Solutions to Lab 6

Multivariate Statistics with R

This week's lab is the first in a five part block on latent variable models (factor analysis, path analysis, and structural equation modeling). For lots of useful info and links, visit Tim's Multivariate Stats Course page. And now, let's get going!

Task 1: Find and load the `bfi` dataset (in the `psych` package).

```
### ANSWER ###

library(psych) # bfi is autoloaded
```

Question 1.1: what columns contain the Big-Five Inventory data?

```
### ANSWER ###
```

Looking at `?bfi`, we can see that the BFI items are stored in columns `1:25`: `A1-5`, `C1-5`, `E1-5`, `N1-5`, and `O1-5` (for *Agreeableness*, *Conscientiousness*, *Extraversion*, *Neuroticism*, and *Openness*).

Task 2: Find a package in `R` that does parallel analysis.

Question 2.1: What is it's name?

```
### ANSWER ###
```

A cursory [*insert your favourite websearch engine*] search suggests package `paran`. Check out its documentation.

Question 2.2: What is the name of the function?

```
### ANSWER ###
```

That would be the **only** function in the package, `paran()`.

Task 3: Read the function documentation (help file).

```
### ANSWER ###

# let's first install and load the package
install.packages("paran")
library(paran)

?paran
```

Question 3.1: What parameters does this parallel analysis function take?

```
### ANSWER ###
```

The help file lists quite a few. The main ones to focus on are `x`, `iterations`, and `graph`.

Question 3.2: What do they do?

```
### ANSWER ###
```

- `x`: data to be factor analysed (and nothing but).
- `iterations`: number of Monte Carlo datasets to be generated in order to get the chance-expected factor Eigenvalues (as discussed in this week's lecture, parallel analysis compares the empirical factor structure to one expected by chance in random ratasets with similar properties as the one analysed).
- `graph`: Whether or not to display the scree plot (`FALSE` by default).

Task 4: Use the function to determine how many factors are in the `bfi` dataset.

```
### ANSWER ###

paran(bfi)
```

Question 4.1: Assuming that didn't work, what went wrong?

`### ANSWER ###`

There are missing data (`NA` values) in the `bfi` dataset and `paran()` cannot handle those.

Question 4.2: Does the parallel analysis function need to be given just the columns you need to analyse?

```
### ANSWER ###

#Yes, otherwise the results will be dodgy
paran(bfi[complete.cases(bfi), ])
```

```
##
## Using eigendecomposition of correlation matrix.
## Computing: 10%  20%  30%  40%  50%  60%  70%  80%  90%  100%
##
##
## Results of Horn's Parallel Analysis for component retention
## 840 iterations, using the mean estimate
##
## ----------------------------------------------------
## Component   Adjusted    Unadjusted    Estimated
##             Eigenvalue  Eigenvalue    Bias
## ----------------------------------------------------
## 1           4.898690    5.107359      0.208668
## 2           2.631306    2.812007      0.180701
## 3           2.041202    2.200654      0.159451
## 4           1.800864    1.942465      0.141601
## 5           1.496158    1.621137      0.124978
## 6           1.148731    1.257755      0.109024
## 7           1.012924    1.106897      0.093973
## ----------------------------------------------------
##
## Adjusted eigenvalues > 1 indicate dimensions to retain.
## (7 components retained)
```

Question 4.3: How many complete cases exist in these personality data?

```
### ANSWER ###

sum(complete.cases(bfi))
```

```
## [1] 2236
```

Task 5: Run the function on the appropriate subset of `bfi`.

```
### ANSWER ###

# first let's get the complete rows and the right columns
df <- bfi[complete.cases(bfi), 1:25]
# now let's run paran()
paran(df)
```

```
##
## Using eigendecomposition of correlation matrix.
## Computing: 10%  20%  30%  40%  50%  60%  70%  80%  90%  100%
##
##
## Results of Horn's Parallel Analysis for component retention
## 750 iterations, using the mean estimate
##
## ----------------------------------------------------
## Component   Adjusted    Unadjusted    Estimated
##             Eigenvalue  Eigenvalue    Bias
## ----------------------------------------------------
## 1           4.874447    5.068516      0.194069
## 2           2.595990    2.762479      0.166488
## 3           2.007285    2.152622      0.145337
## 4           1.766495    1.892332      0.125837
## 5           1.408504    1.517532      0.109028
## ----------------------------------------------------
##
## Adjusted eigenvalues > 1 indicate dimensions to retain.
## (5 components retained)
```

Question 5.1: How many factors exist in these personality data?

```
### ANSWER ###
```

Five, as in The Big *Five* Inventory.

Question 5.2: What is a scree plot and how do you plot it with this function?

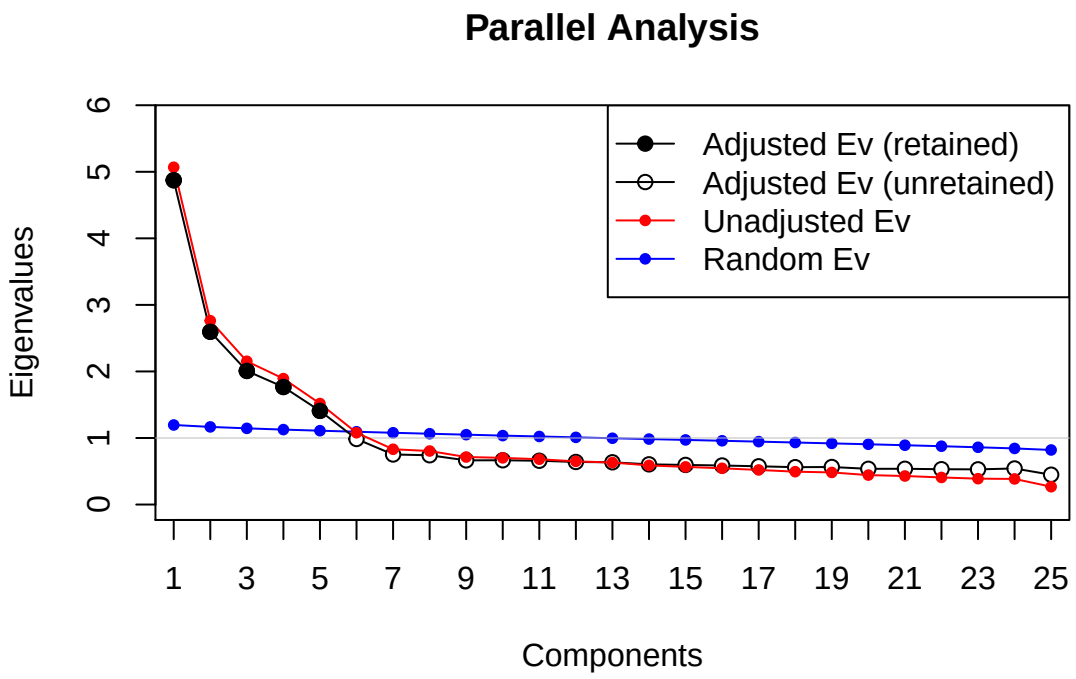
```
### ANSWER ###
```

A scree (a term borrowed from geology) plot shows the Eigenvalues of all factors in the data from the lasrgest to the smallest. (Before the analytical decision to extract a certain number of factors is made, there are always as many factors as there are columns/variables.)

```
paran(df, graph = TRUE)
```

```
##
```

```
## Using eigendecomposition of correlation matrix.
## Computing: 10%  20%  30%  40%  50%  60%  70%  80%  90%  100%
##
##
## Results of Horn's Parallel Analysis for component retention
## 750 iterations, using the mean estimate
##
## ----------------------------------------------------
## Component   Adjusted    Unadjusted    Estimated
##             Eigenvalue  Eigenvalue    Bias
## ----------------------------------------------------
## 1           4.873210    5.068516      0.195305
## 2           2.595820    2.762479      0.166658
## 3           2.007741    2.152622      0.144881
## 4           1.766289    1.892332      0.126043
## 5           1.407958    1.517532      0.109574
## ----------------------------------------------------
##
## Adjusted eigenvalues > 1 indicate dimensions to retain.
## (5 components retained)
```



Task 6: Find R's built in factor analysis function.

Question 6.1: Which one is it?

```
### ANSWER ###

factanal()
```

Question 6.2: What parameters does this function need?

`### ANSWER ###`

Again, looking at the function documentation (`?factanal`), we see that at the very least, it needs the data to be factor analysed (`x`) and the number of factors to extract (`factors`).

Question 6.3: What are its options? Discuss.

`### ANSWER ###`

You can specify if you want the function to give you factor `scores` and if so, which type (regression scores or Bartlett scores) and also what kind of `rotation` you would like the function to perform on the extracted factors (see lecture notes on what a factor rotation is).

Task 7: Run an fa, extracting the predicted number of factors from `paran()`.

```
### ANSWER ###

fa <- factanal(df, factors = 5)
print(fa)
```

```
##
## Call:
## factanal(x = df, factors = 5)
##
## Uniquenesses:
##    A1    A2    A3    A4    A5    C1    C2    C3    C4    C5    E1    E2
## 0.843 0.602 0.485 0.694 0.525 0.669 0.579 0.675 0.516 0.561 0.640 0.454
##    E3    E4    E5    N1    N2    N3    N4    N5    O1    O2    O3    O4
## 0.543 0.461 0.585 0.277 0.341 0.474 0.502 0.657 0.676 0.725 0.516 0.758
##    O5
## 0.714
##
## Loadings:
##    Factor1 Factor2 Factor3 Factor4 Factor5
## A1                         -0.375
## A2          0.195   0.143   0.579
## A3          0.280   0.113   0.649
## A4          0.172   0.226   0.453  -0.132
## A5 -0.118   0.337           0.581
## C1                  0.528           0.215
## C2                  0.617   0.137   0.125
## C3                  0.556   0.120
## C4  0.222          -0.647
## C5  0.266  -0.193  -0.572
## E1         -0.578          -0.139
## E2  0.227  -0.675  -0.100  -0.157
## E3          0.498           0.326   0.311
## E4 -0.123   0.602           0.390
## E5          0.498   0.314   0.128   0.224
## N1  0.814                  -0.208
## N2  0.783                  -0.203
## N3  0.717
## N4  0.563  -0.374  -0.191
```

```
## N5  0.521  -0.183           0.109  -0.150
## O1           0.176   0.112           0.523
## O2  0.173          -0.115   0.119  -0.467
## O3           0.273           0.149   0.619
## O4  0.211  -0.221           0.130   0.360
## O5                                  -0.524
##
##                Factor1 Factor2 Factor3 Factor4 Factor5
## SS loadings      2.685   2.305   2.011   1.952   1.574
## Proportion Var   0.107   0.092   0.080   0.078   0.063
## Cumulative Var   0.107   0.200   0.280   0.358   0.421
##
## Test of the hypothesis that 5 factors are sufficient.
## The chi square statistic is 1357.5 on 185 degrees of freedom.
## The p-value is 1.88e-177
```

Question 7.1: What does uniqueness mean?

```
### ANSWER ###
```

It is the proportion of variance in the variable that is **NOT** captured by the extracted factors (*e.g.*, uniqueness of .67 means that 67% of the item's variance is not expressed by the combination of the 5 extracted factors).

Question 7.2: Are items fairly unique in general?

```
### ANSWER ###
```

Yes, often well over half of their variance is unique. In fact, only six of the 25 items have uniqueness of less than .5.

Question 7.3: Was what you ran by default oblique or orthogonal?

```
### ANSWER ###
```

If you look at the function documentation you'll see that in the **Usage** section, rotation is set by default to `"varimax"`, which - as you learnt in the lecture - is an orthogonal rotation.

Question 7.4: What is the name of an oblique rotation?

```
### ANSWER ###
```

A quick Bing[1] search will yield several oblique rotation methods but rather, uncharacteristically (and unhelpfully!), the help file doesn't tell you which ones are available in the `factanal()` function. If you dig a bit deeper, the Quick-R page on factor analysis will tell you that:

> The **rotation=** options include "varimax", "promax", and "none".

A good answer then is *promax*.

Task 8: Use the oblique rotation available in `factanal()`.

[1] Other search engines are available...[2]

[2] ...and highly recommended!

```r
### ANSWER ###

fa <- factanal(df, factors = 5, rotation = "promax")
print(fa)
```

```
##
## Call:
## factanal(x = df, factors = 5, rotation = "promax")
##
## Uniquenesses:
##    A1    A2    A3    A4    A5    C1    C2    C3    C4    C5    E1    E2
## 0.843 0.602 0.485 0.694 0.525 0.669 0.579 0.675 0.516 0.561 0.640 0.454
##    E3    E4    E5    N1    N2    N3    N4    N5    O1    O2    O3    O4
## 0.543 0.461 0.585 0.277 0.341 0.474 0.502 0.657 0.676 0.725 0.516 0.758
##    O5
## 0.714
##
## Loadings:
##    Factor1 Factor2 Factor3 Factor4 Factor5
## A1  0.224   0.121          -0.387
## A2                          0.582
## A3          0.181           0.646
## A4                  0.182   0.453  -0.182
## A5 -0.150   0.260           0.558
## C1                  0.549           0.158
## C2  0.124  -0.142   0.658   0.102
## C3                  0.593
## C4                 -0.675
## C5  0.119  -0.120  -0.581           0.108
## E1 -0.131  -0.632   0.149
## E2         -0.715
## E3          0.468           0.263   0.302
## E4          0.605           0.338
## E5  0.222   0.473   0.235           0.195
## N1  0.909   0.174          -0.153
## N2  0.860   0.115          -0.153
## N3  0.682
## N4  0.398  -0.393  -0.124
## N5  0.433  -0.195           0.194  -0.153
## O1          0.118                   0.525
## O2  0.164                   0.188  -0.473
## O3          0.215                   0.629
## O4         -0.299           0.149   0.369
## O5                                  -0.533
##
##                Factor1 Factor2 Factor3 Factor4 Factor5
## SS loadings      2.617   2.293   2.038   1.807   1.576
## Proportion Var   0.105   0.092   0.082   0.072   0.063
## Cumulative Var   0.105   0.196   0.278   0.350   0.413
##
## Factor Correlations:
##         Factor1 Factor2 Factor3 Factor4 Factor5
## Factor1   1.000  0.3698   0.376  0.1253   0.234
## Factor2   0.370  1.0000   0.247 -0.0245  -0.088
```

```
## Factor3   0.376  0.2468   1.000  0.2205   0.198
## Factor4   0.125 -0.0245   0.221  1.0000   0.183
## Factor5   0.234 -0.0880   0.198  0.1826   1.000
##
## Test of the hypothesis that 5 factors are sufficient.
## The chi square statistic is 1357.5 on 185 degrees of freedom.
## The p-value is 1.88e-177
```

Question 8.1: Is the structure "simple" now?

`### ANSWER ###`

Yes, as most items only have one sizeable factor loading.

Question 8.2: What does that mean?

`### ANSWER ###`

It means that the items that contribute to one of the factors, say Factor 5, do not contribute to other factors. However, bear in mind that due to the oblique rotation, Factor 5 is now correlated with other factors (*e.g.*, .234 with Factor 1).

Question 8.3: What are the factors? (Name them based on high loading items)

`### ANSWER ###`

- Factor 1 loads most highly on the *Neuroticism* items.
- Factor 2 loads most highly on the *Extraversion* items.
- Factor 3 loads most highly on the *Conscientiousness* items.
- Factor 4 loads most highly on the *Agreeableness* items.
- Factor 5 loads most highly on the *Openness* items.

Question 8.4: What do the empty cells mean?

`### ANSWER ###`

The print out hides small values ($< .1$).

Task 9: Try and alter how the result prints out. Let's say we want to see only loadings $> .3$ and we want the items sorted by factors that load on them.

Hint: Look for the `print` method in the help file for `factanal()`.

`### ANSWER ###`

The factor analysis object (`class(fa)` returns `[1] "factanal"`) has a special print method (documented under `?loadings`) that supports sorting and hiding small values.

```
### ANSWER ###

# for example
print(fa, cutoff = .3, sort = TRUE)
```

```
##
## Call:
## factanal(x = df, factors = 5, rotation = "promax")
##
## Uniquenesses:
##    A1    A2    A3    A4    A5    C1    C2    C3    C4    C5    E1    E2
## 0.843 0.602 0.485 0.694 0.525 0.669 0.579 0.675 0.516 0.561 0.640 0.454
##    E3    E4    E5    N1    N2    N3    N4    N5    O1    O2    O3    O4
## 0.543 0.461 0.585 0.277 0.341 0.474 0.502 0.657 0.676 0.725 0.516 0.758
##    O5
## 0.714
##
## Loadings:
##    Factor1 Factor2 Factor3 Factor4 Factor5
## N1  0.909
## N2  0.860
## N3  0.682
## E1         -0.632
## E2         -0.715
## E4          0.605           0.338
## C1                  0.549
## C2                  0.658
## C3                  0.593
## C4                 -0.675
## C5                 -0.581
## A2                          0.582
## A3                          0.646
## A5                          0.558
## O1                                  0.525
## O3                                  0.629
## O5                                 -0.533
## A1                         -0.387
## A4                          0.453
## E3          0.468                  0.302
## E5          0.473
## N4  0.398  -0.393
## N5  0.433
## O2                                 -0.473
## O4                                  0.369
##
##                Factor1 Factor2 Factor3 Factor4 Factor5
## SS loadings      2.617   2.293   2.038   1.807   1.576
## Proportion Var   0.105   0.092   0.082   0.072   0.063
## Cumulative Var   0.105   0.196   0.278   0.350   0.413
##
## Factor Correlations:
##         Factor1 Factor2 Factor3 Factor4 Factor5
## Factor1   1.000  0.3698   0.376  0.1253   0.234
## Factor2   0.370  1.0000   0.247 -0.0245  -0.088
## Factor3   0.376  0.2468   1.000  0.2205   0.198
## Factor4   0.125 -0.0245   0.221  1.0000   0.183
## Factor5   0.234 -0.0880   0.198  0.1826   1.000
##
## Test of the hypothesis that 5 factors are sufficient.
```

```
## The chi square statistic is 1357.5 on 185 degrees of freedom.
## The p-value is 1.88e-177
```

Question 9.1: Are the factors independent?

```
### ANSWER ###
```

No, they are weakly-to-moderately correlated.

Question 9.2: What component of the printout tells us this?

```
### ANSWER ###
```

That would be the "Factor Correlations" section of the printout.

Task 10: Create scores for each subject

Hint: The factor analysis function has a `scores` parameter.

```
### ANSWER ###

fa <- factanal(df, factors = 5, scores = "Bartlett", data = df,
               na.action = na.exclude)
```

Task 11: Add these to the dataset.

Hint: Check the function documentation to find out where the scores are stored.

```
### ANSWER ###

df$f1= fa$scores[ , "Factor1"]
```

Bravo!

Extra credit if you finish early

1. Try doing all of this with IQ data set `Holzinger` from `psych`.
2. Do an FA on some of your own data, or... anything else: practise creates skill.
3. Play with the options to `paran()` and `factanal()`.

To prepare for next week's tutorials and lectures:

1. Install the package `umx`.
2. Read the `?umxRAM` help, and run one model from its help examples.
3. **Advanced credit:** Try and re-run one of the factor analyses using `umxFactanal()`.

Scientific as opposed to statistical Questions:

1. Do you think personality has 5 or 6 major domains?

2. Is the BFI data good?
3. What would happen to the parallel analysis if we sampled facets better?
4. What could go wrong if the data have a hierarchical structure like we know personality does?