Exploratory Factor Analysis

Joseph Nathan Cohen

Fall 2019

Explanation

Exploratory Factor Analysis (EFA) is a method that uses correlations or covariance metrics to reduce a larger number of variables to a smaller number of latent variables (or "factors"). This process is sometimes called "reducing the dimensionality" of a data set. We will perform these operations using the **fa()** function in the ***psych*** package.

Illustrative Example

Survey researchers ask a battery of 50 political opinion questions. They suspect that many of these opinions are related, and that these 50 questions could be boiled down to fewer "meta-issues" or dimensions of political opinion. For example, we might find that attitudes towards taxes, government regulation, and free trade might be correlated such that we could reduce people's answers on these three questions to one "free market-government intervention" scale. Alternatively, we might be able to combine bleiefs about abortion, school prayer, and public funding of religious schools down to a metric on attitudes towards "religion". You would use an exploratory factor analysis to identify these kinds of variable clusters empirically.

Guidelines

Some guidelines:[1]

- The method assumed univariate and multivariate normality in data,
- It assumes a linear relationship between factors and variables,
- Factors should have at least three variables
- Recommended sample size of N $>$ 300
- Ratio of respondents to variables should be at least 10:1, but more than 30:1 is preferable
- Correlations should be greater than 0.30 among variables that are potentially related to a common factor
- Variables related to teh same factor should be positively correlated, so reverse-code variables with strong negative correlations associated with the same factor.

The Model

EFA is based on regressions, but predicts individual variable values as a function of underlying latent variables (or "factors"):

$$X_j = a_{j1} \cdot F_1 + a_{j2} \cdot F_2 + ... + a_{jm} \cdot F_m + \epsilon_j$$

In which:

- X_j is variable j to be expressed as a sum of factors F_m
- F_m is factor m
- a_{jm} is factor loading m for variable j
- ϵ_j is an error term

[1] Source: An Gie Yong and Sean Pierce (2013) "A Beginner's Guide to Exploratory Factor Analysis" *Tutorials in Quantitative Methods for Psychology*, 9 (2): 79 - 94.

An extended exposition of the method is given in Everitt and Hothern (2011, Ch. 5).[2] In effect, the procedure uses variable correlations as a basis for estimating the latent factors' coefficients (or "factor loadings") in the system of equations above.

Implementation

Data

Our data is this module's simulated emotional disposition data, contained in the Excel spreadsheet "Simulated Emotional Disposition Data.xlsx". The set scores 400 respondents over 15 traits: happiness, optimism, sociability, anxiety, anger, jealousy, resentment, fear, boredom, tiredness, annoyance, irritability, hopefulness, friendliness, and ambition. These traits are all rated on a scale of zero (fully inapplicable) to ten (fully applicable).

```
library(readxl)
DATA <- read_xlsx("Simulated Emotional Disposition Data.xlsx", sheet = 1)

#Convert DATA object to data frame:
DATA <- data.frame(DATA)

#A quick look at the data (table is truncated for legibility):
head(DATA[1:8], 5)
```

```
##   id happiness optimism sociability anxiety anger jealousy resentment
## 1  1         6        7           7       5     5        5          6
## 2  2         7        8           7       3     8        6          5
## 3  3         5        5           6       3     5        7          5
## 4  4         3        3           3       5     4        5          4
## 5  5         6        6           8       7     6        6          4
```

From the outset, it is important to stress that we are working with simulated data (click here for the script used to generate the data). These data have been designed to create strong results. You would be hard pressed to find results that work this well in the real world. Keep that in mind when you conduct your own analyses and are tempted to use these findings as a benchmark against which to compare your own results.

Step 1: Standardize Variables

This is a step to take if you are trying to reduce variables that are denominated on different scales. This is not the case here, as all of our variables are denominated on the aforementioned zero to ten scale.

If this were not the case, and we wanted to standardize our continuous variables so that they are denominated on a common scale, we would use the **standardize()** function in the ***psycho*** package.[3]

```
library(psycho)
STD.DATA <- standardize(DATA[2:16])

#A quick look at the standardized data (table is truncated
#and figures are rounded for legibility):
head(round(STD.DATA[1:8],1), 3)
```

```
##   happiness optimism sociability anxiety anger jealousy resentment fear
## 1       0.6      1.0         0.9     0.0   0.1     -0.1        0.8 -0.6
## 2       1.1      1.4         0.9    -1.2   1.5      0.7       -0.1 -0.2
```

[2] Brian Everitt and Torsten Hothorn (2011) *An Introduction to Applied Multivariate Analysis with R* Springer.

[3] We use this command in lieu of the base packages **scale()** command because **standardize()** ignores character variables instead of returning an error. This feature ends up being convenient when you are working with a set that has numeric and text variables interspersed throughout the data frame

```
## 3       0.1      0.0         0.4    -1.2   0.1      1.5       -0.1  0.6
```

Step 2: Calculate correlation or covariance matrix for EFA

Next, calculate the correlation or covariance matrix of the variables you wish to use in the EFA. If all of your variables are continuous, then you can use the **cor()** function in the base package. That is what we will do here.

However, if your EFA includes nominal or ordinal variables, use **polychoric()** in the ***psych*** package to get polychoric correlations.

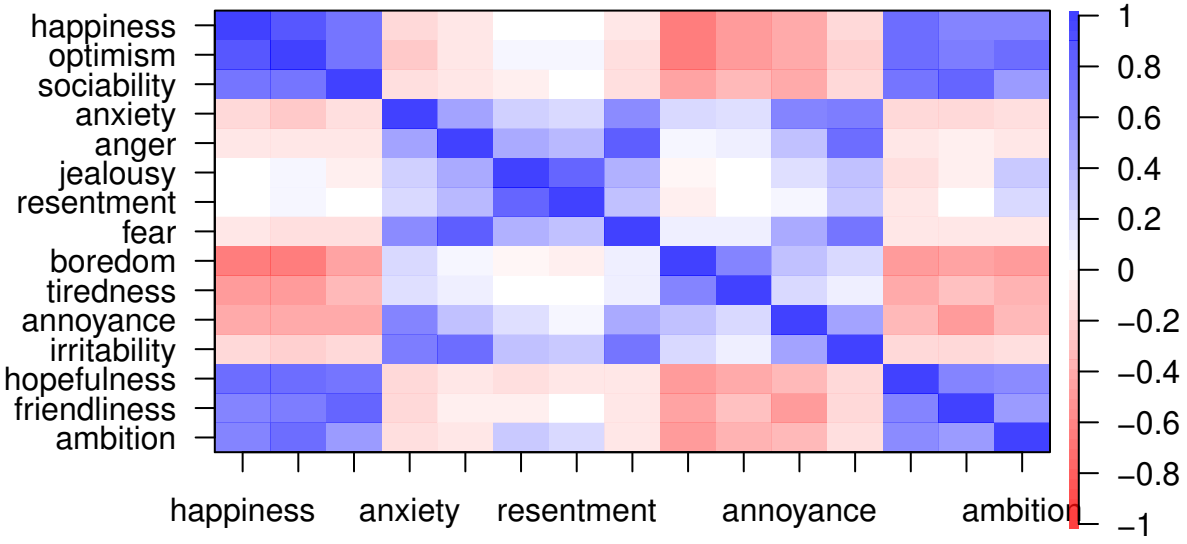
Note that, in the operation below, I do not use the first column of the data frame. That column contains unit identifiers.

```
# Run the operation to obtain polychoric correlations of the
# variables we wish to run in the EFA
CORR <- cor(DATA[-1])

# Isolate the number of observations used in calculating the correlations.
# You will use this in the next step.
N.OBS <- nrow(DATA)

#Here's the first few columns of the resulting matrix.
#It is rounded to the second decimal place to improve legibility
round(CORR[,c(1:6)], 2)
```

```
##              happiness optimism sociability anxiety anger jealousy
## happiness         1.00     0.86        0.71   -0.21 -0.10    -0.01
## optimism          0.86     1.00        0.71   -0.27 -0.13     0.03
## sociability       0.71     0.71        1.00   -0.17 -0.12    -0.08
## anxiety          -0.21    -0.27       -0.17    1.00  0.48     0.23
## anger            -0.10    -0.13       -0.12    0.48  1.00     0.42
## jealousy         -0.01     0.03       -0.08    0.23  0.42     1.00
## resentment        0.00     0.03       -0.02    0.19  0.35     0.78
## fear             -0.11    -0.16       -0.16    0.60  0.82     0.38
## boredom          -0.66    -0.67       -0.49    0.20  0.05    -0.06
## tiredness        -0.51    -0.50       -0.37    0.16  0.06     0.02
## annoyance        -0.42    -0.44       -0.44    0.62  0.31     0.14
## irritability     -0.20    -0.25       -0.21    0.69  0.75     0.32
## hopefulness       0.76     0.76        0.69   -0.18 -0.11    -0.15
## friendliness      0.64     0.68        0.80   -0.19 -0.09    -0.07
## ambition          0.64     0.74        0.51   -0.17 -0.11     0.26
```

You can visualize the correlation matrix using **cor.plot()** in the ***psych*** package.

```
library(psych)
cor.plot(CORR)
```



Step 3: Run an EFA

Then, use the **fa()** command with the resulting correlation matrix. Here is an example of the implementation. An explanation of your options follows:

```
#Run the EFA using the correlation matrix
library(psych)
EFA.MODEL <- fa(CORR,
                nfactors = 3,        #Specifies number of factors
                n.obs = N.OBS,       #Number of observations
                fm = 'minres',       #Factoring method
                rotate = 'varimax')  #Specify rotation method
```

This operation includes three choices: the number of factors, the factoring method, and the rotation method.

3.1: Factoring Method

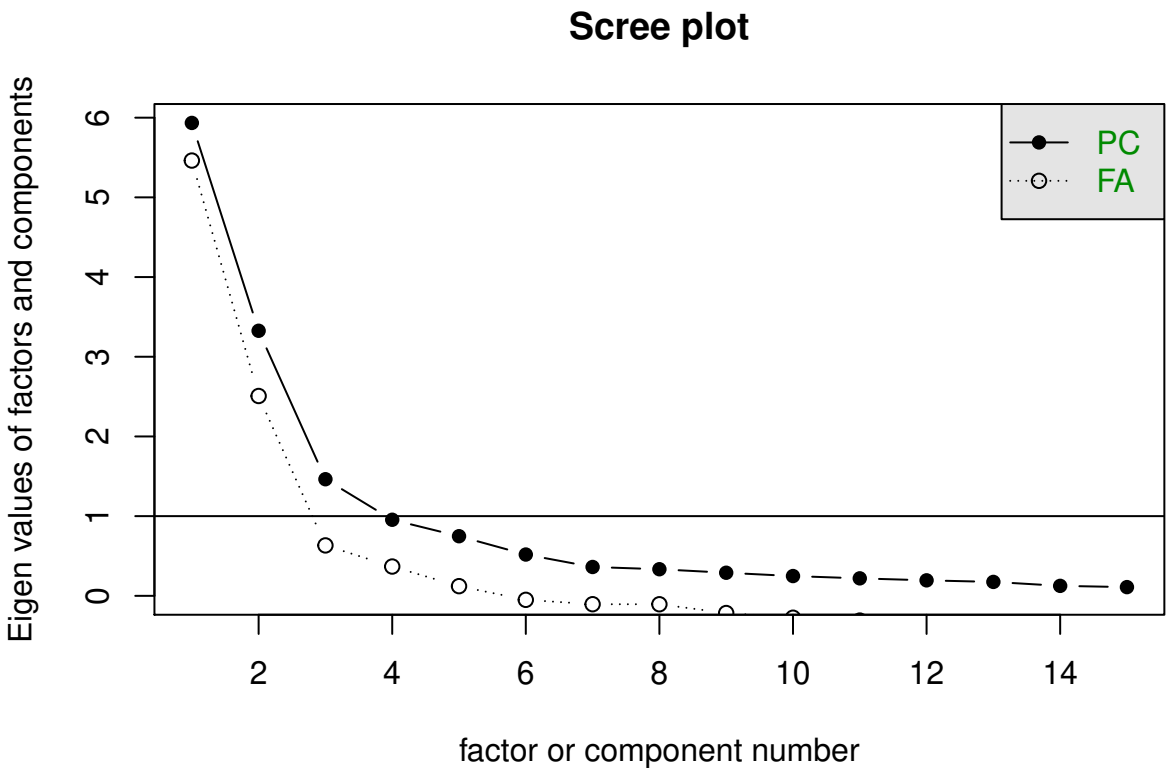
Factoring method (chosen by the "fm=" option) specifies the method by which we calculate factors. The methods are discussed in greater detail in Everett and Hothorn. Here, I used the 'minres' option, which relis on OLS regression. This is a default that works well in most situations. A 'gls' is said to increase the impact of unique/uncorrelated variables in your result. An "ml" specification runs a maximium-likelihood analysis, which many audiences will perceive as "more respectable" according to Everett and Hothorn

3.2: Number of Factors

Number of factors (chosen by the "nfactors=" option) specifies the number of factors to model. The goal is to model enough factors to capture as much variation in the data is possible, but without additional superfluous

factors that capture minimal variation. One way to depict these diminishing returns is by generating a scree plot using your correlation matrix. You can do this with the **scree()** command:

```
scree(CORR)
```



We are looking at the "FA" line in this figure. The "PC" is for a principal components analysis – another operation. The vertical axis can be understood as depicting the variance absorbed by each additional factor added to the model. It suggests that our model absorbs substantially more variance in adding a second or third factor, but that the additional gains when we add a fourth or more factors are minimal. This inflection point at M = 3 suggests that using three factors gives us a balance between explaining variance and parsimony.

3.3: Rotation Method

Factor rotation involves transforming the results of the analysis to make interpreting the factors easier. Here's a handy guide to rotation methods from IBM:

- *Varimax Method.* An orthogonal rotation method that minimizes the number of variables that have high loadings on each factor. This method simplifies the interpretation of the factors.
- *Direct Oblimin Method. A method for oblique (nonorthogonal) rotation. When delta equals 0 (the default), solutions are most oblique. As delta becomes more negative, the factors become less oblique. To override the default delta of 0, enter a number less than or equal to 0.8.
- *Quartimax Method.* A rotation method that minimizes the number of factors needed to explain each variable. This method simplifies the interpretation of the observed variables.
- *Equamax Method.* A rotation method that is a combination of the varimax method, which simplifies the factors, and the quartimax method, which simplifies the variables. The number of variables that load highly on a factor and the number of factors needed to explain a variable are minimized.

- *Promax Rotation.* An oblique rotation, which allows factors to be correlated. This rotation can be calculated more quickly than a direct oblimin rotation, so it is useful for large datasets.

Step 4: Adjust for Factor Loadings

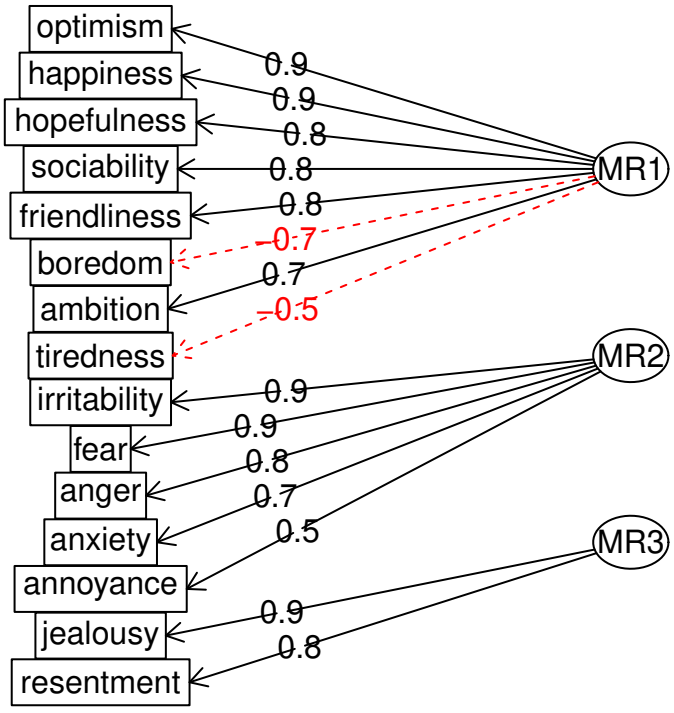
Factor loadings are describe the relationship between variables and factors. They are like coefficients. The resulting factor loadings are stored as a sub-object:

```
EFA.MODEL$loadings
```

```
##
## Loadings:
##              MR1    MR2    MR3
## happiness     0.898
## optimism      0.916 -0.156
## sociability   0.799 -0.103
## anxiety      -0.174  0.724
## anger                0.790  0.242
## jealousy             0.248  0.939
## resentment           0.209  0.768
## fear                 0.853  0.195
## boredom      -0.704
## tiredness    -0.549
## annoyance    -0.437  0.505
## irritability -0.133  0.863  0.125
## hopefulness   0.820        -0.157
## friendliness  0.756 -0.117
## ambition      0.699 -0.141  0.294
##
##                  MR1   MR2   MR3
## SS loadings    5.056 3.076 1.718
## Proportion Var 0.337 0.205 0.115
## Cumulative Var 0.337 0.542 0.657
```

Factor loadings can be visualized using the command **fa.diagram()**:

```
fa.diagram(EFA.MODEL)
```



Ideally, there will be either strong (e.g., >0.80) or low (near-zero) loadings. The variables with high factor loadings can be used to make sense of the factor's meaning. For example, in the figure above, Facotr M2 might be characterized as "visceral negative emotional reations" – at least that's one idea for tying together these related variables. Factor analysis involves some interpretive work, not just math.

Conventionally, for a sample size of N > 300, a rotated loading should be at least 0.32 to be considered substantial. Things can get more complicated when you have variables that *crossload* (with loadings of at least 0.32 on at least two factors). Our "annoyance" variable is cross-loading here (see table above). I'm going to respond by removing it from the analysis.

Also, you should examine the variables to ensure that none of the factor loadings register as negative. If they do, you should reverse-code the variable so that factor loadings are all positive. It looks like I need to reverse-code "tiredness" and "boredom"

Finally, it looks like "jealousy" and "resentment" are grouped in a factor of their own. Recall that it is recommended that a factor should have at least three variables that primarily load onto it. For this and other reasons that emerged in this analysis, I'm going to eliminate these variables.

I perform these adjustments here:

```
DATA.2 <- DATA

#Reverse coding variables:
DATA.2$tiredness <- -1 * DATA.2$tiredness
DATA.2$boredom <- -1 * DATA.2$boredom

#A new correlation matrix without columns of
#variables that I plan to exclude (and with identifier excluded):
CORR.2 <- cor(DATA.2[-c(1,7,8,12)])
```

```
#Rerun the analysis, now with 2 factors:
EFA.MODEL.2 <- fa(CORR.2,
                  nfactors = 2,        #Specifies number of factors
                  n.obs = N.OBS,       #Number of observations
                  fm = 'minres',       #Factoring method
                  rotate = 'varimax')  #Specify rotation method
```

Step 5: Interpret the Results

5.1: Factor Loadings

The operation returns an object that delivers a lot of information. Start by examining the factor loadings, as in Step 4. Look for cross-loaded variables, orphaned variables, factors with only one or two variables primarily loaded onto it, and whatever other problems you might detect.

```
EFA.MODEL.2$loadings
```

```
##
## Loadings:
##               MR1    MR2
## happiness      0.902
## optimism       0.927 -0.145
## sociability    0.787 -0.119
## anxiety       -0.169  0.661
## anger                 0.853
## fear                  0.892
## boredom        0.714
## tiredness      0.556
## irritability -0.135  0.876
## hopefulness    0.806
## friendliness   0.742 -0.112
## ambition       0.705
##
##                  MR1   MR2
## SS loadings    4.857 2.811
## Proportion Var 0.405 0.234
## Cumulative Var 0.405 0.639
```

It can help to see them visualized, using **fa.diagram()** as above. Here, the analysis seems to have performed fine. Pretty strong loadings, no cross-loadings, an sensible groupings.

Below the factor loadings, you will see sum-of-square estimates. This describes the proportion of variable caputred by each factor. Factor MR1 captures about 41% of variance, and MR2 caputres another 23%. Jointly, these factors capture 64% of variance.

5.2: Communiality & Uniqueness Estimates

Communality of variables represent the variance that each variable shares with others via the common factors. *Uniqueness* is the proportion of each variable's variation not attributed to the factor. Both estimates should sum to 100%.

```
EFA.MODEL.2$communalities
```

```
##    happiness     optimism  sociability      anxiety        anger
##    0.8216510    0.8801044    0.6343575    0.4650925    0.7281047
##         fear      boredom    tiredness irritability  hopefulness
```

```
##    0.7968404    0.5159365    0.3122122    0.7858441    0.6598850
## friendliness     ambition
##    0.5631188    0.5050897
```

```
EFA.MODEL.2$uniquenesses
```

```
##    happiness     optimism  sociability      anxiety        anger
##    0.1783500    0.1198960    0.3656421    0.5349090    0.2718943
##         fear      boredom    tiredness irritability  hopefulness
##    0.2031558    0.4840638    0.6877868    0.2141581    0.3401162
## friendliness     ambition
##    0.4368811    0.4949104
```

Consider the "optimism" variable. These results suggest that about 88% of the observed variation in "optimism" scores are a product of the underlying factor that drives other variables like "happiness" or "sociability". If we reduce this variable into a a factor with these other variables, we aren't likely to lose as much informatino.
On the other hand, "tiredness" has relatively more uniqueness (0.68). Our factors are not aborbing a lot of the observed variation here.

5.3: Complexity

Hoffman's Complexity Index gives the average number of factor needed to account for our observed variables variables.[4] A factor analysis with a simple structure will allow us to group like variables into single factors, in which case it would only require about one factor to summarize it. Recall that we removed cross-loaded factors, which required more than one factor to describe their variance. If you were to run complexity metrics on those variables, you'd find that the need about 1.5 factors.

```
EFA.MODEL.2$complexity
```

```
##    happiness     optimism  sociability      anxiety        anger
##     1.020436     1.049221     1.045871     1.130488     1.000487
##         fear      boredom    tiredness irritability  hopefulness
##     1.003389     1.025960     1.016915     1.047792     1.030124
## friendliness     ambition
##     1.045453     1.034794
```

5.4: Fit Statistics

The operation returns several fit statistics.

```
summary(EFA.MODEL.2)
```

```
##
## Factor analysis with Call: fa(r = CORR.2, nfactors = 2, n.obs = N.OBS, rotate = "varimax",
##     fm = "minres")
##
## Test of the hypothesis that 2 factors are sufficient.
## The degrees of freedom for the model is 43  and the objective function was  1.15
## The number of observations was  400  with Chi Square =  451.66  with prob <  1.5e-69
##
## The root mean square of the residuals (RMSA) is  0.05
## The df corrected root mean square of the residuals is  0.06
##
## Tucker Lewis Index of factoring reliability =  0.825
```

[4] Erik Pettersson and Eric Turkheimer (2010) "Item Selection, Evaluation, and Simple Structure in Personality Data" *Journal of Research in Personality*, 44: 407 - 420.

```
## RMSEA index =  0.156  and the 10 % confidence intervals are  0.142 0.167
## BIC =  194.03
```

The results return four clusters of information. Many of these metrics are more important in confirmatory factor analysis, but they can still be information sources:[5]

- The first cluster, comprised of the top line, repeats the specification of the model being summariezed
- The second cluster, a three line set, gives a chi-square test of the null hypothesis that our model fits perfectly. You want a p-value above 0.05. Rejecting the null implies that your system of factors are not fully capturing the variance in your variable set.
- The bottom cluster includes several fit statistics. ** The Tucker-Lewis Index (TLI) gives the fit improvement over the null model. In confirmatory factor analysis (CFA), one would strive for a score of at least 0.95 ** The Root Mean Squared Error of Approximation (RMSEA) is a parsimony-adjusted index, and one strives for a score below 0.08 in CFA ** The Bayesian Information Criterion (BIC) is a fit statistic. An in-depth discussion of this criterion can be read here. Use this to compare different attempts at factor analysis. The lower BIC score is the better-fitting model.

Overall, this is a poor-fitting system of factors. However, the exercise did give us leads about which kinds of variables might be combined to reduce the dimensionality of our set. To get a more definitive answer on whether the kinds of data reductions we see here are doable, the gold standard method is confirmatory factor analysis

[5]For more on CFA fit statistics, our source is this useful handout from Cornell Statistical Consulting