0.00
satisfied    0.22 0.29 0.38 0.12 0.00
relaxed      0.13 0.20 0.43 0.24 0.00
warmhearted  0.17 0.23 0.37 0.22 0.00
blue         0.60 0.30 0.08 0.02 0.00
intense      0.54 0.28 0.15 0.03 0.00
strong       0.31 0.26 0.32 0.11 0.00
scared       0.80 0.15 0.04 0.01 0.00
enthusiastic 0.43 0.29 0.19 0.08 0.00
proud        0.41 0.24 0.26 0.10 0.00
sad          0.66 0.22 0.09 0.03 0.00
active       0.38 0.32 0.22 0.08 0.00
full.of.pep  0.54 0.21 0.20 0.05 0.00
unhappy      0.65 0.24 0.07 0.04 0.00
lively       0.44 0.28 0.22 0.06 0.00
```

Show the SPLOM of the msq scales using `pairs.panels`



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 straightfoward 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

Correcting for item overlap using scoreOverlap

R code

```
small.msq <- msq[ c("active", "energetic", "vigorous", "wakeful", "wi
"full.of.pep", "lively", "sleepy", "tired", "drowsy","intense", "jitte
 "tense", "clutched.up", "quiet", "still",    "placid", "calm", "at.re
small.R <- cor(small.msq,use="pairwise")
keys.list <- list(
EA = c("active", "energetic", "vigorous", "wakeful", "wide.awake", "fu
       "lively", "-sleepy", "-tired", "-drowsy"),
TA =c("intense", "jittery", "fearful", "tense", "clutched.up", "-quiet
       "-placid", "-calm", "-at.rest") ,

high.EA = c("active", "energetic", "vigorous", "wakeful", "wide.awake
       "lively"),
low.EA =c("sleepy", "tired", "drowsy"),
lowTA= c("quiet", "still", "placid", "calm", "at.rest"),
highTA = c("intense", "jittery", "fearful", "tense", "clutched.up")
   )

keys <- make.keys(small.R,keys.list)

adjusted.scales <- scoreOverlap(keys.list,small.R)
```

Correcting for item overlap using `scoreOverlap`. (continued)

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

(Standardized) Alpha:
     EA      TA high.EA  low.EA   lowTA  highTA
   0.93    0.75    0.94    0.93    0.73    0.76

(Standardized) G6*:
     EA      TA high.EA  low.EA   lowTA  highTA
   0.88    0.68    0.94    0.90    0.73    0.75

Average item correlation:
     EA      TA high.EA  low.EA   lowTA  highTA
   0.59    0.23    0.68    0.81    0.35    0.38

Number of items:
     EA      TA high.EA  low.EA   lowTA  highTA
     10      10       7       3       5       5
Signal to Noise ratio based upon average r and n
     EA      TA high.EA  low.EA   lowTA  highTA
   14.1     3.0    14.8    12.9     2.7     3.1

Scale intercorrelations corrected for item overlap and attenuation
 adjusted for overlap correlations below the diagonal, alpha on the diagonal
 corrected correlations above the diagonal:
           EA    TA high.EA low.EA lowTA highTA
EA       0.93  0.27   0.965 -0.803 -0.18  0.253
TA       0.23  0.75   0.282 -0.167 -0.81  0.821
high.EA  0.90  0.24   0.937 -0.620 -0.12  0.324
low.EA  -0.75 -0.14  -0.579  0.928  0.25 -0.023
lowTA   -0.15 -0.60  -0.098  0.204  0.73 -0.335
highTA   0.21  0.62   0.273 -0.019 -0.25  0.757
```

Compare scoreOverlap with non-adjusted

R code

```
adjusted.scales <- scoreOverlap(keys.list,small.R)
raw <- scoreItems(keys.list,small.R)
```

```
Scale intercorrelations corrected for item overlap and attenuation
 adjusted for overlap correlations below the diagonal, alpha on the diagonal
 corrected correlations above the diagonal:
           EA    TA high.EA low.EA lowTA highTA
EA       0.93  0.27   0.965 -0.803 -0.18  0.253
TA       0.23  0.75   0.282 -0.167 -0.81  0.821
high.EA  0.90  0.24   0.937 -0.620 -0.12  0.324
low.EA  -0.75 -0.14  -0.579  0.928  0.25 -0.023
lowTA   -0.15 -0.60  -0.098  0.204  0.73 -0.335
highTA   0.21  0.62   0.273 -0.019 -0.25  0.757
```

```
Scale intercorrelations corrected for attenuation
 raw correlations below the diagonal, alpha on the diagonal
 corrected correlations above the diagonal:
           EA    TA high.EA low.EA lowTA highTA
EA       0.93  0.27   1.024 -0.848 -0.18  0.253
TA       0.23  0.75   0.282 -0.167 -1.06  1.056
high.EA  0.96  0.24   0.937 -0.620 -0.12  0.324
low.EA  -0.79 -0.14  -0.579  0.928  0.25 -0.023
lowTA   -0.15 -0.78  -0.098  0.204  0.73 -0.335
highTA   0.21  0.80   0.273 -0.019 -0.25  0.757
```

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.

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

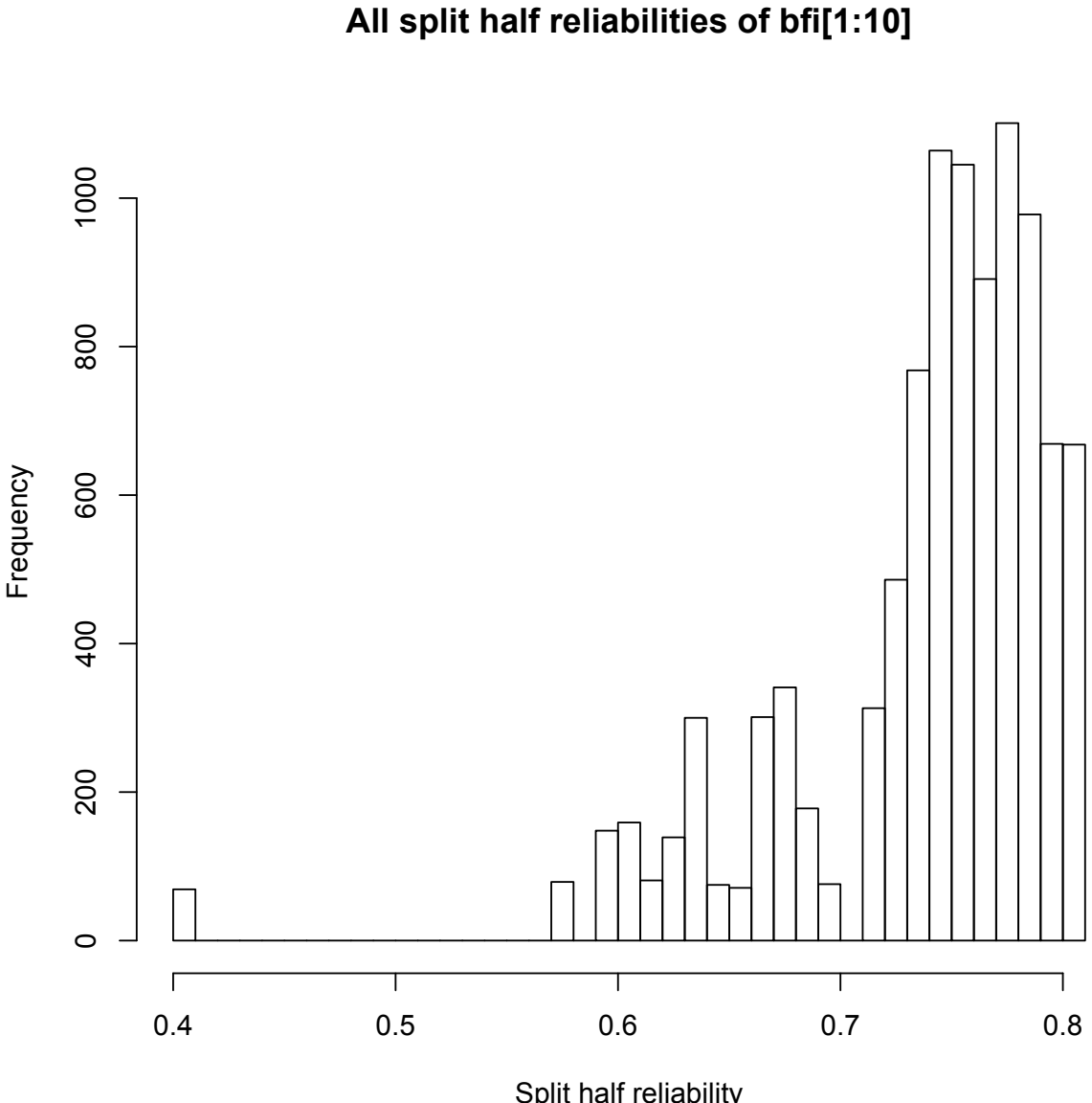
Find α and all split half reliabilities of 5 Agreeableness items and 5 Conscientiousness items from the `bfi` data set included in *psych*.

R code

```
alpha(bfi[1:10) #find alpha, let it automatically reverse items
splitHalf(bfi[1:10],keys=c(1,9,10)) #reverse 3 items
```

```
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], keys = 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.



Using the omega function

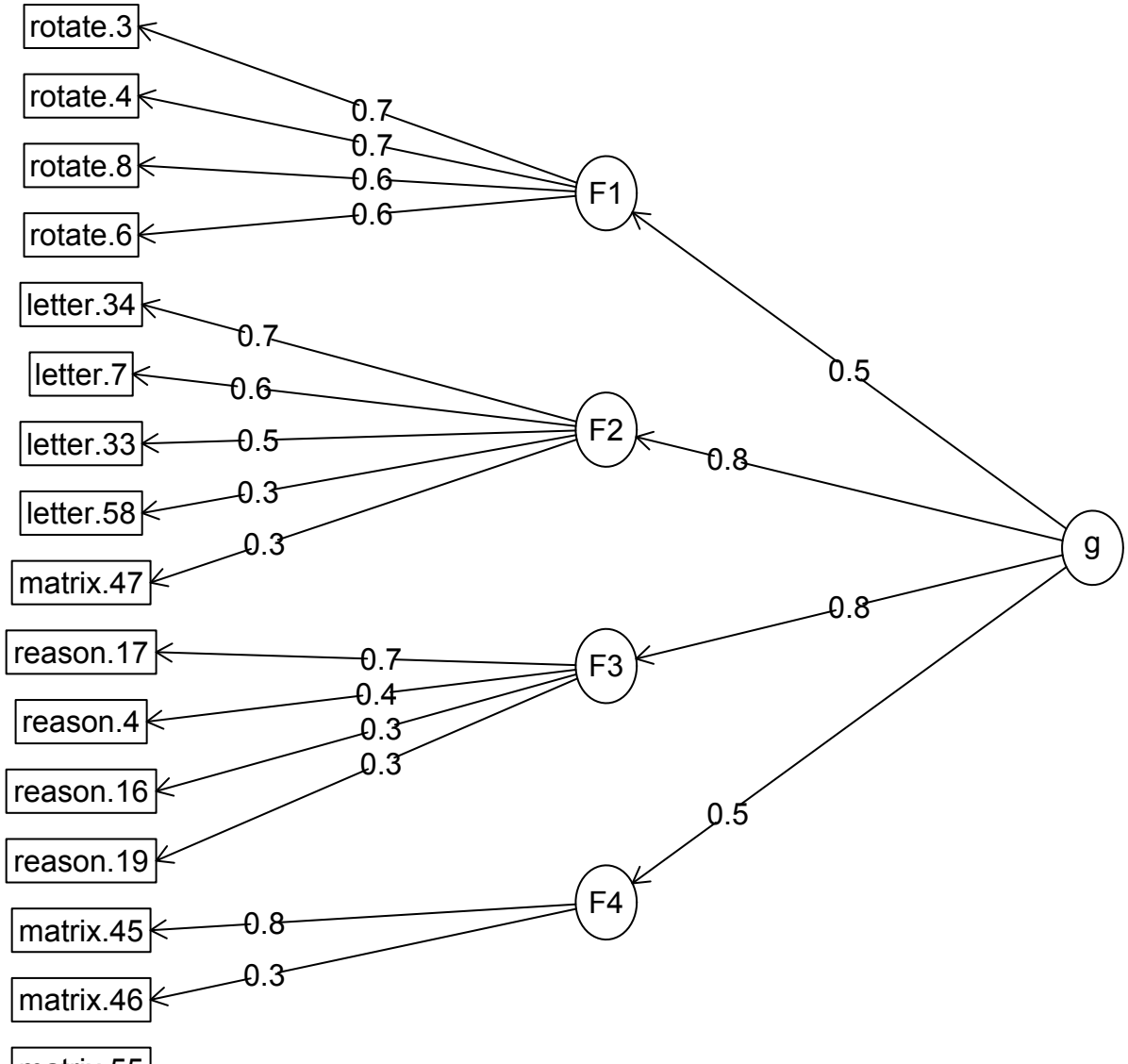
R code

```
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
```

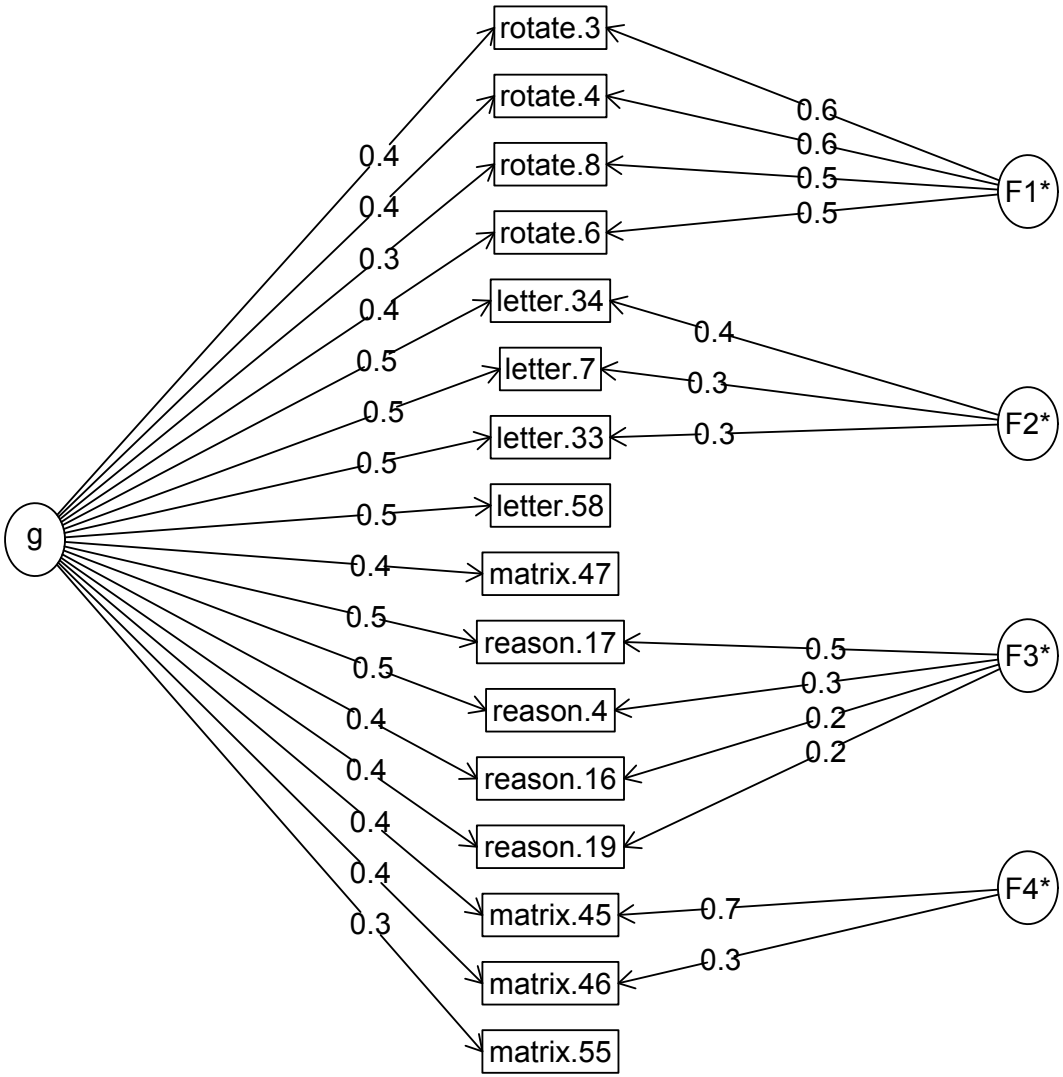
16 ability items from the International Cognitive Ability Resource

general abilty and 4 subfactors of ICAR data



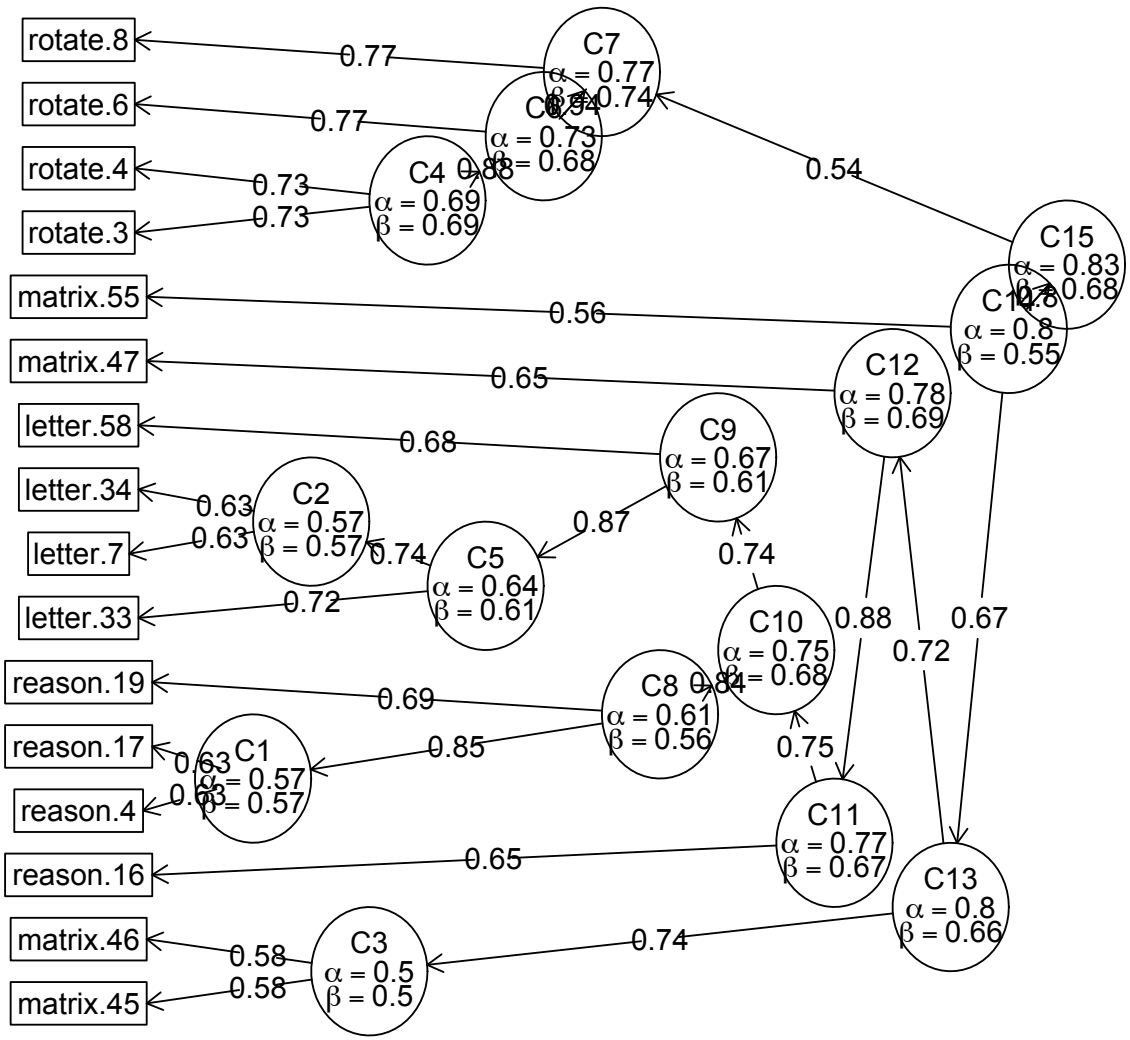
Schmid Leiman transformation of 16 ability items from ICAR

general abilty and 4 subfactors of ICAR data



Hierarchical clustering of 16 ICAR ability items: `iclust`(ability)

Hierarchical clustering of 16 ability items using iclust



Exploratory Factor Analysis

1. How many factors: an unsolved problem
   - Parallel analysis, MAPS, VSS, BIC, RMSEA, etc. available in `nfactors` and `fa.parallel`
2. Factor extraction algorithms available in the `fa` function
   - maximum likelihood, minimum residual, principal factor, ...
3. Factor rotation procedures are done using *GPArotation* package
   - orthogonal: varimax, quartimax, bifactor, ...
   - oblique: oblimin, geomin, biquartimin, ...
4. Displaying the solutions using `fa.plot`

Note, that EFA is not the same as Principal Components Analysis and the two should not be confused.

1. PCA done using `principal`

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." (Henry Kaiser)

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's Mininum Average Partial (MAP) is pretty good
3. For items, the Very Simple Structure (VSS) 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.

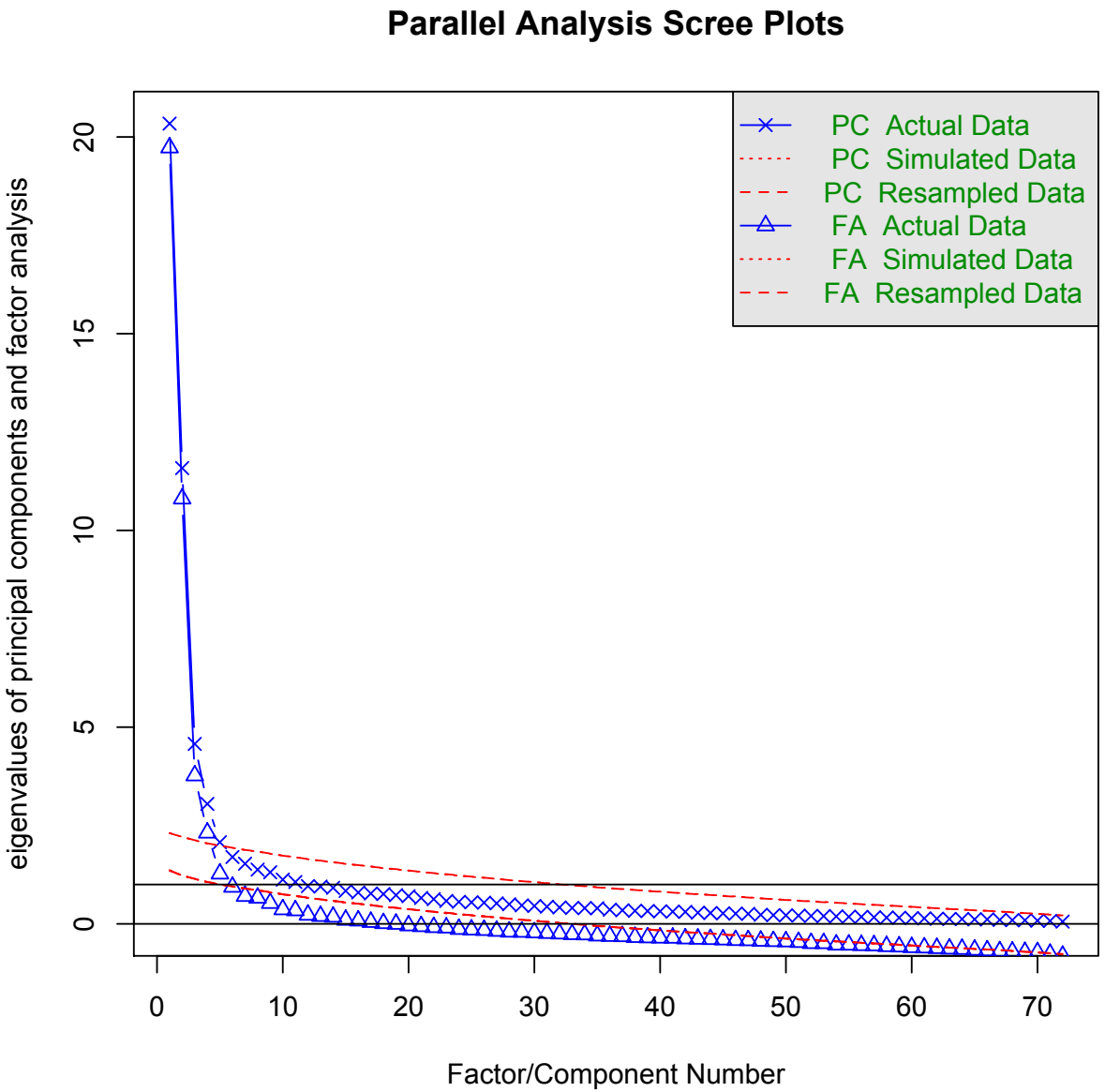
What about parallel analysis? Pearson R or polychoric ρ?

Of the raw (Pearson) correlations compared to the polychoric correlations

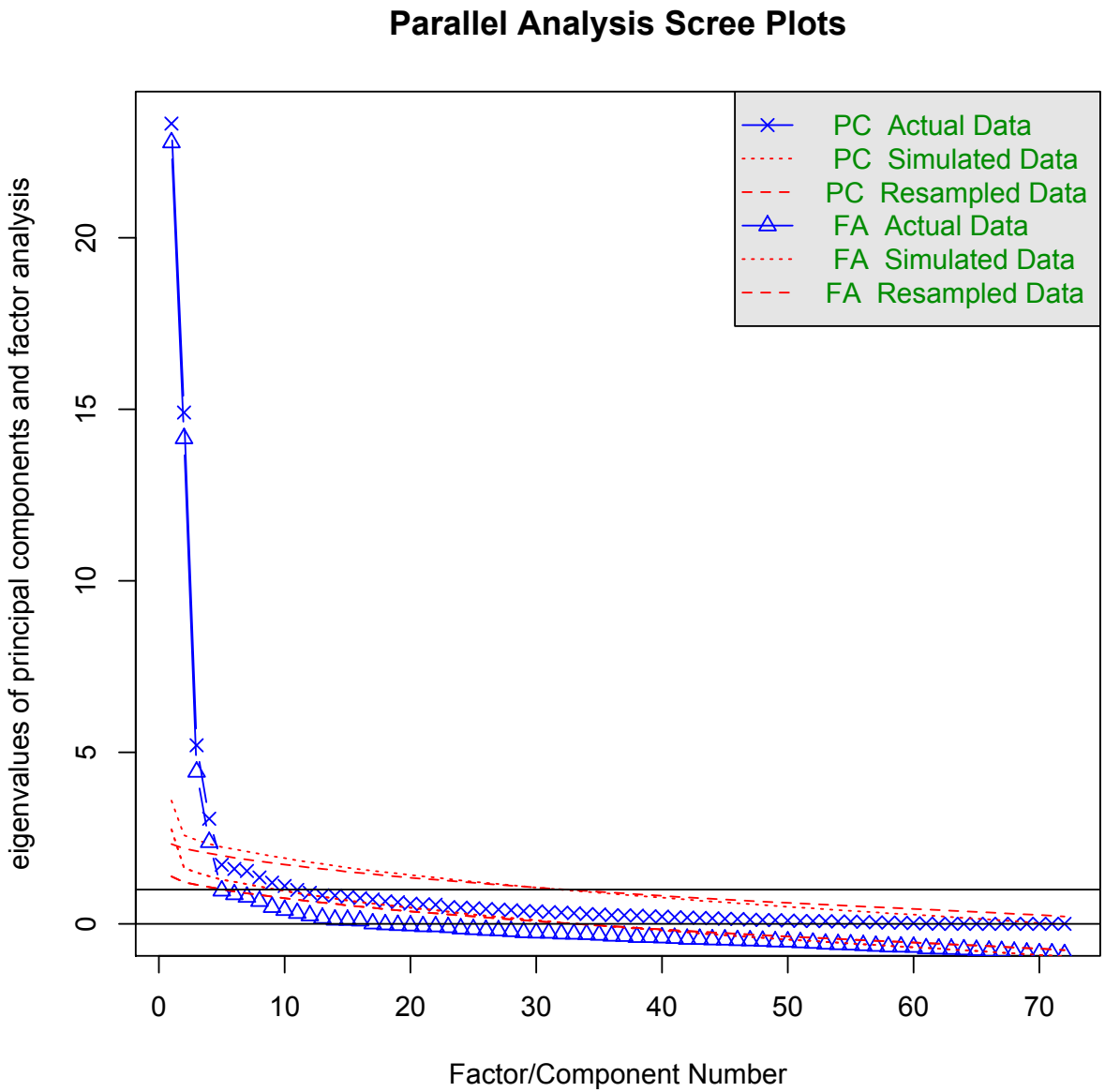
R code

```
> fa.parallel(cleaned[2:73])
Parallel analysis suggests that the number of factors =  5
                    and the number of components =  5


 #use polychoric correlations
fa.parallel(cleaned[2:73],cor="poly")

> fa.parallel(cleaned[2:73],cor="poly")
some warnings are issued
The items do not have an equal number of response
              alternatives, global set to FALSE
Parallel analysis suggests that the number of factors =  4
                  and the number of components =  4
Warning message:
In cor.smooth(mat) : Matrix was not positive definite,
              smoothing was done
>
```

Parallel analysis with Pearson correlations



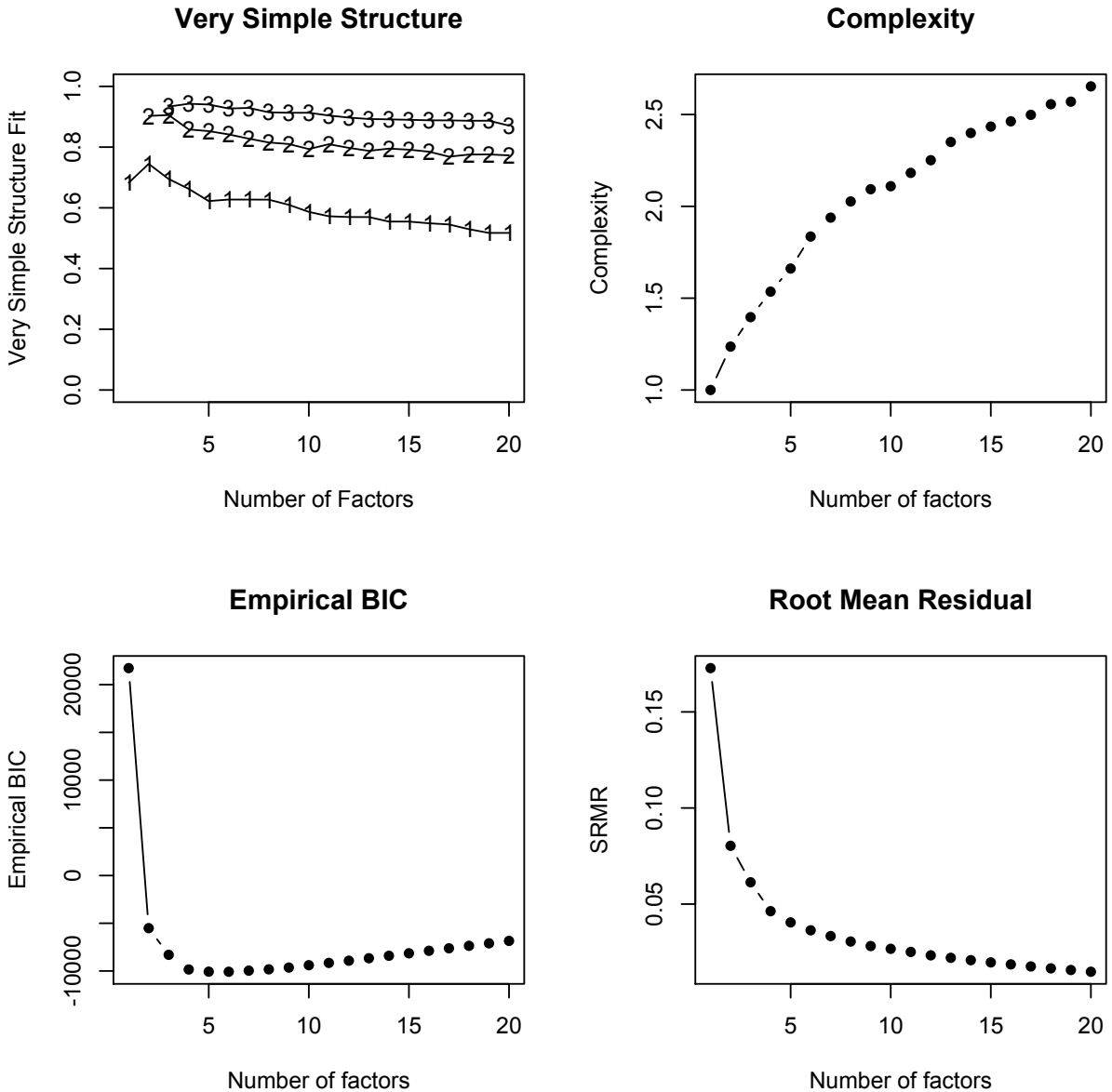
Parallel analysis with polychoric correlations (takes somewhat longer)



How many factors: what does `nfactors` tell us?

```
> nfactors(cleaned[2:73])

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 0.74  with  2  factors
VSS complexity 2 achieves a maximimum of 0.91  with  3  factors
The Velicer MAP achieves a minimum of 0.01  with  9  factors
Empirical BIC achieves a minimum of  -10081.25  with  6  factors
Sample Size adjusted BIC achieves a minimum of  -1843.62  with  11  factors

Statistics by number of factors
   vss1 vss2    map  dof chisq     prob sqresid  fit RMSEA   BIC SABIC complex eChisq  SRMR e
1  0.68 0.00 0.0552 2484  8907  0.0e+00   192.3 0.68 0.115 -4612  3261     1.0  35253 0.173 0
2  0.74 0.90 0.0179 2413  5827 2.9e-282    58.9 0.90 0.087 -7305   343     1.2   7619 0.080 0
3  0.69 0.91 0.0145 2343  4928 8.6e-186    40.3 0.93 0.077 -7823  -397     1.4   4442 0.061 0
4  0.66 0.86 0.0110 2274  4095 1.7e-107    29.0 0.95 0.067 -8281 -1074     1.5   2532 0.046 0
5  0.62 0.85 0.0101 2206  3587  1.9e-69    24.9 0.96 0.061 -8419 -1427     1.7   1933 0.040 0
6  0.63 0.84 0.0094 2139  3274  4.1e-51    22.0 0.96 0.057 -8367 -1588     1.8   1560 0.036 0
7  0.63 0.83 0.0094 2073  3047  2.1e-40    19.7 0.97 0.055 -8235 -1665     1.9   1314 0.033 0
8  0.63 0.82 0.0091 2008  2810  7.0e-30    17.9 0.97 0.052 -8118 -1754     2.0   1100 0.031 0
...
18 0.53 0.78 0.0105 1413  1467  1.5e-01     9.2 0.98 0.032 -6223 -1745     2.6    324 0.017 0
19 0.52 0.78 0.0108 1359  1371  4.0e-01     8.7 0.99 0.030 -6025 -1718     2.6    291 0.016 0
20 0.52 0.77 0.0111 1306  1284  6.6e-01     8.2 0.99 0.028 -5824 -1685     2.7    258 0.015 0
```

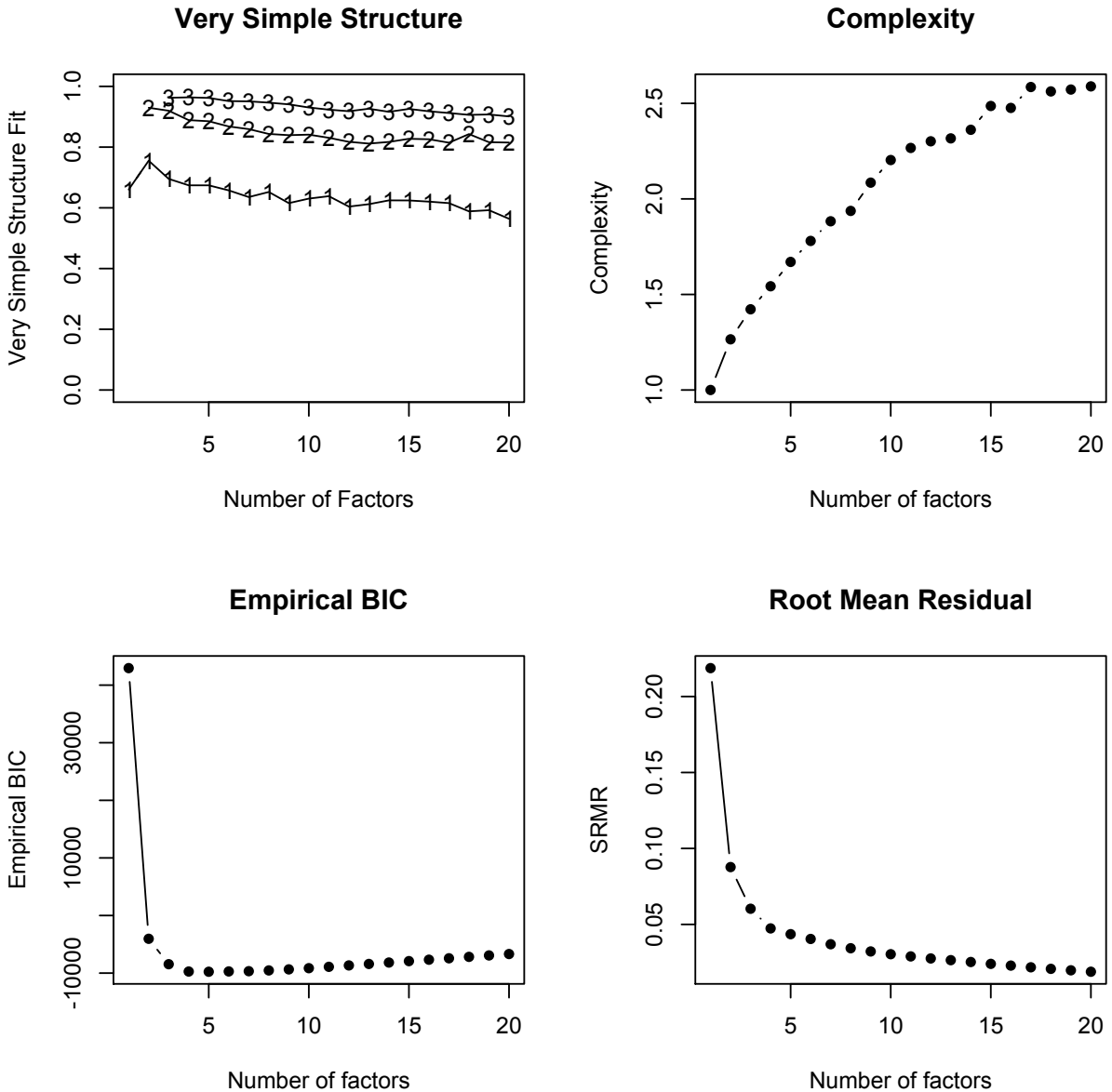
The number of factors from `nfactors`



What if we use polychoric correlations

```
> nfactors(cleaned[2:73],cor="poly")
The items do not have an equal number of response alternatives, global set to FALSE

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 0.76  with  2  factors
VSS complexity 2 achieves a maximimum of 0.93  with  2  factors
The Velicer MAP achieves a minimum of 0.02  with  9  factors
Empirical BIC achieves a minimum of  -9764.2  with  5  factors
Sample Size adjusted BIC achieves a minimum of  39801.49  with  20  factors

Statistics by number of factors
   vss1 vss2   map  dof chisq prob sqresid  fit RMSEA   BIC SABIC complex eChisq  SRMR eCRMS
1  0.66 0.00 0.099 2484 56503    0   280.2 0.66  0.33 42984 50857     1.0  56472 0.219 0.222
2  0.76 0.93 0.027 2413 51500    0    58.1 0.93  0.32 38368 46016     1.3   9088 0.088 0.090
3  0.70 0.92 0.023 2343 50169    0    31.0 0.96  0.32 37417 44843     1.4   4295 0.060 0.063
4  0.67 0.89 0.018 2274 48889    0    21.6 0.97  0.32 36513 43721     1.5   2649 0.047 0.050
5  0.67 0.89 0.018 2206 48230    0    18.6 0.98  0.32 36224 43216     1.7   2242 0.044 0.047
6  0.66 0.87 0.018 2139 47695    0    16.0 0.98  0.33 36053 42833     1.8   1930 0.040 0.044
7  0.64 0.86 0.017 2073 47098    0    13.7 0.98  0.33 35816 42386     1.9   1612 0.037 0.041
8  0.65 0.84 0.017 2008 46595    0    11.8 0.99  0.33 35666 42031     1.9   1391 0.034 0.039
...
19 0.59 0.82 0.022 1359 43055    0     3.5 1.00  0.40 35659 39966     2.6    462 0.020 0.027
20 0.56 0.82 0.022 1306 42770    0     3.1 1.00  0.41 35662 39801     2.6    419 0.019 0.026
Warning message:
In cor.smooth(mat) : Matrix was not positive definite, smoothing was done
```

The number of factors from `nfactors`



EFA of the Motivational State Questionnaire

R code

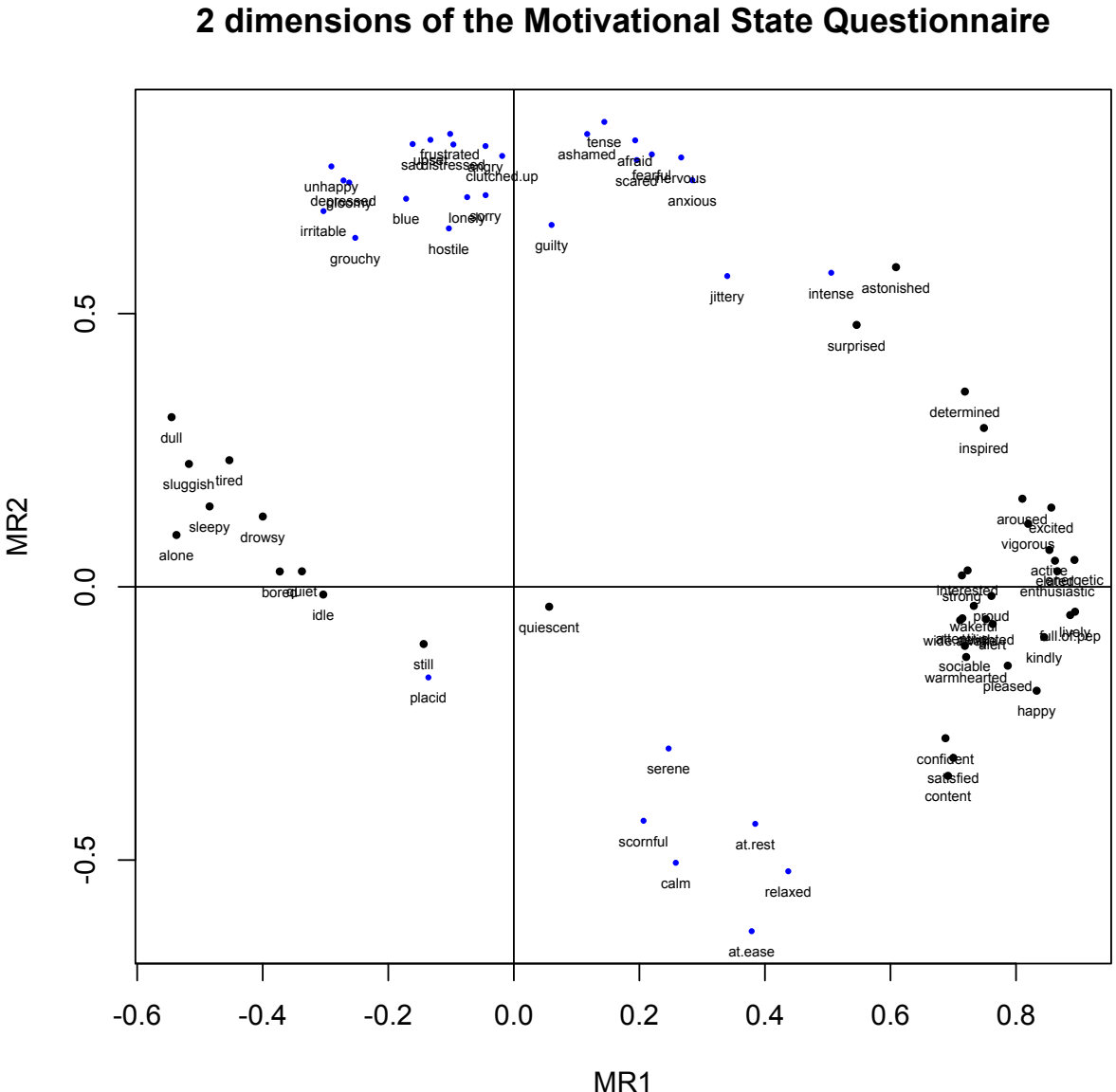
```
f2 <- fa(msq[1:72],2)
summary(f2)
```

```
ummary(f2)

Factor analysis with Call: fa(r = msq[1:72], nfactors = 2)

Test of the hypothesis that 2 factors are sufficient.
The degrees of freedom for the model is 2413  and the objective fun
The number of observations was  3896  with Chi Square =  67730.13

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

Tucker Lewis Index of factoring reliability =  0.637
RMSEA index =  0.083  and the 90 % confidence intervals are  0.083
BIC =  47780.16
 With factor correlations of
      MR1   MR2
MR1  1.00 -0.13
MR2 -0.13  1.00
```

Show the factors, sorted by factor loadings

```
> print(f2,sort=TRUE)
Factor Analysis using method =  minres
Call: fa(r = msq[1:72], nfactors = 2, cor = "poly")
Standardized loadings (pattern matrix) based upon correlation matrix
             item   MR1   MR2    h2   u2 com
lively         20  0.89 -0.05 0.811 0.19 1.0
energetic      55  0.89  0.05 0.789 0.21 1.0
full.of.pep    18  0.89 -0.05 0.800 0.20 1.0
...
sluggish        5 -0.52  0.22 0.348 0.65 1.4
sleepy         59 -0.48  0.15 0.274 0.73 1.2
tired          28 -0.45  0.23 0.285 0.71 1.5
drowsy         51 -0.40  0.13 0.189 0.81 1.2
...
tense          69  0.14  0.85 0.714 0.29 1.1
frustrated     65 -0.10  0.83 0.718 0.28 1.0
ashamed        70  0.12  0.83 0.676 0.32 1.0
upset          48 -0.13  0.82 0.714 0.29 1.1
...
relaxed         8  0.44 -0.52 0.519 0.48 1.9
calm           50  0.26 -0.50 0.354 0.65 1.5
at.rest        26  0.38 -0.43 0.378 0.62 2.0
...

                       MR1   MR2
SS loadings          21.07 17.15
Proportion Var        0.29  0.24
Cumulative Var        0.29  0.53
Proportion Explained  0.55  0.45
Cumulative Proportion 0.55  1.00

 With factor correlations of
     MR1   MR2
```

fa.plot(f2,labels=colnames(msq[1:72]),cex=.5,title="2 dimensions of the Motivational State Questionnaire")

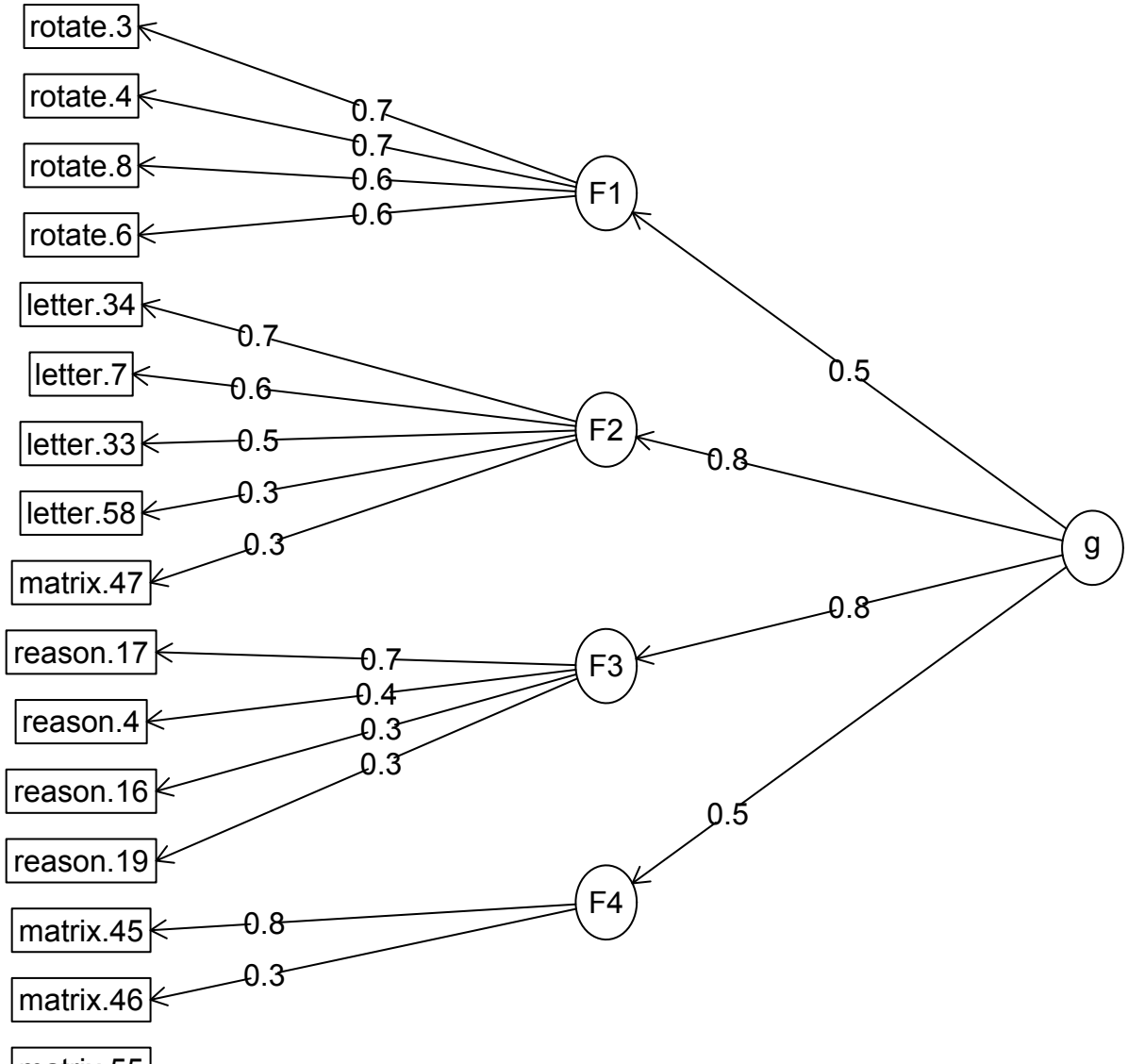


Convert to and sort polar coordinates `round(polar(f2),2)`

	Var	theta21	vecl21
strong	12	1.68	0.51
enthusiastic	14	1.89	0.75
...			
anxious	71	69.09	0.63
nervous	45	71.26	0.69
...			
angry	44	93.20	0.65
sorry	58	93.59	0.52
...			
sad	16	101.25	0.68
blue	10	103.59	0.53
...			
drowsy	51	162.17	0.18
sleepy	59	163.11	0.26
...			
relaxed	8	310.03	0.46
at.rest	26	311.56	0.34
..			
happy	61	347.14	0.73
pleased	60	349.61	0.64
...			
alert	52	354.89	0.59
wide.awake	64	355.09	0.51

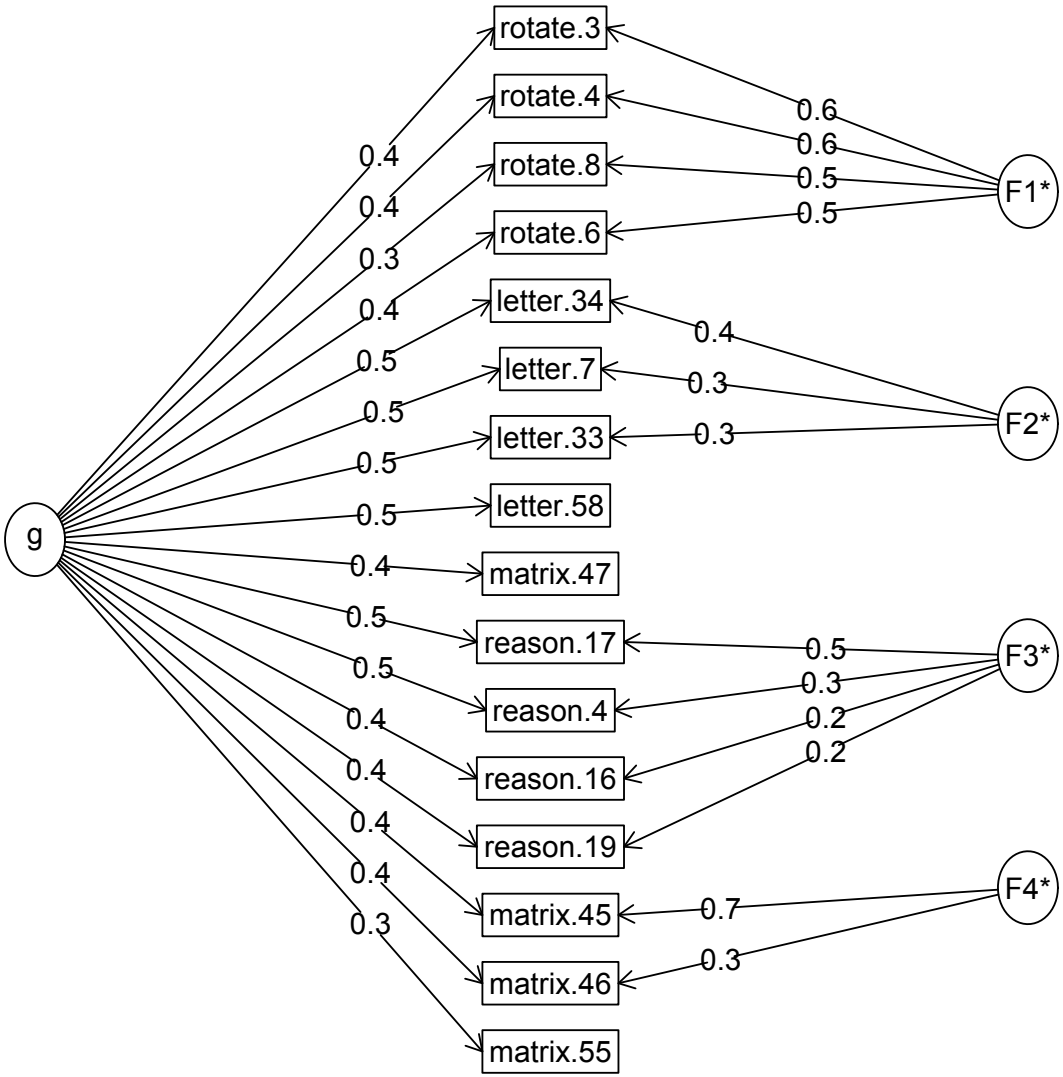
16 ability items from the International Cognitive Ability Resource

general abilty and 4 subfactors of ICAR data



Schmid Leiman transformation of 16 ability items from ICAR

general abilty and 4 subfactors of ICAR data



More types of reliability

1. α is a hodgepodge ratio of general factor and group factor reliability
2. ω_h (omega hierarchical) is an estimate of the general factor variance of a test
3. ω_t (omega total) is an estimate of the total reliable variance of a test
4. When do we use these?
   - When estimating how much of a test measures one thing. $omega_h$
   - When estimating what is the total reliable variance in a test (when adjusting for test reliability in an SEM context)

ω_h and ω_t reliabilities

R code

```
om <- omega(ability, nfactors=4)
```

```
om
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 0.66
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
```

ω continued

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

general/max  2.3   max/min =   3.17
mean percent general =  0.58    with sd =  0.2 and cv of  0.35
Explained Common Variance of the general factor =  0.53

The degrees of freedom are 62  and the fit is  0.05
The number of observations was  1525  with Chi Square =  70.19  with prob <  0.22
The root mean square of the residuals is  0.01
The df corrected root mean square of the residuals is  0.02
RMSEA index =  0.009  and the 90 % confidence intervals are  0 0.014
BIC =  -384.25

Compare this with the adequacy of just a general factor and no group factors
The degrees of freedom for just the general factor are 104  and the fit is  0.78
The number of observations was  1525  with Chi Square =  1186.18  with prob <  5e-183
The root mean square of the residuals is  0.09
The df corrected root mean square of the residuals is  0.09

RMSEA index =  0.083  and the 90 % confidence intervals are  0.078 0.085
BIC =  423.88

Measures of factor score adequacy
                                                 g  F1*   F2*   F3*  F4*
Correlation of scores with factors             0.83 0.80  0.53  0.56 0.71
Multiple R square of scores with factors       0.69 0.64  0.28  0.32 0.50
Minimum correlation of factor score estimates 0.37 0.28 -0.44 -0.37 0.00

 Total, General and Subset omega for each subset
                                                 g  F1*  F2*  F3*  F4*
Omega total for total scores and subscales     0.86 0.77 0.69 0.64 0.53
```

Data set from Preacher and Hayes

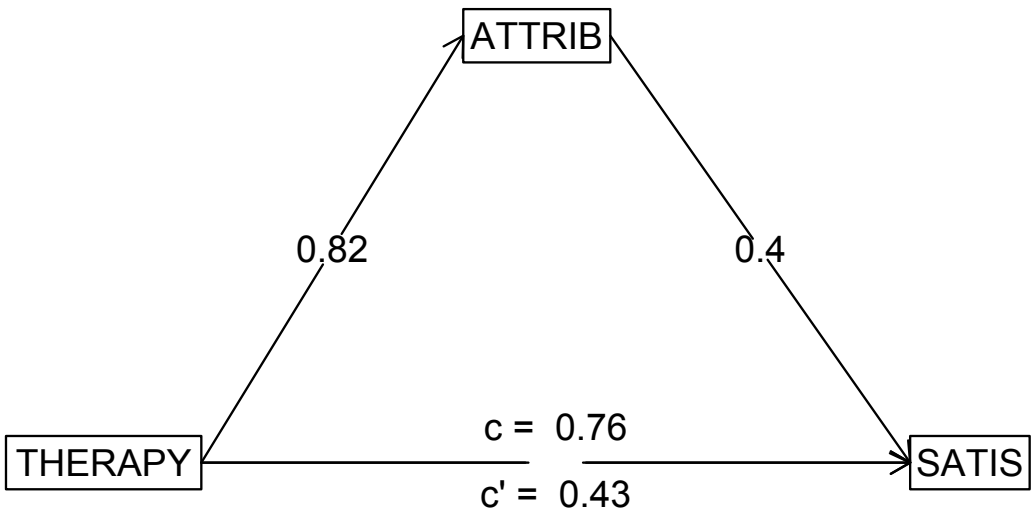
R code

```
# from Preacher and Hayes (2004)
sobel <- structure(list(SATIS = c(-0.59, 1.3, 0.02, 0.01, 0.79, -0.35,
-0.03, 1.75, -0.8, -1.2, -1.27, 0.7, -1.59, 0.68, -0.39, 1.33,
...
"Therapy", "Attributional Positivity"), .Names = c("SATIS", "THERAPY",
"ATTRIB")))
 #n.iter set to 50 (instead of default of 5000) for speed of example
mediate(1,2,3,sobel,n.iter=50)  #The example in Preacher and Hayes
```

```
The DV (Y) was  SATIS . The IV (X) was  THERAPY . The mediating variable(s) =  ATTRIB .

Total Direct effect(c) of  THERAPY  on  SATIS  =  0.76   S.E. =  0.31  t direct =  2.5
Direct effect (c') of  THERAPY  on  SATIS  removing  ATTRIB  =  0.43   S.E. =  0.32  t di
Indirect effect (ab) of  THERAPY  on  SATIS  through  ATTRIB   =  0.33
Mean bootstrapped indirect effect =  0.31  with standard error =  0.16  Lower CI =  0.07
R2 of model =  0.31
 To see the longer output, specify short = FALSE in the print statement

 Full output

 Total effect estimates (c)
        SATIS   se   t   Prob
THERAPY  0.76 0.31 2.5 0.0186

Direct effect estimates     (c')
        SATIS   se    t  Prob
THERAPY  0.43 0.32 1.35 0.190
ATTRIB   0.40 0.18 2.23 0.034

 'a'  effect estimates
       THERAPY  se    t   Prob
ATTRIB    0.82 0.3 2.74 0.0106
```

The Preacher mediation example

Mediation model



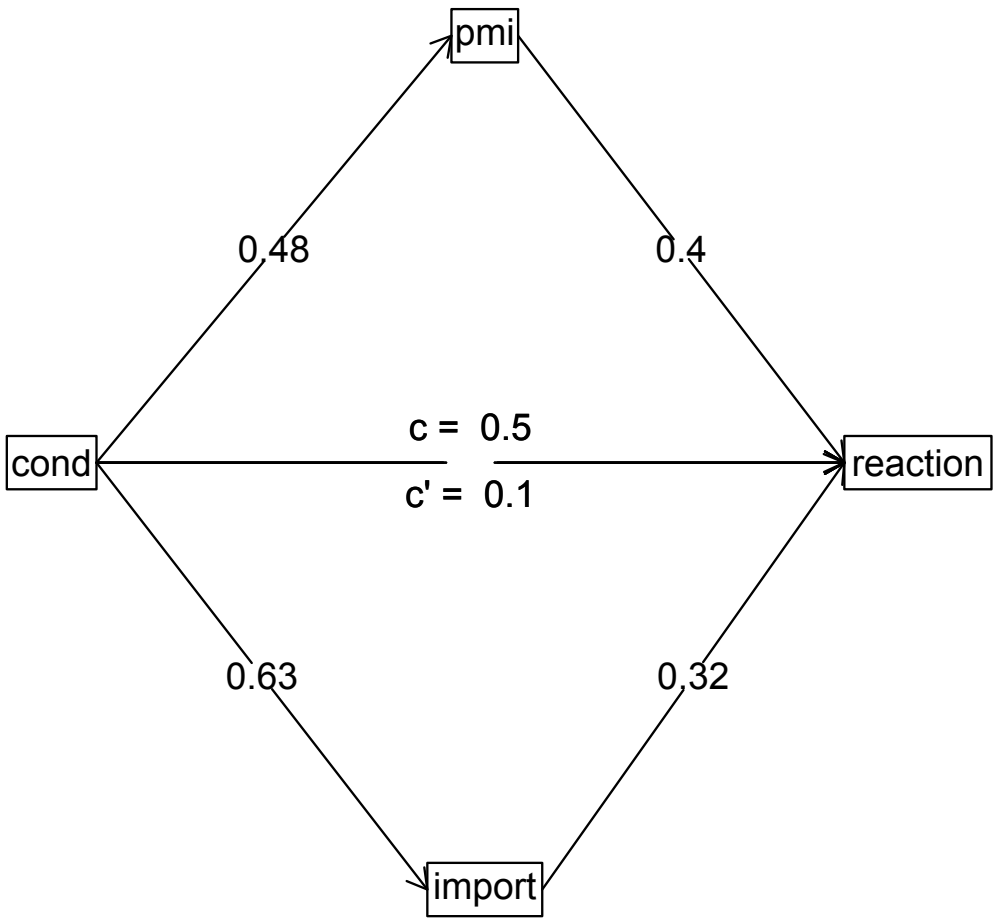
Take the data example from Hayes

R code

```
C.pmi <- structure(c(0.251232840197254, 0.119718779155005, 0.157470345195255,
0.124533519925363, 0.03052112488338, 0.0734039717446355, 0.119718779155005,
...
33.6509729441557), .Dim = c(6L, 6L), .Dimnames = list(c("cond",
"pmi", "import", "reaction", "gender", "age"), c("cond", "pmi",
"import", "reaction", "gender", "age")))

 #n.iter set to 50 (instead of default of 5000) for speed of example
mediate(y="reaction",x = "cond",m=c("pmi","import"),data=C.pmi,n.obs=123,n.iter=50)
```

```
Call: mediate(y = "reaction", x = "cond", m = c("pmi", "import"), data = C.pmi,
    n.obs = 123, n.iter = 50)

The DV (Y) was  reaction . The IV (X) was  cond . The mediating variable(s) =  pmi import

Total Direct effect(c) of  cond  on  reaction  =  0.5   S.E. =  0.28  t direct =  1.79
Direct effect (c') of  cond  on  reaction  removing  pmi import  =  0.1   S.E. =  0.24  t
Indirect effect (ab) of  cond  on  reaction  through  pmi import   =  0.39
Mean bootstrapped indirect effect =  0.7  with standard error =  0.17  Lower CI =  0.39
R2 of model =  0.33
 To see the longer output, specify short = FALSE in the print statement

 Full output

 Total effect estimates (c)
     reaction   se    t   Prob
cond      0.5 0.28 1.79 0.0766

Direct effect estimates     (c')
       reaction   se    t     Prob
cond       0.10 0.24 0.43 6.66e-01
```

The Hayes example mediation



The "New" Psychometrics

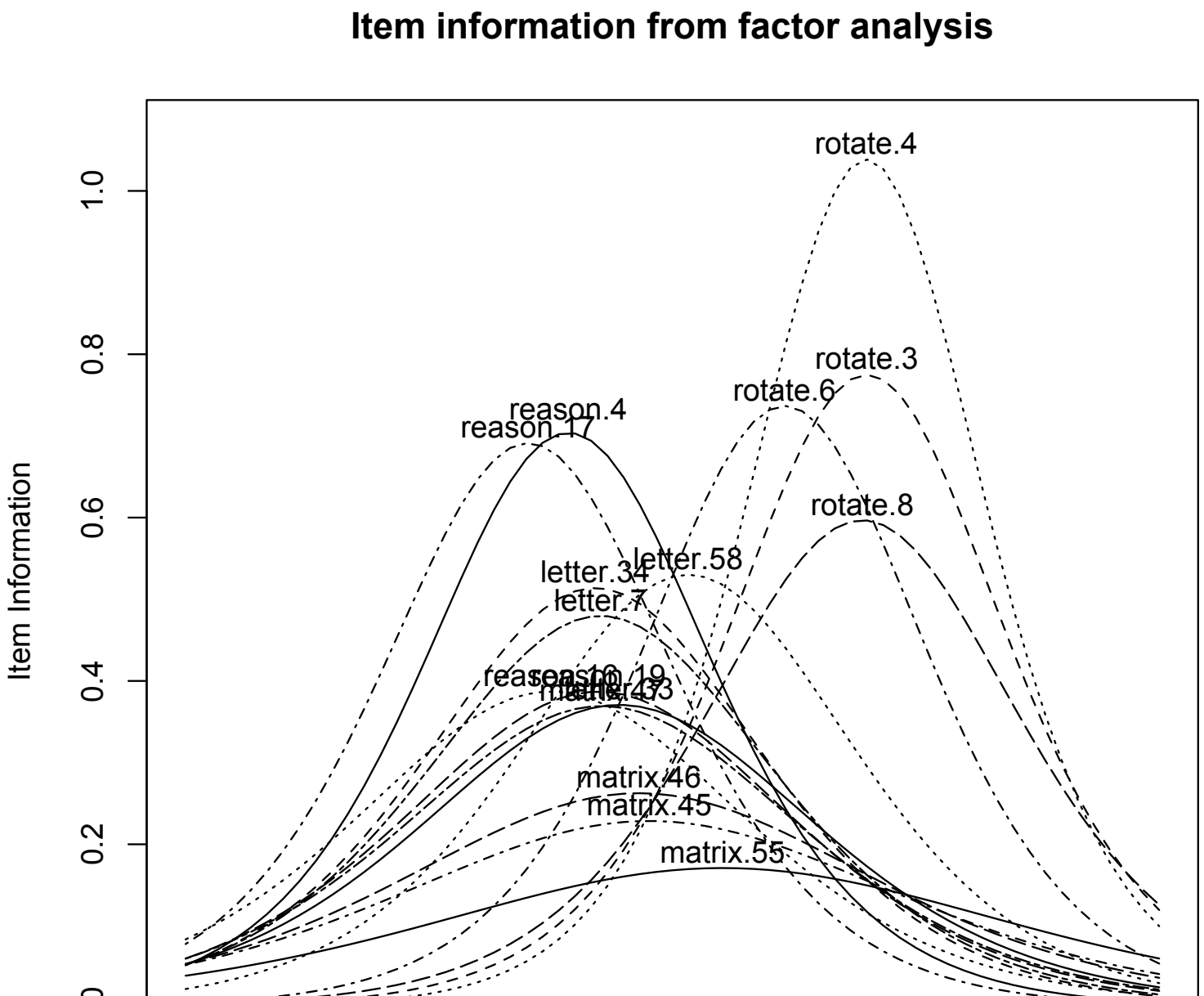
1. Classical Test theory examines responses assuming items are equivalent, or at least congeneric equivalent
2. Item Response Theory models item difficulty as well as item discrimination
3. Although seemingly very different models, factor analysis of categorical items (using tetrachoric or polychoric correlations is equivalent to IRT 2 PL models.
4. Rasch model is just a 1 PL model where items differ in difficulty, but not discrimination.
5. 2PL has difficultly and discrimination estimated from factor analysis of tetrachoric/polychoric items.

R code

```
f1 <- irt.fa(ability)
```

```
 f1 <- irt.fa(ability)

> f1
Item Response Analysis using Factor Analysis

Call: irt.fa(x = ability)
Item Response Analysis using Factor Analysis

 Summary information by factor and item
 Factor =  1
                -3   -2   -1    0    1    2     3
reason.4      0.05 0.24 0.64 0.53 0.16 0.03  0.01
reason.16     0.08 0.22 0.38 0.31 0.14 0.05  0.01
reason.17     0.08 0.33 0.69 0.42 0.11 0.02  0.00
reason.19     0.06 0.17 0.35 0.36 0.19 0.07  0.02
letter.7      0.05 0.18 0.41 0.44 0.20 0.06  0.02
letter.33     0.05 0.15 0.31 0.36 0.20 0.08  0.02
letter.34     0.05 0.19 0.45 0.46 0.20 0.06  0.01
letter.58     0.02 0.09 0.30 0.53 0.35 0.12  0.03
matrix.45     0.05 0.11 0.19 0.23 0.17 0.09  0.04
...
Test Info     0.67 2.11 4.73 5.83 5.28 2.55  0.69
SEM           1.22 0.69 0.46 0.41 0.44 0.63  1.20
```

FA solution with tetrachoric correlations



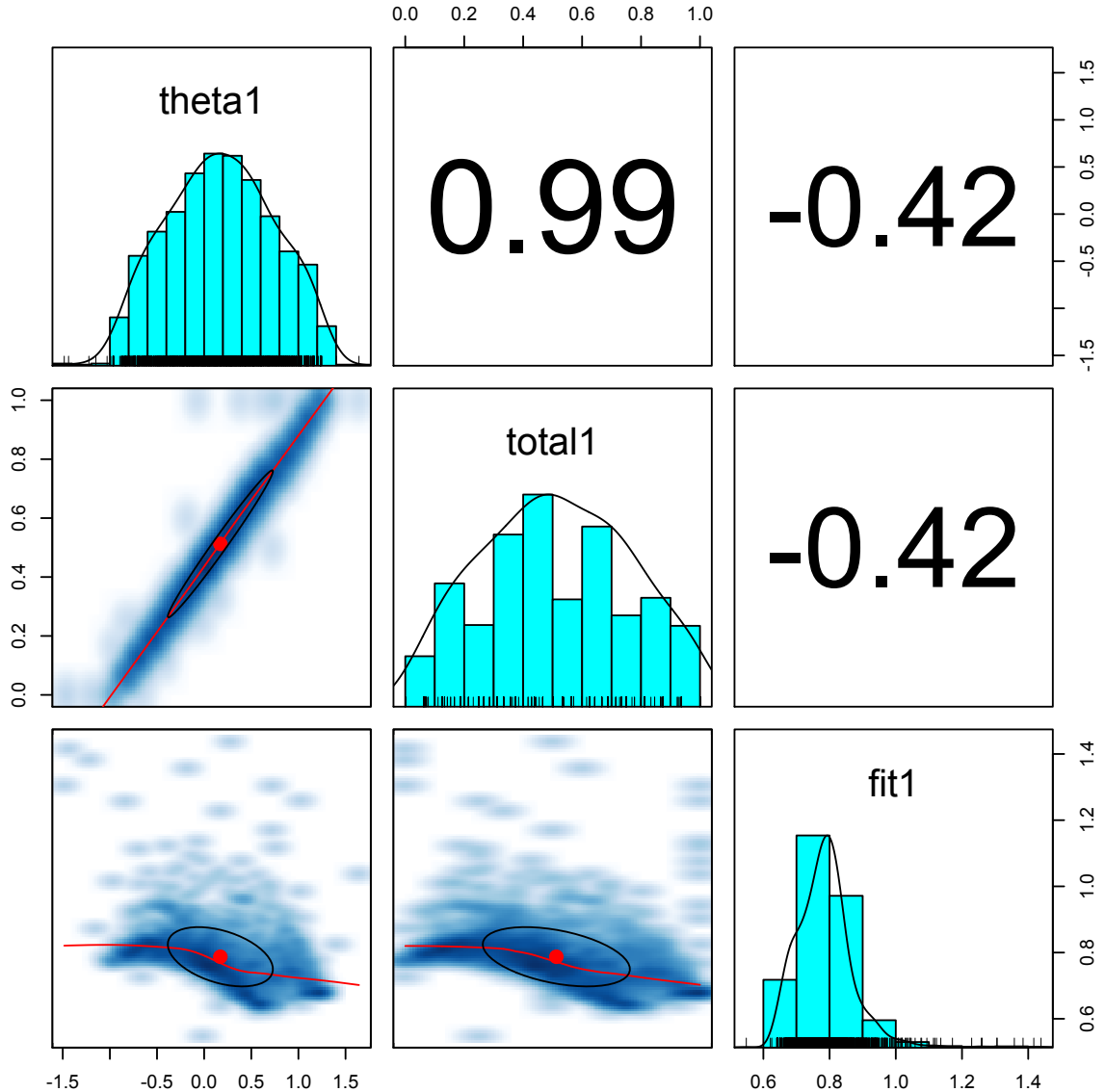
IRT based scoring and Classical Test Theory based scoring

1. CTT and IRT based scores correlate almost perfectly without missing data
2. With lots of missing data, and different items for different people, IRT based scores provide more subtle distinctions.
3. `scoreIrt.2pl` and `scoreIrt.1pl` will do IRT based scores.
4. By default, will find the irt based parameters and then do the scoring.

R code

```
ability.irt <- irt.fa(ability)
ability.scores <- scoreIrt(ability.irt,ability)
 pairs.panels(ability.scores,smoother=TRUE)
```

CTT and IRT based scores are almost identical



Multilevel reliability

1. Classic reliability measures assess the variance of between person differences compared to error of the measurement.

$$\rho_{xx} = \frac{1 - \sigma_e^2}{\sigma_x^2} \tag{1}$$

2. Multilevel reliability is a series of generallizability coeffiicients, generalizing over items, over time, time x items,

$$R_{kF} = \frac{\sigma_{id}^2 + (\sigma_{idxitems}^2/m)}{\sigma_{id}^2 + (\sigma_{idxitems}^2/m) + \sigma_{error}^2/(km)} \tag{2}$$

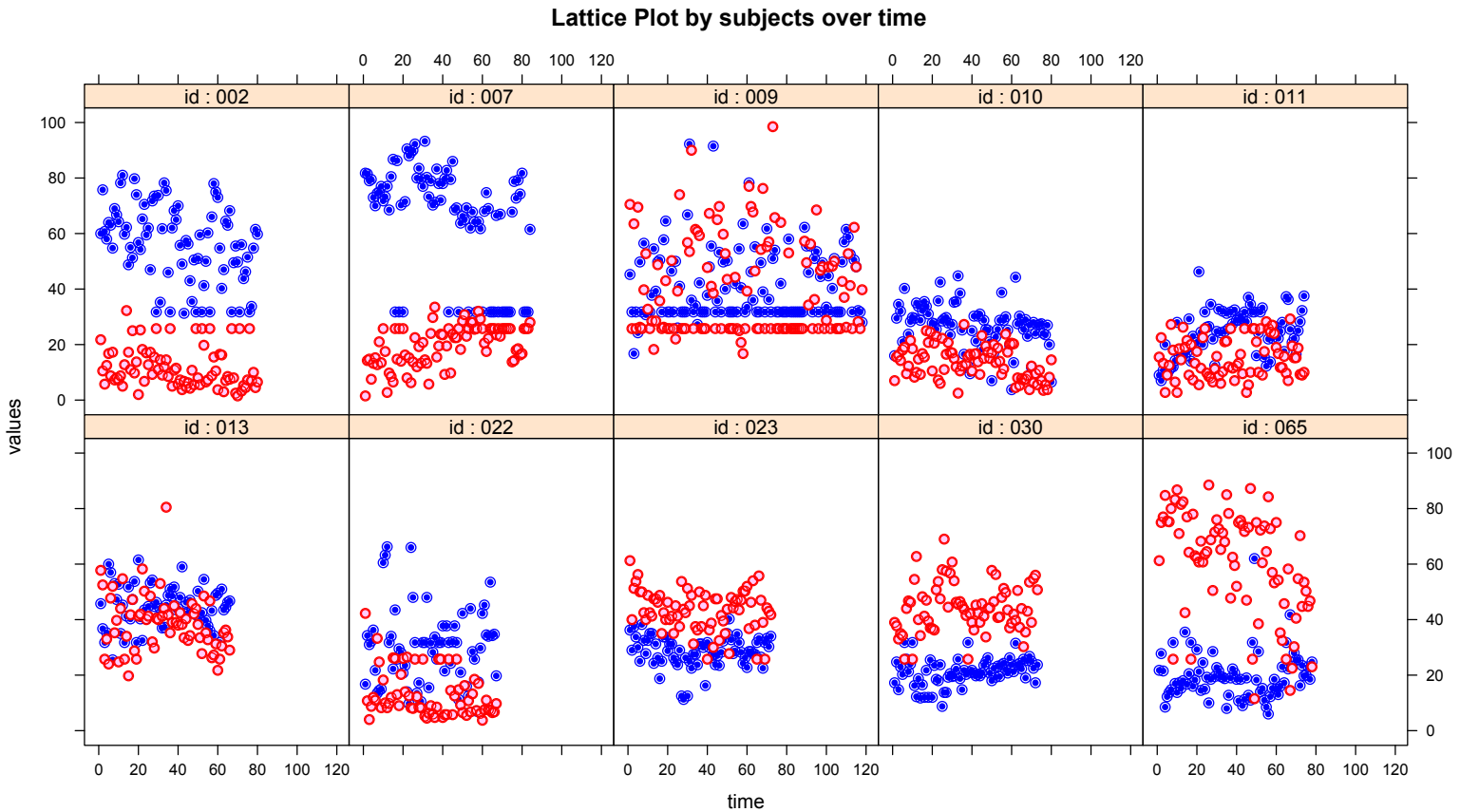
   From Equation 6 (Shrout & Lane, 2012, p 310). See Shrout & Lane (2012) for five other generalizability formula.
3. Implemented in *psych* as `mlr` or `multilevel.reliability`
4. Also simulations using `sim.multi`
5. I show the data from Fisher (2015) who reports 10 subjects measured over 60 (or more) days on 28 affect items.
6. (Download the R data files, minor rearrangement and reliablity measurements.)

Table: The `multilevel.reliablity` function estimates of the generalizability coefficients for the positively and negatively valenced items from Fisher (2015). RkF is the reliability of average of all ratings across all items and times (Fixed time effects), R1R is the generalizability of a single time point across all items (Random time effects), RkR is the generalizability of average time points across all items (Random time effects), Rc is the generalizability of change (fixed time points, fixed items), RkRn is the generalizability of between person differences averaged over time (time nested within people) and Rcn is the generalizability of within person variations averaged over items (time nested within people).

Multilevel reliability estimates

Coefficient	Positive items	Negative items
RkF	1.00	1.00
R1R	0.80	0.77
RkR	1.00	1.00
Rc	0.72	0.71
RkRn	1.00	1.00
Rcn	0.64	0.59

Assessing reliability of within subject differences in affect. Data from Fisher (2015)



psych includes some very old ideas

1. Schmid-Leiman (Schmid & Leiman, 1957) transformations from correlated factor stuctures to higher order structures.
2. Dwyer extension (Dwyer, 1937; Mosier, 1938; Horn, 1973) to extend a factor solution to more variables.
3. This can be used to extend other variables into a factor space, or to relate two domains to each other.

Extend a data set into another

First, create the data set

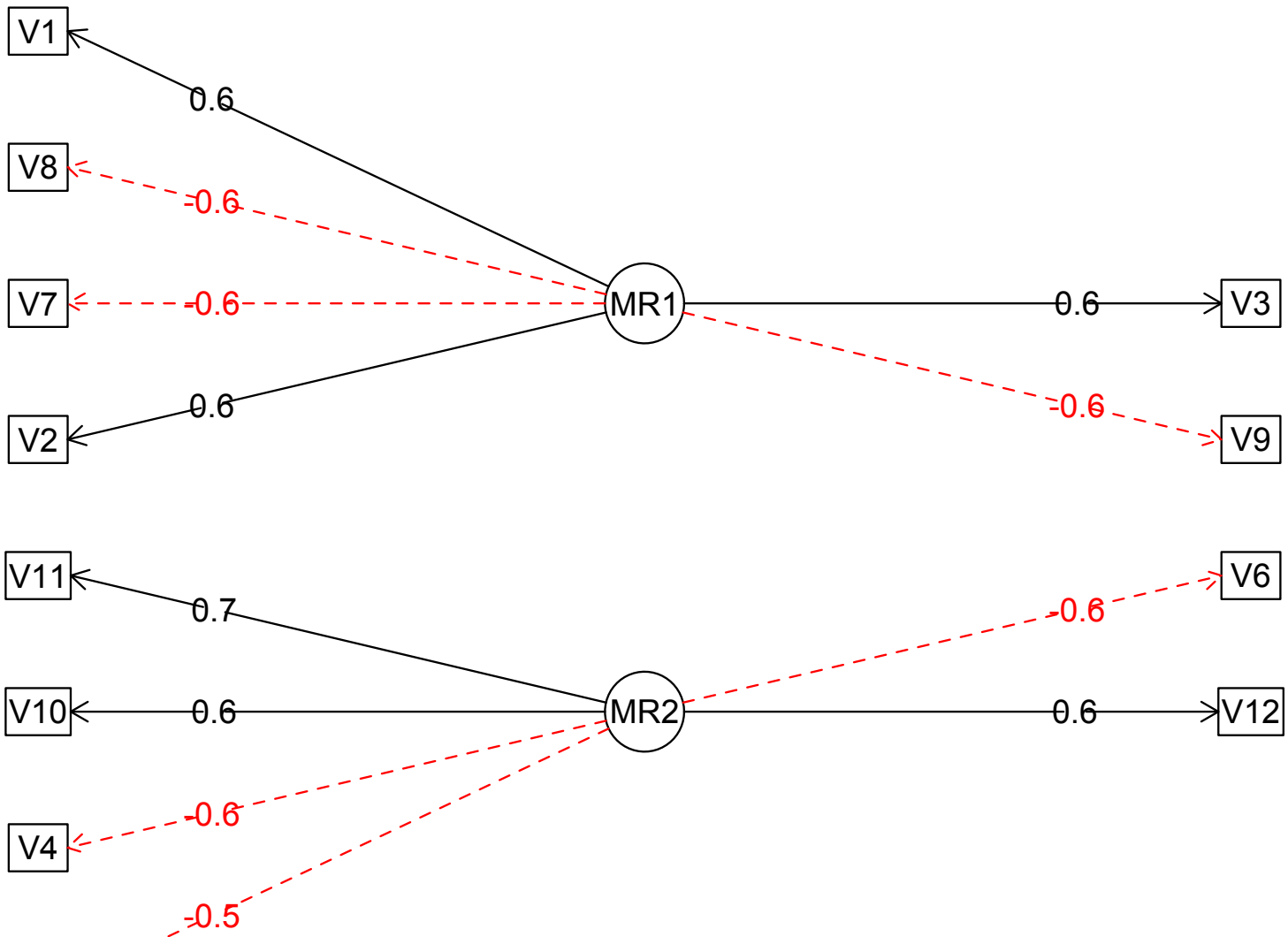
R code

```
set.seed(42)
 d <- sim.item(12)     #two orthogonal factors
R <- cor(d)
Ro <- R[c(1,2,4,5,7,8,10,11),c(1,2,4,5,7,8,10,11)]
 Roe <- R[c(1,2,4,5,7,8,10,11),c(3,6,9,12)]
 fo <- fa(Ro,2)
 fe <- fa.extension(Roe,fo)
 fa.diagram(fo,fe=fe)
```

```
fe

Call: fa.extension(Roe = Roe, fo = fo)
Standardized loadings (pattern matrix) based upon correlation matrix
      MR1   MR2   h2   u2
V3   0.63 -0.02 0.39 0.61
V6   0.04 -0.61 0.37 0.63
V9  -0.61  0.01 0.38 0.62
V12 -0.06  0.58 0.33 0.67

                       MR1  MR2
SS loadings           0.77 0.69
Proportion Var        0.19 0.17
Cumulative Var        0.19 0.37
Proportion Explained  0.53 0.47
Cumulative Proportion 0.53 1.00
     MR1  MR2
```

Factor extension

Factor analysis and extension

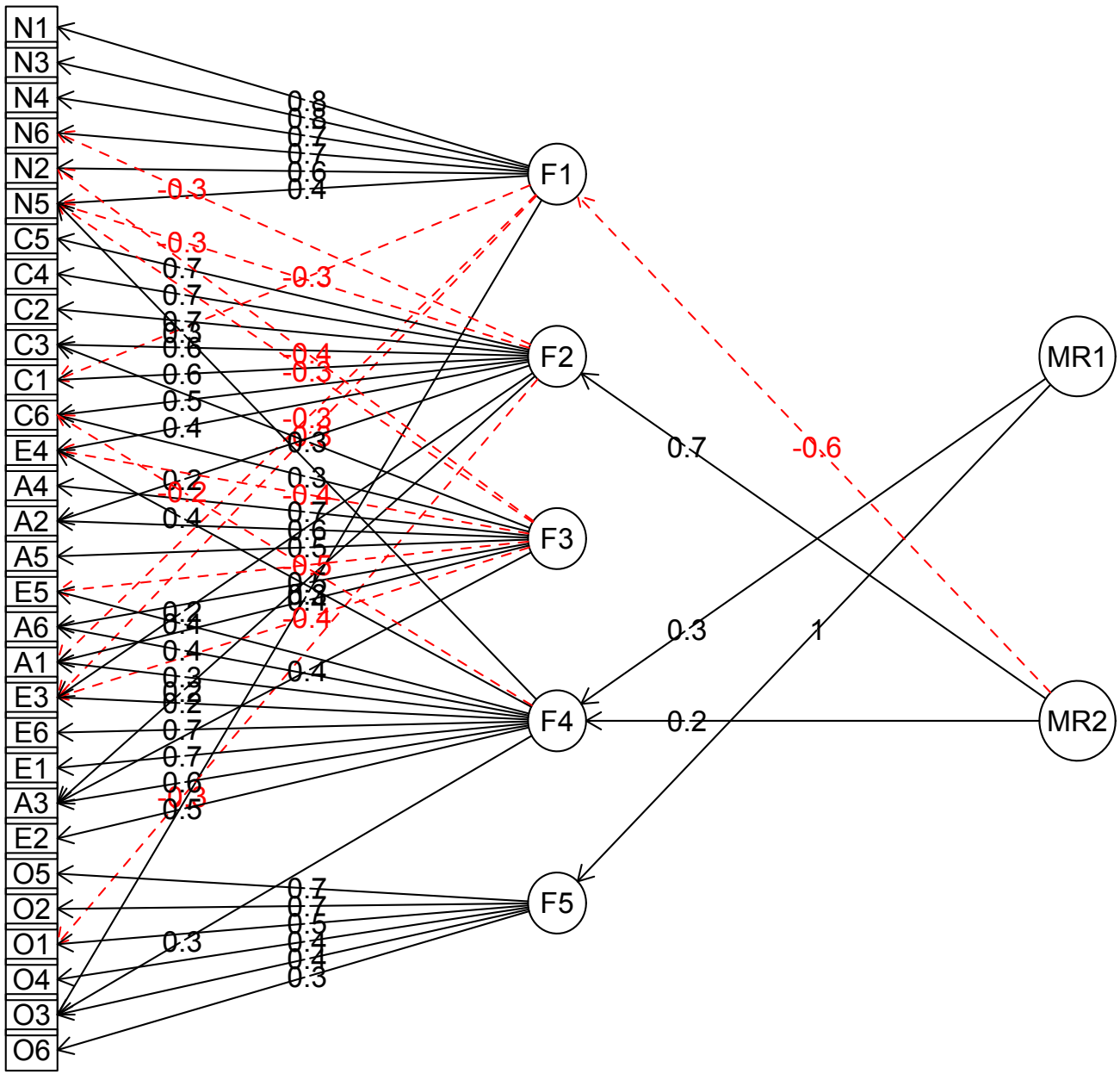


Hiearchical factor analysis

R code

```
neo52 <- fa.multi(neo,5,2)
fa.multi.diagram(neo52)
```

Hierarchical (multilevel) Structure



ESEM can be thought of as factor extension from A to B and B to A

1. If we have two sets of variables that show factor structures within each set
2. And then link the factor structures.
3. Tjhis can be done in SEM, but here show how to do exploratory SEM
4. We make up a toy data set

R code

```
#make up a sem like problem using sim.structure
fx <-matrix(c( .9,.8,.6,rep(0,4),.6,.8,-.7),ncol=2)
 fy <- matrix(c(.6,.5,.4),ncol=1)
 rownames(fx) <- c("V","Q","A","nach","Anx")
 rownames(fy)<- c("gpa","Pre","MA")
Phi <-matrix( c(1,0,.7,.0,1,.7,.7,.7,1),ncol=3)
 gre.gpa <- sim.structural(fx,Phi,fy)
print(gre.gpa)
```

```
Call: sim.structural(fx = fx, Phi = Phi, fy = fy)

 $model (Population correlation matrix)
        V    Q     A  nach   Anx   gpa   Pre    MA
V    1.00 0.72  0.54  0.00  0.00  0.38  0.32  0.25
Q    0.72 1.00  0.48  0.00  0.00  0.34  0.28  0.22
A    0.54 0.48  1.00  0.48 -0.42  0.50  0.42  0.34
nach 0.00 0.00  0.48  1.00 -0.56  0.34  0.28  0.22
Anx  0.00 0.00 -0.42 -0.56  1.00 -0.29 -0.24 -0.20
gpa  0.38 0.34  0.50  0.34 -0.29  1.00  0.30  0.24
Pre  0.32 0.28  0.42  0.28 -0.24  0.30  1.00  0.20
MA   0.25 0.22  0.34  0.22 -0.20  0.24  0.20  1.00

$reliability (population reliability)
   V    Q    A nach  Anx  gpa  Pre   MA
0.81 0.64 0.72 0.64 0.49 0.36 0.25 0.16
```

Exploratory Structural Equation Modling

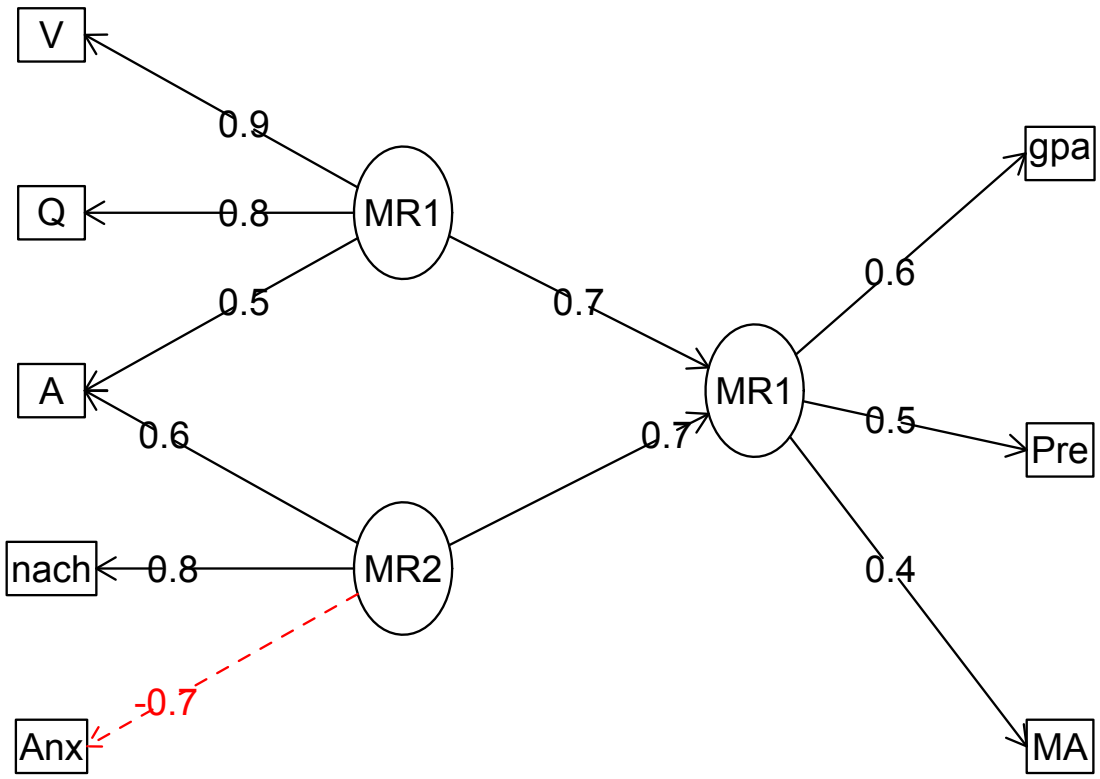
R code

```
example <- esem(gre.gpa$model,varsX=1:5,varsY=6:8,nfX=2,nfY=1,
n.obs=1000,plot=FALSE)
```

```
> example
Exploratory Structural Equation Modeling  Analysis using method =  minres
Call: esem(r = gre.gpa$model, varsX = 1:5, varsY = 6:8, nfX = 2, nfY = 1,
    n.obs = 1000, plot = FALSE)

For the 'X' set:
       MR1   MR2
V     0.91 -0.06
Q     0.81 -0.05
A     0.53  0.57
nach -0.10  0.81
Anx   0.08 -0.71

For the 'Y' set:
    MR1
gpa 0.6
Pre 0.5
MA  0.4

Correlations between the X and Y sets.
     X1   X2   Y1
X1 1.00 0.19 0.68
X2 0.19 1.00 0.67
Y1 0.68 0.67 1.00

The degrees of freedom for the null model are  56  and the empirical chi square  functior
The degrees of freedom for the model are 7  and the empirical chi square function was  21
  with prob <  0.0027
```

ESEM of our toy problem

Exploratory Structural Model



UseR! and the *psych*

1. Not shown
   - Basic IRT analysis using `irt.fa`
   - Statistics for each group using `describeBy`, `statsBy` and `faBy`
   - Circular statistics for diurnal mood data using `cosinor`, `circadian.mean`, etc.
   - Simulation of multiple types of data structures using `sim`, `sim.items`, ...
   - Convenient LaTeX output functions
2. Tutorials for R and the *psych* package at `http://personality-project.org/r`
   - Guide to *psych* package `http://personality-project.org/r/psych`
   - Vignette for *psych* package at http://personality-project.org/r/psych/vignettes/overview.pdf
3. Slides from today at: `http://personality-project.org/r/tutorials/swpa/swpa.17.pdf`

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.

Fisher, A. J. (2015). Toward a dynamic model of psychological assessment: Implications for personalized care. *Journal of Consulting and Clinical Psychology*, *83*(4), 825 – 836.

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

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

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

Shrout, P. & Lane, S. P. (2012). Psychometrics. In *Handbook of research methods for studying daily life*. Guilford Press.