Exploratory Factor Analysis of Simulated Data*

```
> rm(list=ls())
>
> # Factor analysis of simulated data
> # Two orthogonal factors, normal distribution
>
> rm(list=ls())
> n = 50000 # Huge sample size
> # True factor loadings have a simple structure like varimax (All communalities = 0.49)
> # Factor loadings
> L11 = 0.7; L12 = 0.0
> L21 = 0.7; L22 = 0.0
> L31 = 0.7; L32 = 0.0
> L41 = 0.7; L42 = 0.0
> L51 = 0.0; L52 = 0.7
> L61 = 0.0; L62 = 0.7
> L71 = 0.0; L72 = 0.7
> L81 = 0.0; L82 = 0.7
> # Error Variances
> v1 = 1 - L11**2 - L12**2
> v2 = 1 - L21**2 - L22**2
> v3 = 1 - L31**2 - L32**2
> v4 = 1 - L41**2 - L42**2
> v5 = 1 - L51**2 - L52**2
> v6 = 1 - L61**2 - L62**2
> v7 = 1 - L71**2 - L72**2
> v8 = 1 - L81**2 - L82**2
> # Generate data
> set.seed(9999)
> F1 = rnorm(n,0,1); F2 = rnorm(n,0,1)
> d1 = L11*F1 + L12*F2 + rnorm(n,0,sqrt(v1))
> d2 = L21*F1 + L22*F2 + rnorm(n,0,sqrt(v2))
> d3 = L31*F1 + L32*F2 + rnorm(n,0,sqrt(v3))
> d4 = L41*F1 + L42*F2 + rnorm(n,0,sqrt(v4))
> d5 = L51*F1 + L52*F2 + rnorm(n,0,sqrt(v5))
> d6 = L61*F1 + L62*F2 + rnorm(n,0,sqrt(v6))
> d7 = L71*F1 + L72*F2 + rnorm(n,0,sqrt(v7))
> d8 = L81*F1 + L82*F2 + rnorm(n,0,sqrt(v8))
> dmat = cbind(d1,d2,d3,d4,d5,d6,d7,d8)
>
```

```
> factanal(dmat,factors=2,rotation='varimax')


Call:
factanal(x = dmat, factors = 2, rotation = "varimax")

Uniquenesses:
   d1    d2    d3    d4    d5    d6    d7    d8 
0.506 0.510 0.519 0.511 0.507 0.505 0.508 0.510 

Loadings:
   Factor1 Factor2
d1          0.698 
d2          0.694 
d3          0.688 
d4          0.695 
d5  0.697         
d6  0.699         
d7  0.696         
d8  0.695         

               Factor1 Factor2
SS loadings      1.971   1.953
Proportion Var   0.246   0.244
Cumulative Var   0.246   0.491

Test of the hypothesis that 2 factors are sufficient.
The chi square statistic is 10.22 on 13 degrees of freedom.
The p-value is 0.676 

> # The truth again for comparison
> Lambda = rbind(c(L11,L12),
+                c(L21,L22),
+                c(L31,L32),
+                c(L41,L42),
+                c(L51,L52),
+                c(L61,L62),
+                c(L71,L72),
+                c(L81,L82)   )
> Lambda

     [,1] [,2]
[1,]  0.7  0.0
[2,]  0.7  0.0
[3,]  0.7  0.0
[4,]  0.7  0.0
[5,]  0.0  0.7
[6,]  0.0  0.7
[7,]  0.0  0.7
[8,]  0.0  0.7
```

```
> # Example 2: Truth is not like varimax (All communalities = 0.50)
>
> # Factor loadings
> L11 = 0.5; L12 = -0.5
> L21 = 0.5; L22 = -0.5
> L31 = 0.5; L32 = -0.5
> L41 = 0.5; L42 = -0.5
> L51 = 0.5; L52 =  0.5
> L61 = 0.5; L62 =  0.5
> L71 = 0.5; L72 =  0.5
> L81 = 0.5; L82 =  0.5
> # Error Variances
> v1 = 1 - L11**2 - L12**2
> v2 = 1 - L21**2 - L22**2
> v3 = 1 - L31**2 - L32**2
> v4 = 1 - L41**2 - L42**2
> v5 = 1 - L51**2 - L52**2
> v6 = 1 - L61**2 - L62**2
> v7 = 1 - L71**2 - L72**2
> v8 = 1 - L81**2 - L82**2
> # Generate data
> set.seed(8888)
> F1 = rnorm(n,0,1); F2 = rnorm(n,0,1)
> d1 = L11*F1 + L12*F2 + rnorm(n,0,sqrt(v1))
> d2 = L21*F1 + L22*F2 + rnorm(n,0,sqrt(v2))
> d3 = L31*F1 + L32*F2 + rnorm(n,0,sqrt(v3))
> d4 = L41*F1 + L42*F2 + rnorm(n,0,sqrt(v4))
> d5 = L51*F1 + L52*F2 + rnorm(n,0,sqrt(v5))
> d6 = L61*F1 + L62*F2 + rnorm(n,0,sqrt(v6))
> d7 = L71*F1 + L72*F2 + rnorm(n,0,sqrt(v7))
> d8 = L81*F1 + L82*F2 + rnorm(n,0,sqrt(v8))
> dmat = cbind(d1,d2,d3,d4,d5,d6,d7,d8)
>
> notsimple = factanal(dmat,factors=2,rotation='varimax'); notsimple

Call:
factanal(x = dmat, factors = 2, rotation = "varimax")

Uniquenesses:
   d1    d2    d3    d4    d5    d6    d7    d8 
0.496 0.496 0.504 0.504 0.497 0.495 0.503 0.499 

Loadings:
   Factor1 Factor2
d1          0.708 
d2          0.708 
d3          0.702 
d4          0.702 
d5  0.708         
d6  0.709         
d7  0.703         
d8  0.706         

               Factor1 Factor2
SS loadings      2.007   2.000
Proportion Var   0.251   0.250
Cumulative Var   0.251   0.501

Test of the hypothesis that 2 factors are sufficient.
The chi square statistic is 9.58 on 13 degrees of freedom.
The p-value is 0.728
```

```
> # Is there a possible rotation that would get us close to the truth?
> # Procrustes rotation
> # install.packages("MCMCpack", dependencies=TRUE) # Only need to do this once
> library(MCMCpack)
Loading required package: coda
Loading required package: MASS
##
## Markov Chain Monte Carlo Package (MCMCpack)
## Copyright (C) 2003-2023 Andrew D. Martin, Kevin M. Quinn, and Jong Hee Park
##
## Support provided by the U.S. National Science Foundation
## (Grants SES-0350646 and SES-0350613)
##
> # help(procrustes)
> # Estimated factor loadings for the second example
> L = notsimple$loadings; print(L,cutoff=0)

Loadings:
   Factor1 Factor2
d1  0.047   0.708
d2  0.056   0.708
d3  0.054   0.702
d4  0.052   0.702
d5  0.708  -0.050
d6  0.709  -0.054
d7  0.703  -0.052
d8  0.706  -0.052

               Factor1 Factor2
SS loadings      2.007   2.000
Proportion Var   0.251   0.250
Cumulative Var   0.251   0.501
> Lambda = rbind(c(L11,L12),               # True factor loadings
+                c(L21,L22),
+                c(L31,L32),
+                c(L41,L42),
+                c(L51,L52),
+                c(L61,L62),
+                c(L71,L72),
+                c(L81,L82)  )
> Lambda # True Lambda -- How close can we get to this?
     [,1] [,2]
[1,]  0.5 -0.5
[2,]  0.5 -0.5
[3,]  0.5 -0.5
[4,]  0.5 -0.5
[5,]  0.5  0.5
[6,]  0.5  0.5
[7,]  0.5  0.5
[8,]  0.5  0.5
> # Rotate X to approximate Xstar. It's orthogonal: X R ~ Xstar
> pro = procrustes(X = L, Xstar = Lambda)
> round(pro$X.new,2)
   [,1]  [,2]
d1  0.5 -0.51
d2  0.5 -0.50
d3  0.5 -0.50
d4  0.5 -0.50
d5  0.5  0.50
d6  0.5  0.50
d7  0.5  0.50
d8  0.5  0.50
```

```
> round( L %*% pro$R , 2) # Rotating "manually"
   [,1]  [,2]
d1  0.5 -0.51
d2  0.5 -0.50
d3  0.5 -0.50
d4  0.5 -0.50
d5  0.5  0.50
d6  0.5  0.50
d7  0.5  0.50
d8  0.5  0.50
> pro$R %*% t(pro$R) # R is an orthogonal matrix
              [,1]          [,2]
[1,]  1.000000e+00 -1.110223e-16
[2,] -1.110223e-16  1.000000e+00

>
> # Now try a Procrustes rotation of something unrelated to the truth
> M  = rbind(c(0.30,0.64),
+            c(0.30,0.64),
+            c(0.30,0.64),
+            c(0.30,0.64),
+            c(0.30,0.64),
+            c(0.30,0.64),
+            c(0.30,0.64),
+            c(0.30,0.64)  ); M
     [,1] [,2]
[1,]  0.3 0.64
[2,]  0.3 0.64
[3,]  0.3 0.64
[4,]  0.3 0.64
[5,]  0.3 0.64
[6,]  0.3 0.64
[7,]  0.3 0.64
[8,]  0.3 0.64
> procrustes(X = L, Xstar = M)$X.new
         [,1]      [,2]
d1 -0.2470153 0.6654424
d2 -0.2385050 0.6689701
d3 -0.2379146 0.6626890
d4 -0.2401607 0.6618827
d5  0.6661897 0.2441638
d6  0.6688511 0.2407410
d7  0.6624699 0.2407156
d8  0.6656312 0.2410942
>
> round(pro$X.new,2) # Repeating for comparison
   [,1]  [,2]
d1  0.5 -0.51
d2  0.5 -0.50
d3  0.5 -0.50
d4  0.5 -0.50
d5  0.5  0.50
d6  0.5  0.50
d7  0.5  0.50
d8  0.5  0.50


> # The truth is very close to some member of the set of factor matrices that can be
reached by an orthogonal rotation.  The problem is that we don't know which one.
>
```