Supplement to Accompany:

Direct Schmid-Leiman Transformations and Rank Deficient Loadings Matrices

Author

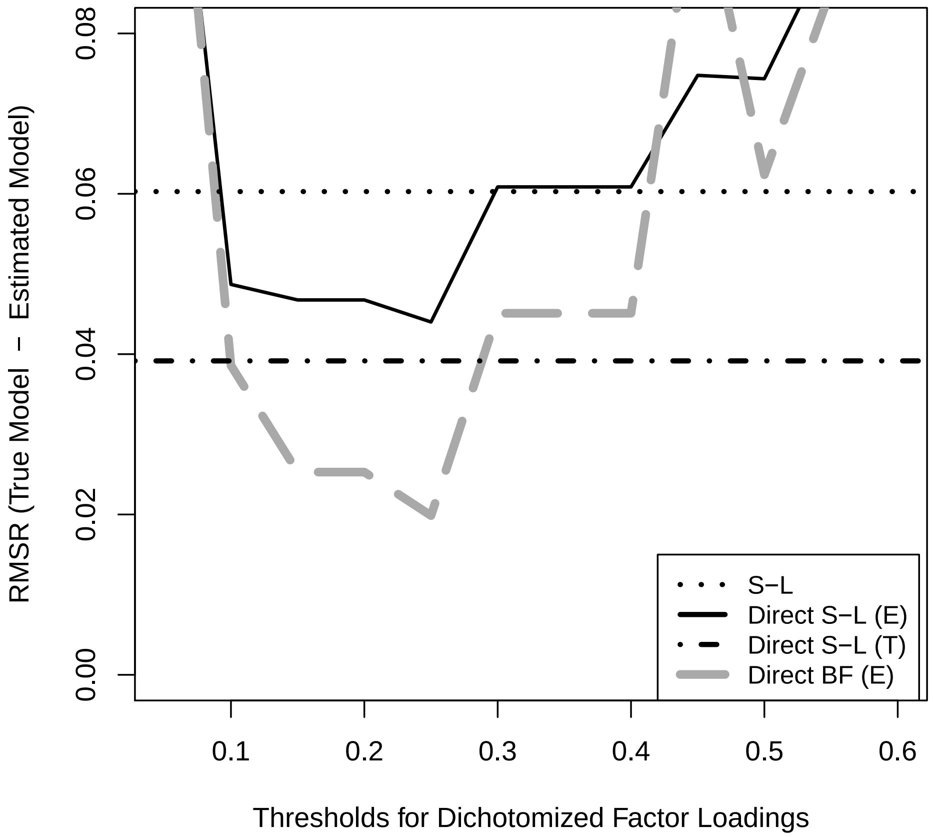
```
> BiFAD <- function(R, B = NULL, nGroup = NULL,
+                       factorMethod = "minres",
+                       rotation="oblimin",  salient = .15,
+                       maxitFA = 5000,
+                       maxitRotate = 5000,
+                       gamma = 0){
+
+ #################################################################
+ ## AUTHOR (blind copy)
+ ## August 18, 2017
+ ##  requires  psych:   package for initial factor extraction
+ ##            GPArotation:  for rotation options
+ ##
+ ## Arguments:
+ ##  R:  Input correlation matrix
+ ##
+ ##  B:  bifactor target matrix. If B=NULL the program will
+ ##        create an empirically defined target matrix.
+ ##
+ ##  nGroup: Number of group factors in bifactor solution.
+ ##
+ ##  factorMethod: factor extraction method. Options include:
+ ##        minres (minimum residual), ml (maximum likelihood),
+ ##        pa (principal axis), gls (generalized least squares).
+ ##
+ ##  rotation:  factor rotation method. Current options include:
+ ##        oblimin, geominQ, quartimin, promax.
+ ##
+ ##  salient:  Threshold value for creating an empirical target
+ ##        matrix.
+ ##
+ ##  maxitFA:  Maximum iterations for the factor extraction
+ ##        method.
+ ##
+ ##  maxitRotate: Maximum iterations for the gradient pursuit
+ ##        rotation algorithm.
+ ##
+ ##  gamma: Optional tuning parameter for oblimin rotation.
+ ##
+ ##  Value:
+ ##
+ ##    B: User defined or empirically generated target matrix.
+ ##
+ ##    BstarSL:  Direct S-L solution.
+ ##
+ ##    BstarFR:  Direct full rank bifactor solution.
+ ##
+ ##    rmsrSL:   Root mean squared residual of (B - BstarSL).
+ ##
+ ##    rmsrFR:   Root mean squared residual of (B - BstarFR).
+ ################################################################
+
+   ## Compute orthogonal Procrustes rotation
+   Procrustes <-function(M1, M2){
+     tM1M2 <- t(M1) %*% M2
+     svdtM1M2 <- svd(tM1M2)
+     P <- svdtM1M2$u
+     Q <- svdtM1M2$v
+     T <- Q %*% t(P)
+     ## Orthogonally rotate M2 to M1
+     M2 %*% T
+   }
+
+   ## FATAL ERROR
+   if(is.null(B) & is.null(nGroup)){
+     stop("\n\n FATAL ERROR: Either B or nGroup must be specified")
+   }
+
+  ## Compute unrotated factor solution
+  if(is.null(nGroup))  nGroup <- ncol(B) - 1
+
+  F <- psych::fa(r = R,
+         nfactors = nGroup,
+         fm = factorMethod,
+         rotate = "none")$loadings[]
+
+  ## Append column of zeros to create rank deficient matrix
+  L0 <- cbind(F,0)
+
+
+  ## Create 0/1 Target matrix if B not supplied
+  Bflag <- 1 # set to 1 if Target matrix (B) is known
+   if(is.null(B))  Bflag <- 0
+
+   if(!Bflag & (!is.null(nGroup))){
+
+    # Start rotation from random position
+    TmatRandom <- qr.Q(qr(matrix(rnorm(nGroup * nGroup ),
+                      nGroup )))
+    switch(rotation,
+          "oblimin" = {
+            gpa.out <- GPArotation::oblimin(F, Tmat=TmatRandom,
+                                            gam = gamma,
+                                            maxit=maxitRotate)
+          }, "geominQ" = {
+            gpa.out <- GPArotation::geominQ(F,
+                                            Tmat = TmatRandom,
+                                            maxit=maxitRotate)
+          }, "quartimin" = {
+            gpa.out <- GPArotation::quartimin(F,
+                                              Tmat = TmatRandom,
+                                              maxit=maxitRotate)
+          }, "promax" = {
+          gpa.out <- promax(F)
+          })
+
+    B <- gpa.out$loadings[]
+
+    ## Record signs of loadings
+    signB <- sign(B)
+
+    ## Convert Target matrix into signed 0/1 matrix
+    B <- signB* matrix(as.numeric(abs(B) >= salient),
+                        nrow(R), ncol = nGroup)
+
+    ## Append ones vector for general factor
+    B <- cbind(1,B)
+
+  } ## END Create 0/1 Target matrix
+
+
+  ## Rotate rank deficient loading matrix to Target
+  BstarSL <- Procrustes(B, L0)
+
+  ## Compute non-hierarchical bifactor solution
+  F2    <- psych::fa(r = R,
+                    nfactors = nGroup+1,
+                    fm=factorMethod,
+                    rotate="none")$loadings[]
+
+  ## Rotate full rank loading matrix to Target
+  BstarFR <- Procrustes(B, F2)
+
+  ## Compute root mean squared residual of Target matrix (B)
+  ## and best fitting SL and FR solutions (Bstar)
+  rmsrSL <- rmsrFR <- NA
+  if(Bflag == 1){
+    rmsrSL = sqrt(mean((B - BstarSL)^2))
+    rmsrFR = sqrt(mean((B - BstarFR)^2))
+   }
+
+  list(B = B,
+       BstarSL = BstarSL,
+       BstarFR = BstarFR,
+       rmsrSL = rmsrSL,
+       rmsrFR = rmsrFR
+       )
+ } ## END BiFAD

> ###############################################################
> ##     Utility Functions
> ###############################################################
>
> MatchBiFactors <- function(A, B){
+   ## Match (and reorder) the factors of B to A
+   ## using Tucker congruence coefficients (cosines)
+   ## This function assumes that A and B are
+   ## bifactor loadings matrices
+   nc <- ncol(A)
+   F1 <- A[,2:nc]
+   F2 <- B[,2:nc]
+   D.F1<-diag(1/sqrt(apply(F1^2,2,sum)))
+   D.F2<-diag(1/sqrt(apply(F2^2,2,sum)))
+   cosF1F2<-D.F1 %*% t(F1) %*% F2 %*% D.F2
+   best<-apply(abs(cosF1F2),1,which.max)
+
+   ## find factor orientation
+   signvec<-rep(0,(nc-1))
+
+   for(i in 1:(nc-1)){
+     signvec[i]<- sign(cosF1F2[i,best[i]])
+   }
+
+   f1sign<- -1
+   if(sign(sum(A[,1]))==sign(sum(B[,1]))) f1sign <- 1
+
+   B[,c(1,best+1)] %*% diag(c(f1sign,signvec))
+ }## END MatchFactors
> ##
> ## Compute root mean squared residual: RMSR
> RMSR <- function(F1, F2) sqrt(mean((F1 - F2)^2))
> ##
> ################################################################
> ## EXAMPLES
>
> example <- 2
> if(example == 1){
+ # Mansolf and Reise Table 2 Example
+  Btrue<- matrix(c(.48, .40,  0,   0,   0,
+                   .51, .35,  0,   0,   0,
+                   .67, .62,  0,   0,   0,
+                   .34, .55,  0,   0,   0,
+                   .44,  0, .45,   0,   0,
+                   .40,  0, .48,   0,   0,
+                   .32,  0, .70,   0,   0,
+                   .45,  0, .54,   0,   0,
+                   .55,  0,   0, .43,   0,
+                   .33,  0,   0, .33,   0,
+                   .52,  0,   0, .51,   0,
+                   .35,  0,   0, .69,   0,
+                   .32,  0,   0,   0, .65,
+                   .66,  0,   0,   0, .51,
+                   .68,  0,   0,   0, .39,
+                   .32,  0,   0,   0, .56), 16, 5, byrow=TRUE)
+
+  R <-Btrue %*% t(Btrue)
+  diag(R) <- 1
+ }
> if(example == 2){
+ # Mansolf and Reise Table 3 Example (non cluster structure)
+  Btrue<- matrix(c(
+    .5,   .44,    .50,   .00,  .00,
+    .5,   .36,    .00,   .00,  .00,
+    .5,   .44,    .00,   .00,  .00,
+    .5,   .31,    .00,   .00,  .00,
+    .5,   .00,    .37,   .40,  .30,
+    .5,   .00,    .43,   .00,  .00,
+    .5,   .00,    .65,   .00,  .00,
+    .5,   .00,    .33,   .00,  .00,
+    .5,   .00,    .30,   .30,  .00,
+    .5,   .00,    .30,   .49,  .00,
+    .5,   .00,    .00,   .32,  .00,
+    .5,   .00,    .00,   .52,  .00,
+    .5,   .00,    .00,   .00,  .54,
+    .5,   .00,    .00,   .00,  .50,
+    .5,   .00,    .00,   .00,  .31,
+    .5,   .00,    .00,   .00,  .68), 16, 5, byrow=TRUE)
+
+  R <-Btrue %*% t(Btrue)
+  diag(R) <- 1
+ }
> ## Beginning of For loop
>  thresholds <- seq(.6, .05, by = -.05)
>  rmsrVecSL <- rmsrVecFR <- rep(0, length(thresholds))
> for( i in 1: length(thresholds)){
+   cat("\nWorking on ", i)
+   out<-BiFAD(R, B=NULL,
+                   nGroup=4,
+                   factorMethod = "minres",
+                   rotation="oblimin",
+                   salient = thresholds[i],
+                   maxitFA = 5000,
+                   maxitRotate = 5000,
+                   gamma = 0)
+   ## sort factors to match order in Btrue
+   BstarSL <- MatchBiFactors(Btrue, out$BstarSL)
+   BstarFR <- MatchBiFactors(Btrue, out$BstarFR)
+   ## Compute RMSR
+   rmsrVecSL[i] <- RMSR(F1=Btrue, F2=BstarSL)
+   rmsrVecFR[i] <- RMSR(F1=Btrue, F2=BstarFR)
+ }
```

```
Working on  1
Working on  2
Working on  3
Working on  4
Working on  5
Working on  6
Working on  7
Working on  8
Working on  9
Working on  10
Working on  11
Working on  12
```

```
> plot(thresholds, rmsrVecSL,
+      type="l",
+      ylab="RMSR (True Model  -  Estimated Model)",
+      xlab="Thresholds for Dichotomized Factor Loadings",
+      ylim=c(.0, .08),
+      cex.lab=1,
+      cex.axis=1,
+      lwd=2)
>  points(thresholds, rmsrVecFR,
+      type="l", lty=5,lwd = 5, col="darkgrey")
> SL<-psych::schmid(R, nfactors=4, rotate="oblimin")$sl[,1:5]
> SL <-MatchBiFactors(A=Btrue, B=SL)
> RMSR(F1=Btrue, F2=SL)
```

```
[1] 0.06027858
```

```
> abline(h=RMSR(F1=Btrue, F2=SL), lty=3, lwd=3)
>  if(example==1){
+    yval=.08
+  } else{
+    yval = .015
+  }
> ## Best fit solution
> out<-BiFAD(R, B=Btrue,
+               nGroup=NULL,
+               factorMethod = "minres",
+                 rotation="oblimin",
+                 salient = thresholds[i],
+                 maxitFA = 5000,
+                 maxitRotate = 5000,
+                 gamma = 0)
> abline(h=out$rmsrSL, lty=4, lwd=3)
> ## S-L:  standard Schmid-Leiman from the psych package
> ## Direct S-L (E): Target = empirically defined
> ## Direct S-L (T): Target = True Model
> legend(x=.42,
+        y=yval,
+        lty=c(3,1, 4, 5),
+        lwd=c(3,3,3,5),
+        col=c("black", "black", "black", "darkgrey"),
+        legend=c("S-L",
+                 "Direct S-L (E)",
+                 "Direct S-L (T)",
+                 "Direct BF (E)"),
+         cex=.9)
```



Direct Bifactor Rotations

In this section I demonstrate how to conduct an exploratory Direct Bifactor rotation. This method will produce a full rank loadings matrix.

```
> ## Mansolf and Reise Table 3 Example (non cluster structure)
>   Btrue<- matrix(c(
+     .5,   .44,    .50,   .00,  .00,
+     .5,   .36,    .00,   .00,  .00,
+     .5,   .44,    .00,   .00,  .00,
+     .5,   .31,    .00,   .00,  .00,
+     .5,   .00,    .37,   .40,  .30,
+     .5,   .00,    .43,   .00,  .00,
+     .5,   .00,    .65,   .00,  .00,
+     .5,   .00,    .33,   .00,  .00,
+     .5,   .00,    .30,   .30,  .00,
+     .5,   .00,    .30,   .49,  .00,
+     .5,   .00,    .00,   .32,  .00,
+     .5,   .00,    .00,   .52,  .00,
+     .5,   .00,    .00,   .00,  .54,
+     .5,   .00,    .00,   .00,  .50,
+     .5,   .00,    .00,   .00,  .31,
+     .5,   .00,    .00,   .00,  .68), 16, 5, byrow=TRUE)
>   R <-Btrue %*% t(Btrue)
>   diag(R) <- 1
```

Compute a full rank non-hierarchical bifactor solution with an empirically generated target matrix.

```
>   out<-BiFAD(R, B=NULL,
+                nGroup=4,
+                factorMethod = "minres",
+                rotation="oblimin",
+                salient = .25,
+                maxitFA = 5000,
+                maxitRotate = 5000,
+                gamma = 0)
>   DirectBiFac <-MatchBiFactors(A=Btrue, B=out$BstarFR)
>   print( round(DirectBiFac,2 ))
```

```
      [,1] [,2]  [,3]  [,4] [,5]
 [1,] 0.46 0.46  0.51  0.04 0.02
 [2,] 0.48 0.39  0.01  0.02 0.01
 [3,] 0.47 0.47  0.02  0.02 0.02
 [4,] 0.48 0.34  0.01  0.02 0.01
 [5,] 0.48 0.01  0.36  0.42 0.32
 [6,] 0.49 0.02  0.43  0.04 0.00
 [7,] 0.49 0.02  0.65  0.05 0.00
 [8,] 0.49 0.02  0.33  0.04 0.00
 [9,] 0.48 0.02  0.29  0.33 0.02
[10,] 0.47 0.02  0.29  0.52 0.03
[11,] 0.48 0.03 -0.01  0.34 0.02
[12,] 0.47 0.03 -0.01  0.54 0.03
[13,] 0.50 0.01  0.00  0.00 0.54
[14,] 0.50 0.01  0.00  0.00 0.50
[15,] 0.50 0.02  0.00  0.01 0.31
[16,] 0.50 0.01  0.00 -0.01 0.68
```

```
>   RMSR(Btrue, DirectBiFac)

[1] 0.01986072
```