Package ‘BifactorIndicesCalculator’

May 13, 2021

Type Package
Title Bifactor Indices Calculator
Version 0.2.2
Maintainer David Dueber <david.dueber@uky.edu>
Description The calculator computes bifactor indices such as explained common variance (ECV), hierarchical Omega (OmegaH), percentage of uncontaminated correlations (PUC), item explained common variance (I-ECV), and more. This package is an R version of the 'Excel' based 'Bifactor Indices Calculator' (Dueber, 2017) <doi:10.13023/edp.tool.01> with added convenience features for directly utilizing output from several programs that can fit confirmatory factor analysis or item response models.
License GPL (>= 3)
Encoding UTF-8
LazyData true
Depends R (>= 3.5)
Imports tidyr, lavaan, mirt, MplusAutomation, mnormt
RoxygenNote 7.1.0
URL https://github.com/ddueber/BifactorIndicesCalculator
BugReports https://github.com/ddueber/BifactorIndicesCalculator/issues
Suggests testthat (>= 2.1.0), psych
NeedsCompilation no
Author David Dueber [aut, cre]
Repository CRAN
Date/Publication 2021-05-12 23:02:36 UTC

R topics documented:

ARPB ................................................................. 2
bifactorIndices ..................................................... 3
ARPB computes absolute relative bias in factor loadings between the general factor of a bifactor model and a unidimensional model.

Usage

ARPB(Lambda, UniLambda)

Arguments

Lambda is a matrix of factor loadings
UniLambda is a matrix of factor loadings

Details

ARPB is called by bifactorIndices and the various convenience functions for exploratory models and/or Mplus output, which are the only functions in this package intended for casual users.
**Value**

A list where the first element is the average absolute relative parameter bias, and the second element is a vector of absolute relative bias by item.

**See Also**

`bifactorIndices`

**Examples**

```r
Lambda <- matrix(c(.82, .10, 0, 0,
                   .77, .35, 0, 0,
                   .79, .32, 0, 0,
                   .66, .39, 0, 0,
                   .51, 0, .71, 0,
                   .56, 0, .43, 0,
                   .68, 0, .13, 0,
                   .60, 0, .50, 0,
                   .83, 0, 0, .47,
                   .60, 0, 0, .27,
                   .78, 0, 0, .28,
                   .55, 0, 0, .75),
                   ncol = 4, byrow = TRUE)
colnames(Lambda) <- c("General", "SF1", "SF2", "SF3")
UniLambda <- c(.78, .84, .82, .77, .69, .62, .69, .66, .82, .56, .74, .65)
ARPB(Lambda, UniLambda)
```

**Description**

Computes all available bifactor indices for the input given.

**Usage**

```r
bifactorIndices(
  Lambda,
  Theta = NULL,
  UniLambda = NULL,
  standardized = TRUE,
  Phi = NULL,
  Thresh = NULL
)
```
Arguments

Lambda is a matrix of factor loadings or an object that can be converted to a matrix of factor loadings by `getLambda`. Currently fitted lavaan objects and fitted mirt objects are supported in addition to raw factor loading matrix input. For Mplus output files, use `bifactorIndicesMplus`.

Theta is a vector of residual variances. If omitted, Theta will be computed from input for Lambda.

UniLambda is a matrix of factor loadings or an object that can be converted to a matrix of factor loadings such as a fitted lavaan objects or fitted mirt object. Defaults to NULL, as UniLambda is only required if you wish to compute ARPB.

standardized lets the function know whether to look for standardized or unstandardized results from lavaan and defaults to TRUE. If Lambda is not a lavaan object, then standardized will be ignored.

Phi is the correlation matrix of factors and defaults to NULL. User should generally ignore this parameter. If not provided, bifactorIndices will try to determine Phi from Lambda when Lambda is a fitted lavaan model or will assume it is the identity matrix otherwise.

Thresh is a list of vectors of item thresholds, used only when items are categorical. bifactorIndices will try to determine Thresh from Lambda when Lambda is a fitted lavaan model and the indicators are categorical. Thresh defaults to null, which indicates items are continuous.

Details

Currently, factor loading matrices, fitted lavaan objects, and fitted mirt objects are supported. For Mplus output, see `bifactorIndicesMplus`. IRT parameters from mirt are converted to standardized factor loadings via the correspondence described in Kamata & Bauer (2008). If you wish to use standardized coefficients, item error variance will be computed directly from standardized factor loadings. ARPB will only be computed if the factor loadings from a unidimensional model are included, while ECV_GS and ECV_SG will only be computed for models with a general factor, and PUC will only be computed for a true bifactor model. Note that if a correlated traits model is provided, the omega indices will simply be the regular omega values for those factors. Interpretations for individual indices as well as details about their computation can be found in the man page for the individual indices.

Formulas for all indices can be found in Rodriguez et al. (2016). When indicators are categorical, the methodology of Green and Yang (2009) is used for computing Omega and OmegaH.

Value

A list of bifactor indices, including three different ECV indices, IECV, PUC, Omega, OmegaH, Factor Determinacy (FD), Construct Replicability (H) and ARPB. Please note that many of these indices are interpretable even when the model being used is not a bifactor model; some indices may be useful for two-tier, trifactor, correlated traits, and even unidimensional models.
References


See Also

bifactorIndicesMplus, bifactorIndicesMplus_expl, bifactorIndicesMplus_ESEM, ECV_SS, ECV_SG, ECV_GS, IECV, PUC, Omega_S, Omega_H, cat_Omega_S, cat_Omega_H, H, FD, ARPB

Examples

# Computing bifactor indices from fitted lavaan object
# (using mirt object is similar). Use of the unidimensional
# model is optional; it is only used to compute ARPB.

SRS_UnidimensionalModel <-
"SRS =~ SRS_1 + SRS_2 + SRS_3 + SRS_4 + SRS_5 +
SRS_6 + SRS_7 + SRS_8 + SRS_9 + SRS_10 +
SRS_11 + SRS_12 + SRS_13 + SRS_14 + SRS_15 +
SRS_16 + SRS_17 + SRS_18 + SRS_19 + SRS_20"

SRS_Unidimensional <- lavaan::cfa(SRS_UnidimensionalModel,
SRS_data,
ordered = paste0("SRS_", 1:20),
orthogonal = TRUE)

SRS_BifactorModel <-
"SRS =~ SRS_1 + SRS_2 + SRS_3 + SRS_4 + SRS_5 +
SRS_6 + SRS_7 + SRS_8 + SRS_9 + SRS_10 +
SRS_11 + SRS_12 + SRS_13 + SRS_14 + SRS_15 +
SRS_16 + SRS_17 + SRS_18 + SRS_19 + SRS_20
Function =~ SRS_5 + SRS_9 + SRS_12 + SRS_15 + SRS_18
Pain =~ SRS_1 + SRS_2 + SRS_8 + SRS_11 + SRS_17
SelfImage =~ SRS_4 + SRS_6 + SRS_10 + SRS_14 + SRS_19
MentalHealth =~ SRS_3 + SRS_7 + SRS_13 + SRS_16 + SRS_20"

SRS_bifactor <- lavaan::cfa(SRS_BifactorModel,
SRS_data,
ordered = paste0("SRS_", 1:20),
orthogonal = TRUE)

bifactorIndices(SRS_bifactor, UniLambda = SRS_Unidimensional)
# Computing bifactor indices from standardized factor loading matrices

Lambda <- matrix(c(.82, .10, 0, 0,
                  .77, .35, 0, 0,
                  .79, .32, 0, 0,
                  .66, .39, 0, 0,
                  .51, 0, .71, 0,
                  .56, 0, .43, 0,
                  .68, 0, .13, 0,
                  .60, 0, .50, 0,
                  .83, 0, 0, .47,
                  .60, 0, 0, .27,
                  .78, 0, 0, .28,
                  .55, 0, 0, .75),
                  ncol = 4, byrow = TRUE)
colnames(Lambda) <- c("General", "SF1", "SF2", "SF3")
bifactorIndices(Lambda)

# bifactorIndices can also be used on two-tier models

MTMM_model <- "
Trait1 =~ T1M_1 + T1M_2 + T1M_3 +
         T1M_1 + T1M_2 + T1M_3 +
         T1M_1 + T1M_2 + T1M_3 
Trait2 =~ T2M_1 + T2M_2 + T2M_3 +
         T2M_1 + T2M_2 + T2M_3 +
         T2M_1 + T2M_2 + T2M_3 
Trait3 =~ T3M_1 + T3M_2 + T3M_3 +
         T3M_1 + T3M_2 + T3M_3 +
         T3M_1 + T3M_2 + T3M_3 
Method1 =~ T1M_1 + T1M_2 + T1M_3 +
         T1M_1 + T1M_2 + T1M_3 +
         T1M_1 + T1M_2 + T1M_3 
Method2 =~ T2M_1 + T2M_2 + T2M_3 +
         T2M_1 + T2M_2 + T2M_3 +
         T2M_1 + T2M_2 + T2M_3 
Method3 =~ T3M_1 + T3M_2 + T3M_3 +
         T3M_1 + T3M_2 + T3M_3 +
         T3M_1 + T3M_2 + T3M_3 

Trait1 ~~ 0*Method1
Trait1 ~~ 0*Method2
Trait1 ~~ 0*Method3
Trait2 ~~ 0*Method1
Trait2 ~~ 0*Method2
Trait2 ~~ 0*Method3
Trait3 ~~ 0*Method1
Trait3 ~~ 0*Method2
Trait3 ~~ 0*Method3
Method1 ~~ 0*Method2
BifactorIndicesCalculator

**BifactorIndicesCalculator**: A package for computing statistical indices relevant to bifactor measurement models.

### Description

The BifactorIndicesCalculator package provides functions use to compute indices described in Rodriguez et al. (2016) for confirmatory or exploratory models.

### BifactorIndicesCalculator functions

The BifactorIndicesCalculator package includes helper functions that can compute the various indices from Rodriguez et al. (2016) for fitted lavaan or mirt objects using `bifactorIndices` and Mplus output files using `bifactorIndicesMplus`. For users of other software and those interested, a matrix of standardized factor loadings can also be input to `bifactorIndices`.

Calculation of bifactor indices for exploratory bifactor models can also be accomplished using `bifactorIndices_expl` for exploratory models fit using psych or using `bifactorIndicesMplus_expl` for exploratory models fit using Mplus. For ESEM models fit using Mplus, please use `bifactorIndicesMplus_ESEM`.

### References


### Usage

```r
bifactorIndicesMplus

Lambda = file.choose(),
UniLambda = NULL,
standardized = TRUE
```

Computes all available bifactor indices given an Mplus .out file for a bifactor model.
**Arguments**

- **Lambda**: is an Mplus .out file. Defaults to an open file dialog box.
- **UniLambda**: is an object that the function can convert to a matrix of factor loadings. The expected behavior is to store an Mplus output file as a variable and pass that variable as UniLambda. Defaults to NULL, as UniLambda is only required if you wish to compute ARPB.
- **standardized**: lets the function know whether it should be looking in the unstandardized results or the STDYX results from the Mplus output.

**Details**

To use this function, simply call it without any arguments and a dialog box will pop up for you to select a .out file of a confirmatory bifactor model.

ARPB will only be computed if the factor loadings from a unidimensional model (as a vector or as the result of using `readModels` on an Mplus .out file) are included. Note that if a correlated traits model is provided, the omega indices will simply be the regular omega values for those factors. Interpretations for individual indices as well as details about their computation can be found in the man page for the individual indices.

**Value**

A list of bifactor indices, including three different ECV indices, IECV, PUC, Omega, OmegaH, and ARPB. Please note that many of these indices are interpretable even when the model being used is not a bifactor model; some indices may be useful for two-tier, trifactor, correlated traits, and even unidimensional models.

**See Also**

- `bifactorIndices`
- `bifactorIndices_expl`
- `bifactorIndicesMplus_expl`
- `bifactorIndicesMplus_ESEM`
- `ECV_SS`, `ECV_SG`, `ECV_GS`, `IECV`, `PUC`, `Omega_S`, `Omega_H`, `H`, `FD`, `ARPB`

**Description**

Computes all available bifactor indices given an Mplus .out file for a bifactor ESEM

**Usage**

```r
bifactorIndicesMplus_ESEM(
  Lambda = file.choose(),
  ItemsBySF = NULL,
  LoadMin = 0.2
)
```
**Arguments**

- **Lambda**
  is an Mplus .out file. Defaults to an open file dialog box.

- **ItemsBySF**
  is a list, indexed by factor, of vectors of item names belonging to each factor. You must NOT include the general factor in this list, and the list must have names which match the factor names in Mplus. Defaults to NULL, in which case composition of specific factors is automated by comparing loadings to `LoadMin`.

- **LoadMin**
  is the minimum loading size so that an item is considered to "belong" to a factor. If `ItemsBySF` is not provided, then items are assigned to factors based on whether their loading on that factor is greater than `LoadMin`. If `ItemsBySF` is provided, then warnings are issued whenever items load above `LoadMin` on factors to which they do not belong, or do not load above `LoadMin` on factors to which they do belong.

**Details**

To use this function, simply call it without any arguments and a dialog box will pop up for you to select a .out file for an ESEM model.

Only standardized models are considered for exploratory models. PUC and ARPB are not supported for exploratory models currently, although that may change.

**Value**

A list of bifactor indices, including three different ECV indices, Omega, and OmegaH.

**See Also**

- `bifactorIndices`, `bifactorIndicesMplus`, `bifactorIndicesMplus_expl`, `bifactorIndices_expl`
Arguments

Lambda is an Mplus .out file. Defaults to an open file dialog box.

ItemsBySF is a list, indexed by factor, of vectors of item names belonging to each factor. You must include the general factor in this list, and the list must have names which match the factor names in Mplus. Defaults to NULL, in which case composition of specific factors in automated by comparing loadings to LoadMin.

LoadMin is the minimum loading size so that an item is considered to "belong" to a factor. If ItemsBySF is not provided, then items are assigned to factors based on whether their loading on that factor is greater than LoadMin. If ItemsBySF is provided, then warnings are issued whenever items load above LoadMin on factors to which they do not belong, or do not load above LoadMin on factors to which they do belong.

Details

To use this function, simply call it without any arguments and a dialog box will pop up for you to select a .out file of an exploratory bifactor model.

EFA models are not currently (3/3/2020) supported by MplusAutomation::ReadModels(), but they will be in the very near future, at which time this function will be completed.

Value

A list of bifactor indices, including three different ECV indices, Omega, and OmegaH.

See Also

bifactorIndices, bifactorIndicesMplus, bifactorIndicesMplus_ESEM, bifactorIndices_expl

Description

Computes all available bifactor indices for the input given.

Usage

bifactorIndices_expl(Lambda, ItemsBySF = NULL, LoadMin = 0.2)

Arguments

Lambda is a factor loading matrix from EFA or an object which can be converted to such. Currently only psych::fa() objects are supported.
ItemsBySF is a list, indexed by factor, of vectors of item names belonging to each specific factor. You must NOT include the general factor in this list, and the list must have names which match the factor names in Lambda. It is recommended you look at the EFA solution first to see which factor is which. Defaults to NULL, in which case composition of specific factors is automated by comparing loadings to LoadMin.

LoadMin is the minimum loading size so that an item is considered to "belong" to a factor. If ItemsBySF is not provided, then items are assigned to factors based on whether their loading on that factor is greater than LoadMin. If ItemsBySF is provided, then warnings are issued whenever items load above LoadMin on factors to which they do not belong, or do not load above LoadMin on factors to which they do belong. LoadMin defaults to 0.2.

Details

Only standardized models are considered for exploratory models. PUC and ARPB are not supported for exploratory models currently, although that may change.

Value

A list of bifactor indices, including three different ECV indices, Omega, and OmegaH.

See Also

bifactorIndices, bifactorIndicesMplus, bifactorIndicesMplus_expl, bifactorIndicesMplus_ESEM, ECV_SS, ECV_SG, ECV_GS, IECV, Omega_S, Omega_H, H, FD

Examples

# psych::fa() can not access the rotations We have to load the library.
library(psych)
SRS_BEFA <- fa(SRS_data, nfactors = 5, rotate = "bifactor")

# inspect the solution to see which exploratory factors belong to which subdomain
SRS_BEFA$loadings
ItemsBySF = list(MR4 = paste0("SRS_", c(5, 9, 12, 15, 18)),
                 MR2 = paste0("SRS_", c(1, 2, 8, 11, 17)),
                 MR3 = paste0("SRS_", c(4, 6, 10, 14, 19)),
                 MR5 = paste0("SRS_", c(3, 7, 13, 16, 20)))

bifactorIndices_expl(SRS_BEFA, ItemsBySF = ItemsBySF)
cat_Omega_H

**Description**

Computes hierarchical omega reliability estimate for all factors as described in Rodriguez, Reise, and Haviland (2016).

**Usage**

```r
cat_Omega_H(Lambda, Thresh, Phi = NULL, Denom = NULL)
```

**Arguments**

- **Lambda**: is a matrix of standardized factor loadings
- **Thresh**: is a list (indexed by items) of vectors of item thresholds
- **Phi**: is the latent variable covariance matrix. Defaults to `NULL`, and the identity matrix will be used. No other options are currently available.
- **Denom**: specifies how the variance of the total score will be computed. Defaults to `NULL`, and the model implied total score variance will be used. No other options are currently available.

**Details**

cat_Omega_H is called by `bifactorIndices` and the various convenience functions for exploratory models and/or Mplus output, which are the only functions in this package intended for casual users.

**Value**

A numeric, the hierarchical omega reliability estimate for all factors using the technique of Green and Yang (2009).

**References**


**See Also**

`Omega_H`, `bifactorIndices`
Examples

```r
Lambda <- matrix(c(0.82, 0.10, 0, 0,
                    0.77, 0.35, 0, 0,
                    0.79, 0.32, 0, 0,
                    0.66, 0.39, 0, 0,
                    0.51, 0, 0.71, 0,
                    0.56, 0, 0.43, 0,
                    0.68, 0, 0.13, 0,
                    0.60, 0, 0.50, 0,
                    0.83, 0, 0.47, 0,
                    0.60, 0, 0, 0.27,
                    0.78, 0, 0, 0.28,
                    0.55, 0, 0, 0.75),
                  ncol = 4, byrow = TRUE)
colnames(Lambda) <- c("General", "SF1", "SF2", "SF3")

Thresh <- list(c(-1, 0, 1), c(-0.5, 0, 0.5),
                c(0, 1, 2), c(0, 0.5, 1),
                c(-2, -1, 0), c(-1, -0.5, 0),
                c(-1, 0, 2), c(-0.5, 0, 1),
                c(-2, 0, 1), c(-1, 0, 0.5),
                c(-1, 0, 1), c(-0.5, 0, 0.5))

cat_Omega_H(Lambda, Thresh)
```

Description

Computes an omega reliability estimate for all factors as described in Rodriguez, Reise, and Haviland (2016).

Usage

```r
cat_Omega_S(Lambda, Thresh, Phi = NULL, Denom = NULL)
```

Arguments

- **Lambda**: is a matrix of standardized factor loadings
- **Thresh**: is a list (indexed by items) of vectors of item thresholds (items must be on a standardized metric).
- **Phi**: is the latent variable covariance matrix. Defaults to NULL, and the identity matrix will be used. No other options are currently available.
- **Denom**: specifies how the variance of the total score will be computed. Defaults to NULL, and the model implied total score variance will be used. No other options are currently available.
Details

cat_Omega_S is called by `bifactorIndices` and the various convenience functions for exploratory models and/or Mplus output, which are the only functions in this package intended for casual users.

Value

A numeric, the omega reliability estimate for all factors using the technique of Green and Yang (2009).

References


See Also

`Omega_H, bifactorIndices`

Examples

```r
Lambda <- matrix(c(.82, .10, 0, 0, 
                   .77, .35, 0, 0, 
                   .79, .32, 0, 0, 
                   .66, .39, 0, 0, 
                   .51, 0, .71, 0, 
                   .56, 0, .43, 0, 
                   .68, 0, .13, 0, 
                   .60, 0, .50, 0, 
                   .83, 0, 0, .47, 
                   .60, 0, 0, .27, 
                   .78, 0, 0, .28, 
                   .55, 0, 0, .75),
ncol = 4, byrow = TRUE)
colnames(Lambda) <- c("General", "SF1", "SF2", "SF3")

Thresh = list(c(-1, 0, 1), c(-0.5, 0, 0.5), 
              c(0, 1, 2), c(0, 0.5, 1), 
              c(-2, -1, 0), c(-1, -0.5, 0), 
              c(-1, 0, 2), c(-0.5, 0, 1), 
              c(-2, 0, 1), c(-1, 0, 0.5), 
              c(-1, 0, 1), c(-0.5, 0, 0.5))

cat_Omega_S(Lambda, Thresh)
```
Description

Computes an ECV index for all factors which can be interpreted as the proportion of common variance of the items in each specific factor which is due to the general factor; \texttt{ECV\_GS} should be read 'ECV of the general factor with respect to a specific factor.' Here, ECV is computed only with respect to the items of a specific factor using the general factor loadings in the numerator; Stucky and Edelen (2015, p. 201) refer to this index as the 'within-domain ECV' for the specific factor. In the Excel version of the bifactor indices calculator (Dueber, 2017), this index is not computed.

Usage

\texttt{ECV\_GS}(\texttt{Lambda})

Arguments

\texttt{Lambda} is a matrix of factor loadings. Be sure that all factors have the same variance before calling this function.

Details

\texttt{ECV\_GS} is called by \texttt{bifactorIndices} and the various convenience functions for exploratory models and/or Mplus output, which are the only functions in this package intended for casual users.

Value

A vector of ECVs for all factors

References


See Also

\texttt{ECV\_SS, ECV\_SG, bifactorIndices}
ECV_SG

Examples

```r
Lambda <- matrix(c(.82, .10, 0, 0,
 .77, .35, 0, 0,
 .79, .32, 0, 0,
 .66, .39, 0, 0,
 .51, 0, .71, 0,
 .56, 0, .43, 0,
 .68, 0, .13, 0,
 .60, 0, .50, 0,
 .83, 0, 0, .47,
 .60, 0, 0, .27,
 .78, 0, 0, .28,
 .55, 0, 0, .75),
 ncol = 4, byrow = TRUE)
colnames(Lambda) <- c("General", "SF1", "SF2", "SF3")
ECV_SG(Lambda)
```

Description

Computes an ECV index for all factors which can be interpreted as the proportion of common variance of all items which is due to the specific factor; ECV_SG should be read 'ECV of a specific factor with respect to the general factor.' Here, ECV is computed with respect to the items of the general factor using the specific factor loadings in the numerator; Stucky and Edelen (2015, p. 199) refer to this index simply as 'specific-dimension ECV.' Note that ECV_SG of the general factor is simply the ECV. In the Excel version of the Bifactor Indices Calculator (Dueber, 2017), this form of ECV is referred to as 'ECV (S&E).’ ECV_SG is called by bifactorIndices and the various convenience functions for exploratory models and/or Mplus output, which are the only functions in this package intended for casual users.

Usage

`ECV_SG(Lambda)`

Arguments

- **Lambda** is a matrix of factor loadings. Be sure that all factors have the same variance before calling this function.

Value

A vector of ECVs for all factors
References


See Also

ECV_SS, ECV_GS, bifactorIndices

Examples

```r
Lambda <- matrix(c(.82, .10, 0, 0,
                   .77, .35, 0, 0,
                   .79, .32, 0, 0,
                   .66, .39, 0, 0,
                   .51, 0, .71, 0,
                   .56, 0, .43, 0,
                   .68, 0, .13, 0,
                   .60, 0, .50, 0,
                   .83, 0, 0, .47,
                   .60, 0, 0, .27,
                   .78, 0, 0, .28,
                   .55, 0, 0, .75),
                  ncol = 4, byrow = TRUE)
colnames(Lambda) <- c("General", "SF1", "SF2", "SF3")
ECV_GS(Lambda)
```

<table>
<thead>
<tr>
<th>ECV_SS</th>
<th>ECV_SS</th>
</tr>
</thead>
</table>

Description

Computes an ECV index for all factors which can be interpreted as the proportion of common variance of the items in each factor which is due to that factor; ECV_SS should be read 'ECV of a specific factor with respect to itself.' Here, ECV is computed only with respect to items which load on the factor. Note that ECV_SS of the general factor is simply the ECV. Stucky and Edelen (2015, p. 201) do not refer to this form of ECV. In the Excel version of the bifactor indices calculator (Dueber, 2017), this index is referred to as 'ECV (NEW).’ ECV_SS is useful in that it can be computed when there is no general factor, such as in a two-tier model, and interpreted in the same way as ECV for general factors.

Usage

ECV_SS(Lambda)
Arguments

Lambda is a matrix of factor loadings. Be sure that all factors have the same variance before calling this function.

Details

ECV_SS is called by bifactorIndices and the various convenience functions for exploratory models and/or Mplus output, which are the only functions in this package intended for casual users.

Value

A vector of ECVs for all factors

References


See Also

ECV_SG, ECV_GS, bifactorIndices

Examples

```r
Lambda <- matrix(c(.82, .10, 0, 0,
                    .77, .35, 0, 0,
                    .79, .32, 0, 0,
                    .66, .39, 0, 0,
                    .51, 0, .71, 0,
                    .56, 0, .43, 0,
                    .68, 0, .13, 0,
                    .60, 0, .50, 0,
                    .83, 0, 0, .47,
                    .60, 0, 0, .27,
                    .78, 0, 0, .28,
                    .55, 0, 0, .75),
                   ncol = 4, byrow = TRUE)
colnames(Lambda) <- c("General", "SF1", "SF2", "SF3")
ECV_SS(Lambda)
```
Factor Determinacy

Description

FD computes factor determinacies for all factors provided standardized factor loadings and an inter-factor correlation matrix.

Usage

FD(Lambda, Phi)

Arguments

Lambda is a matrix of standardized factor loadings
Phi is the matrix of factor intercorrelations. For bifactor models Phi is diagonal with ones on the diagonal.

Details

FD is called by bifactorIndices and the various convenience functions for exploratory models and/or Mplus output, which are the only functions in this package intended for casual users.

Value

a vector of factor determinacies.

See Also

bifactorIndices

Examples

Lambda <- matrix(c(.82, .10, 0, 0,
                   .77, .35, 0, 0,
                   .79, .32, 0, 0,
                   .66, .39, 0, 0,
                   .51, 0, .71, 0,
                   .56, 0, .43, 0,
                   .68, 0, .13, 0,
                   .60, 0, .50, 0,
                   .83, 0, 0, .47,
                   .60, 0, 0, .27,
                   .78, 0, 0, .28,
                   .55, 0, 0, .75),
                   ncol = 4, byrow = TRUE)
Phi <- matrix(c(1, 0, 0, 0,
               0, 1, 0, 0,
               0, 0, 1, 0,
               0, 0, 0, 1),
               ncol = 4, byrow = TRUE)
### getGen

getGen detects whether or not a single factor loads on all items, and returns the column index of the general factor if it exists.

**Usage**

```
getGen(Lambda)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lambda</td>
<td>is a factor loading matrix</td>
</tr>
</tbody>
</table>

**Value**

The index of the general factor, or `NULL` if there is no general factor.

### getLambda

getLambda computes or extracts a matrix of factor loadings given some input. Methods exist to support an input of a dataframe, an `mplus.model` from `MplusAutomation`, a `SingleGroupClass` object from `mirt`, and a `lavaan` object from `lavaan`. Please do not use a tibble, as they do not support row names, and it is best if your items are given names.

**Usage**

```
getLambda(x, standardized = TRUE)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>an object to be converted into a factor loading matrix, or an object containing a fitted model from which a factor loading matrix will be extracted. Supported classes are <code>data.frame</code>, <code>matrix</code>, <code>mplus.model</code>, <code>lavaan</code>, and <code>SingleGroupClass</code>.</td>
</tr>
<tr>
<td>standardized</td>
<td>can be used to specify whether a standardized or unstandardized factor loading matrix should be returned. Only relevant for <code>lavaan</code> and <code>mplus.model</code> input. The standardized matrix for <code>mplus.model</code> is taken from stdyx results.</td>
</tr>
</tbody>
</table>
getTheta

Description

getTheta extracts or computes a vector of residual variance for items. If a factor loading matrix is provided, then the vector of residual variances is computed from that matrix if standardized is TRUE.

Usage

getTheta(x, standardized = TRUE)

Arguments

x an object that can be converted into a factor loading matrix, or an object containing a fitted model from which a vector of residual variances can be extracted. Supported classes are data.frame, matrix, mplus.model, lavaan, and SingleGroupClass

standardized can be used to specify whether a standardized or unstandardized factor loading matrix should be returned. Only relevant for lavaan and mplus.model input. The standardized matrix for mplus.model is taken from stdyx results.

Value

a vector of residual variances for items. If x is a fitted model, then the residual variances are extracted from the fitted model. lavaan, mirt (SingleGroupClass), and Mplus (mplus.model) models are supported. If Mplus does not report residual variances for categorical variables, then factor loadings are used to compute the residual variance for standardized models and an error is thrown for unstandardized models. In both cases, the user is alerted that residual variances could not be found in the input and perhaps the model should be rerun.

See Also

getLambda
**H**

*Construct Replicability*

**Description**

H computes construct replicability for all factors given standardized factor loadings.

**Usage**

H(Lambda)

**Arguments**

Lambda is a matrix of standardized factor loadings

**Details**

H is called by bifactorIndices and the various convenience functions for exploratory models and/or Mplus output, which are the only functions in this package intended for casual users.

**Value**

a vector of construct reliabilities.

**See Also**

bifactorIndices

**Examples**

```
Lambda <- matrix(c(.82, .10, 0, 0,
                   .77, .35, 0, 0,
                   .79, .32, 0, 0,
                   .66, .39, 0, 0,
                   .51, 0, .71, 0,
                   .56, 0, .43, 0,
                   .68, 0, .13, 0,
                   .60, 0, .50, 0,
                   .83, 0, 0, .47,
                   .60, 0, 0, .27,
                   .78, 0, 0, .28,
                   .55, 0, 0, .75),
                   ncol = 4, byrow = TRUE)
colnames(Lambda) <- c("General", "SF1", "SF2", "SF3")
H(Lambda)
```
**Description**

Computes an ECV index for each item which can be interpreted as the proportion of common variance of that item due to the general factor. Stucky and Edelen (2015, p. 201) define I-ECV, which is also computed in the Excel version of the bifactor indices calculator (Dueber, 2017).

**Usage**

IECV(Lambda)

**Arguments**

Lambda is a matrix of factor loadings. Be sure that all factors have the same variance before calling this function.

**Details**

IECV is called by bifactorIndices and the various convenience functions for exploratory models and/or Mplus output, which are the only functions in this package intended for casual users.

**Value**

A vector of item ECVs

**References**


**See Also**

ECV_SS, ECV_SG, ECV_GS, bifactorIndices

**Examples**

```r
Lambda <- matrix(c(.82, .10, 0, 0,
                   .77, .35, 0, 0,
                   .79, .32, 0, 0,
                   .66, .39, 0, 0,
                   .51, 0, .71, 0,
                   .56, 0, .43, 0,
                   ), nrow=3)
```
isBifactor <- function(Lambda) {
  ncol <- ncol(Lambda)
  for (i in 1:ncol) {
    if (colnames(Lambda)[i] != "General") {
      return(FALSE)
    }
    for (j in 1:ncol) {
      if (j != i && colnames(Lambda)[j] != "SF1" && colnames(Lambda)[j] != "SF2" && colnames(Lambda)[j] != "SF3") {
        return(FALSE)
      }
    }
  }
  return(TRUE)
}

Description
Determines whether a model has bifactor structure.

Usage
isBifactor(Lambda)

Arguments
Lambda
Matrix of factor loadings

Value
Logical. If each item loads on a general factor and at most one specific factor, returns TRUE. Otherwise FALSE.

MTMM_data
Simulated data for multi-trait multi-method

Description
A dataset containing 500 responses to a hypothetical 27 item instrument measuring three traits from three different sources. Each trait is measured by 3 items collected from each of three sources (methods).

Usage
MTMM_data

Format
A data frame with 500 rows and 27 columns The item names indicate the trait number, method number, and item number
Description
Computes hierarchical omega reliability estimate for all factors as described in Rodriguez, Reise, and Haviland (2016).

Usage
Omega_H(Lambda, Theta)

Arguments
Lambda is a matrix of factor loadings
Theta is a vector of indicator error variances

Details
Omega_H is called by bifactorIndices and the various convenience functions for exploratory models and/or Mplus output, which are the only functions in this package intended for casual users.

Value
A numeric, the omega reliability estimate for all factors.

References

See Also
Omega_S, bifactorIndices

Examples
Lambda <- matrix(c(.82, .10, 0, 0,
                   .77, .35, 0, 0,
                   .79, .32, 0, 0,
                   .66, .39, 0, 0,
                   .51, 0, .71, 0,
                   .56, 0, .43, 0,
                   .68, 0, .13, 0,
                   .60, 0, .50, 0,
                   .83, 0, 0, .47,
                   .60, 0, 0, .27,
                   .78, 0, 0, .28,
                   .55, 0, 0, .75),
Description

Computes an omega reliability estimate for all factors as described in Rodriguez, Reise, and Haviland (2016).

Usage

Omega_S(Lambda, Theta)

Arguments

Lambda is a matrix of factor loadings
Theta is a vector of indicator error variances

Details

Omega_S is called by bifactorIndices and the various convenience functions for exploratory models and/or Mplus output, which are the only functions in this package intended for casual users.

Value

A numeric, the omega reliability estimate for all factors.

References


See Also

Omega_H, bifactorIndices
Examples

```r
Lambda <- matrix(c(.82, .10, 0, 0,  
                   .77, .35, 0, 0,  
                   .79, .32, 0, 0,  
                   .66, .39, 0, 0,  
                   .51, 0, .71, 0,  
                   .56, 0, .43, 0,  
                   .68, 0, .13, 0,  
                   .60, 0, .50, 0,  
                   .83, 0, 0, .47,  
                   .60, 0, 0, .27,  
                   .78, 0, 0, .28,  
                   .55, 0, 0, .75),  
                   ncol = 4, byrow = TRUE)

colnames(Lambda) <- c("General", "SF1", "SF2", "SF3")
Theta <- rep(1, nrow(Lambda)) - rowSums(Lambda^2)
Omega_S(Lambda, Theta)
```

Description

PUC computes the proportion of uncontaminated correlations for a bifactor mode

Usage

```r
PUC(Lambda)
```

Arguments

Lambda is a matrix of factor loadings

Details

PUC is called by `bifactorIndices` and the various convenience functions for exploratory models and/or Mplus output, which are the only functions in this package intended for casual users.

Value

numeric

See Also

`bifactorIndices`
Examples

Lambda <- matrix(c(.82, .10, 0, 0,
                   .77, .35, 0, 0,
                   .79, .32, 0, 0,
                   .66, .39, 0, 0,
                   .51, 0, .71, 0,
                   .56, 0, .43, 0,
                   .68, 0, .13, 0,
                   .60, 0, .50, 0,
                   .83, 0, 0, .47,
                   .60, 0, 0, .27,
                   .78, 0, 0, .28,
                   .55, 0, 0, .75),
                   ncol = 4, byrow = TRUE)

colnames(Lambda) <- c("General", "SF1", "SF2", "SF3")
PUC(Lambda)

SRS_data

Response Data to the SRS-22r

Description

A dataset containing 500 responses to the 20 item SRS-22r scoliosis quality of life outcome measure.

Usage

SRS_data

Format

A data frame with 500 rows and 20 columns. Row names indicate the item number.

Details

The Function subscale consists of items 5, 9, 12, 15, and 18. The Pain subscale consists of items 1, 2, 8, 11, and 17. The SelfImage subscale consists of items 4, 6, 10, 14, and 19. The MentalHealth subscale consists of items 3, 7, 13, 16, and 20.
Index

* datasets
  MTMM_data, 24
  SRS_data, 28
ARPB, 2, 4, 5, 8
bifactorIndices, 2, 3, 3–12, 14–19, 22, 23, 25–27
bifactorIndices_expl, 5, 7–10, 10
BifactorIndicesCalculator, 7
bifactorIndicesMplus, 4, 5, 7, 7, 9–11
bifactorIndicesMplus_ESEM, 5, 7, 8, 8, 10, 11
bifactorIndicesMplus_expl, 5, 7–9, 9, 11

  cat_Omega_H, 5, 12
  cat_Omega_S, 5, 13

  ECV_GS, 4, 5, 8, 11, 15, 17, 18, 23
  ECV_SG, 4, 5, 8, 11, 15, 16, 18, 23
  ECV_SS, 5, 8, 11, 15, 17, 17, 23

FD, 5, 8, 11, 19
getGen, 20
getLambda, 4, 20, 21
getTheta, 21

H, 5, 8, 11, 22
IECV, 5, 8, 11, 23
isBifactor, 24

MTMM_data, 24
Omega_H, 5, 8, 11, 12, 14, 25, 26
Omega_S, 5, 8, 11, 25, 26
PUC, 4, 5, 8, 27
readModels, 8
SRS_data, 28