R Examples Repository

Linear algebra calculations (http://dwoll.de/rexrepos/posts/linearAlgebra.html)
Principal components analysis (http://dwoll.de/rexrepos/posts/multPCA.html)
Factor analysis (http://dwoll.de/rexrepos/posts/multFA.html)
Factor analysis for ordinal data
Multidimensional scaling (http://dwoll.de/rexrepos/posts/multMDS.html)
Multivariate multiple regression (http://dwoll.de/rexrepos/posts/multRegression.html)
Hotelling's T^2-test (http://dwoll.de/rexrepos/posts/multHotelling.html)
Multivariate analysis of variance (http://dwoll.de/rexrepos/posts/multMANOVA.html)
Linear discriminant analysis (http://dwoll.de/rexrepos/posts/multLDA.html)
Assess multivariate normality (http://dwoll.de/rexrepos/posts/normality.html)

TAGS

FactorAnalysis (http://dwoll.de/rexrepos/tags.html#FactorAnalysis-ref)

Exploratory factor analysis for ordinal categorical data

- Install required packages
- Factor analysis
  - Simulate categorical data based on continuous variables
  - Factor analysis for ordered categorical data
- Factor scores
- Visualize loadings
- Determine number of factors
- Detach (automatically) loaded packages (if possible)
- Get the article source from GitHub

Install required packages

`GPArotation` (http://cran.r-project.org/package=GPArotation), `mvtnorm` (http://cran.r-project.org/package=mvtnorm), `polycor` (http://cran.r-project.org/package=polycor), `psych` (http://cran.r-project.org/package=psych)

```
wants <- c("GPArotation", "mvtnorm", "polycor", "psych")
has   <- wants %in% rownames(installed.packages())
if(any(!has)) install.packages(wants[!has])
```

Factor analysis

Simulate categorical data based on continuous variables

First, let's simulate 200 observations from 6 variables, coming from 2 orthogonal factors. I'll take a couple of intermediate steps and start with multivariate normal continuous data that I later dichotomize. That way, we can compare Pearson correlations with polychoric correlations, and compare factor loadings from continuous data with that from dichotomous data and the true loadings.

```
set.seed(123)
N <- 200                      # number of observations
P <- 6                        # number of variables
Q <- 2                        # number of factors

# true P x Q loading matrix -> variable-factor correlations
Lambda <- matrix(c(0.7,-0.4, 0.8,0, -0.2,0.9, -0.3,0.4, 0.3,0.7, -0.8,0.1),
                 nrow=P, ncol=Q, byrow=TRUE)
```

Now simulate the actual data from the model $x = \Lambda f + e$, with x being the observed variable values of a person, Λ the true loadings matrix, f the latent factor score, and e iid, mean 0, normal errors.

```
# factor scores (uncorrelated factors)
library(mvtnorm)              # for rmvnorm()
FF <- rmvnorm(N, mean=c(5, 15), sigma=diag(Q))

# matrix with iid, mean 0, normal errors
E   <- rmvnorm(N, rep(0, P), diag(P))
X   <- FF %*% t(Lambda) + E    # matrix with variable values
dfX <- data.frame(X)           # data also as a data frame
```

Now let's categorize the data. We'll keep the data in two formats: as a data frame with ordered factors, and as a numeric matrix. `hetcor()` from package `polycor` gives us the polychoric correlation matrix we'll later use for the FA.

```
# categorize variables into a list of ordered factors
lOrd <- lapply(dfX, function(x) {
              cut(x, breaks=quantile(x), include.lowest=TRUE,
                  ordered=TRUE, labels=LETTERS[1:4]) })
dfOrd  <- data.frame(lOrd)     # combine list into a data frame
ordNum <- data.matrix(dfOrd)   # categorized data as a numeric matrix
```

Factor analysis for ordered categorical data

Use the polychoric correlation matrix to do a regular FA.

```
library(polycor)              # for hetcor()
pc <- hetcor(dfOrd, ML=TRUE)   # polychoric corr matrix
```

```
library(psych)
faPC <- fa(r=pc$correlations, nfactors=2, n.obs=N, rotate="varimax")
faPC$loadings
```

```
Loadings:
   MR2    MR1
X1  0.546 -0.196
X2  0.607
X3 -0.173  0.842
X4 -0.197  0.311
X5  0.336  0.537
X6 -0.619

                 MR2   MR1
SS loadings    1.231 1.133
Proportion Var 0.205 0.189
Cumulative Var 0.205 0.394
```

It is possible to skip the step of calculating the polychoric correlation matrix, and directly use `fa.poly()` from package `psych`, which does the same thing in the end. This function accepts the raw dichotomous data as a numeric matrix.

```
# polychoric FA
faPCdirect <- fa.poly(ordNum, nfactors=2, rotate="varimax")
```

```
faPCdirect$fa$loadings         # loadings are the same as above ...
```

```
Loadings:
   MR2    MR1
X1  0.546 -0.196
X2  0.605
X3 -0.174  0.841
X4 -0.198  0.311
X5  0.336  0.538
X6 -0.621

                 MR2   MR1
SS loadings    1.231 1.133
Proportion Var 0.205 0.189
Cumulative Var 0.205 0.394
```

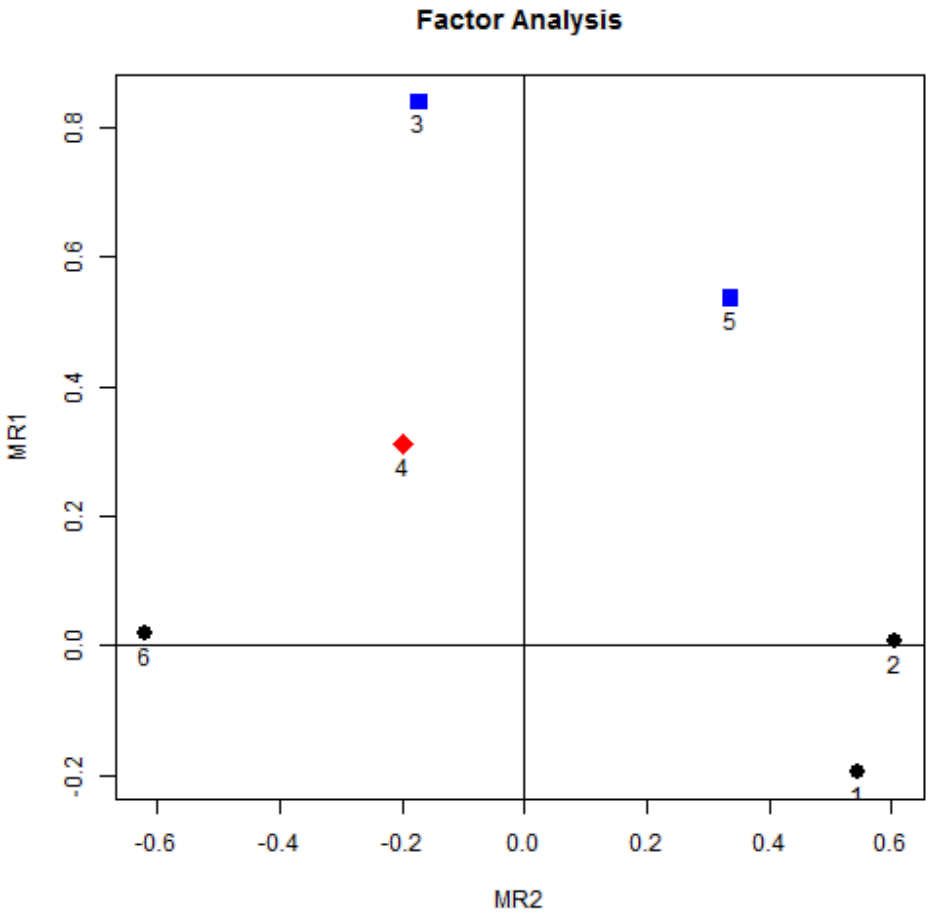
Factor scores

For factor scores, look at package `ltm` which has a `factor.scores()` function specifically for polytomous outcome data. An example is provided on this page (http://eur.academia.edu/DimitrisRizopoulos/Teaching) -> "Factor Scores - Ability Estimates".

Visualize loadings

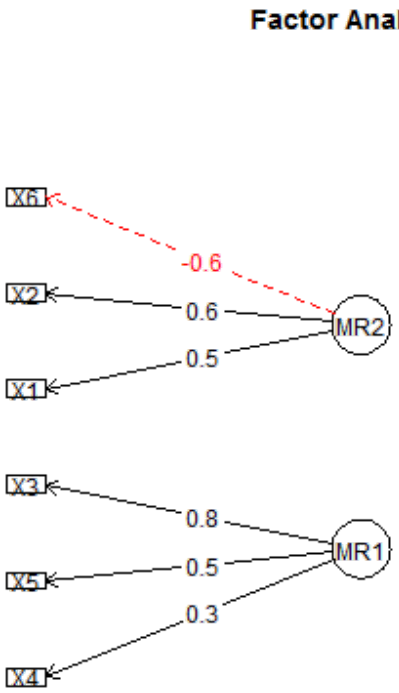
You can visualize the loadings from the factor analysis using `factor.plot()` and `fa.diagram()`, both from package `psych`. For some reason, `factor.plot()` accepts only the `$fa` component of the result from `fa.poly()`, not the full object.

```
factor.plot(faPCdirect$fa, cut=0.5)
```



plot of chunk rerMultFApoly01

```
fa.diagram(faPCdirect)
```

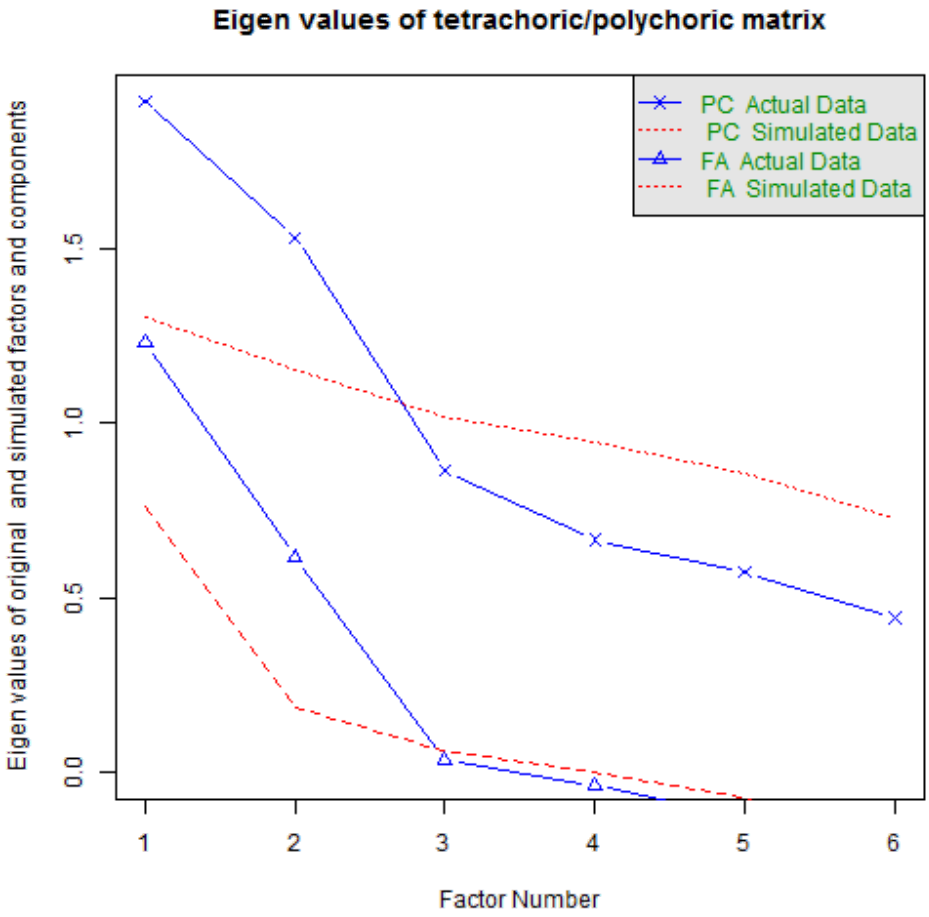


plot of chunk rerMultFApoly01

Determine number of factors

Parallel analysis and a "very simple structure" analysis provide help in selecting the number of factors. Again, package `psych` has the required functions. `vss()` takes the polychoric correlation matrix as an argument.

```
fap <- fa.parallel.poly(ordNum)   # parallel analysis for dichotomous data
```



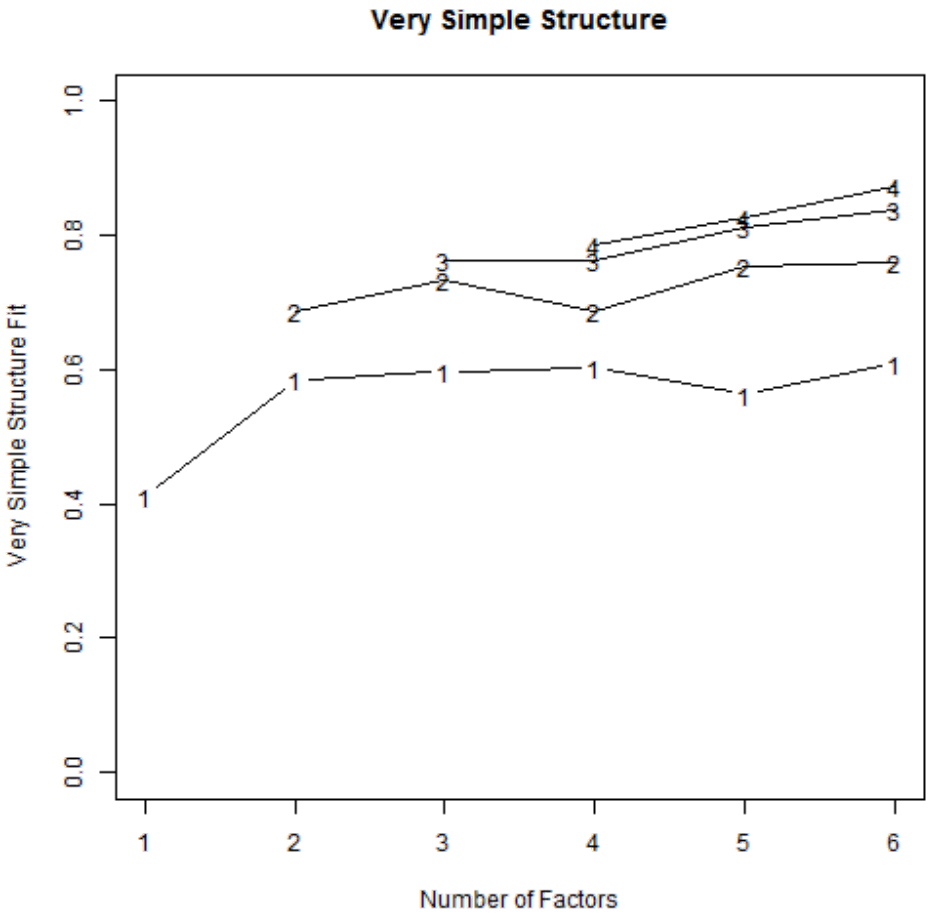
plot of chunk rerMultFApoly02

```
fap
```

```
Call: fa.parallel.poly(x = ordNum)
Parallel analysis suggests that the number of factors =  2  and the number of components =  2

 Eigen Values of
  Original factors Simulated data Original components simulated data
1             1.23           0.76                1.92           1.30
2             0.62           0.19                1.53           1.15
```

```
vss(pc$correlations, n.obs=N, rotate="varimax")  # very simple structure
```



plot of chunk rerMultFApoly03

```
Very Simple Structure
Call: VSS(x = x, n = n, rotate = rotate, diagonal = diagonal, fm = fm,
    n.obs = n.obs, plot = plot, title = title)
VSS complexity 1 achieves a maximimum of 0.61  with  6  factors
VSS complexity 2 achieves a maximimum of 0.76  with  6  factors

The Velicer MAP criterion achieves a minimum of NA  with  1  factors

Velicer MAP
[1] 0.08 0.11 0.22 0.40 1.00   NA

Very Simple Structure Complexity 1
[1] 0.41 0.58 0.60 0.60 0.56 0.61

Very Simple Structure Complexity 2
[1] 0.00 0.69 0.73 0.69 0.75 0.76
```

Detach (automatically) loaded packages (if possible)

```
try(detach(package:GPArotation))
try(detach(package:psych))
try(detach(package:polycor))
try(detach(package:sfsmisc))
try(detach(package:mvtnorm))
```

(based on an answer I wrote on CrossValidated)

Get the article source from GitHub

R markdown (https://github.com/dwoll/RExRepos/raw/master/Rmd/multFApoly.Rmd) - markdown (https://github.com/dwoll/RExRepos/raw/master/md/multFApoly.md) - R code (https://github.com/dwoll/RExRepos/raw/master/R/multFApoly.R) - all posts (https://github.com/dwoll/RExRepos/)

top

2013 Daniel Wollschlaeger - licensed under CC-BY-SA (http://creativecommons.org/licenses/by-sa/3.0/)