Package ‘FactorAssumptions’

March 6, 2020

Type Package

Title Set of Assumptions for Factor and Principal Component Analysis

Version 1.1.2

Maintainer Jose Storopoli <thestoropoli@gmail.com>

Description Tests for Kaiser-Meyer-Olkin (KMO) and communalities in a dataset. It provides a final sample by removing variables in a iterable manner while keeping account of the variables that were removed in each step. It follows the best practices and assumptions according to Hair, Black, Babin & Anderson (2018, ISBN:9781473756540).

License GPL-3

URL https://github.com/storopoli/FactorAssumptions

BugReports https://github.com/storopoli/FactorAssumptions/issues

Depends R (>= 3.6.0), MASS, psych

Suggests knitr, rmarkdown, testthat (>= 2.1.0)

VignetteBuilder knitr

Encoding UTF-8

LazyData true

RoxygenNote 7.0.2

NeedsCompilation no

Author Jose Storopoli [aut, cre] (<https://orcid.org/0000-0002-0559-5176>)

Repository CRAN

Date/Publication 2020-03-06 17:10:02 UTC

R topics documented:

communalities_optimal_solution

Calculates the Optimal Solution for Communalities in a Dataframe

Description

communalities_optimal_solution() call upon either the principal or fa functions from psych package to iterate over the variables of a dataframe.

Usage

```
communalities_optimal_solution(
  df,
  nfactors,
  type,
  rotate = "varimax",
  fm = "minres",
  squared = TRUE
)
```

Arguments

df — a dataframe with only int or num type of variables

nfactors — number of factors to extract in principal components or factor analysis

type — either principal for *Principal Components Analysis* or fa for *Factor Analysis*

rotate — rotation to be employed (default is *varimax*). "none", "varimax", "quartimax", "bentlerT", "equamax", "varimin", "geominT" and "bifactor" are orthogonal rotations. "Promax", "promax", "oblimin", "simplimax", "bentlerQ, "geominQ" and "biquartimin" and "cluster" are possible oblique transformations of the solution. The default is to do a oblimin transformation, although versions prior to 2009 defaulted to varimax. SPSS seems to do a Kaiser normalization before doing Promax, this is done here by the call to "promax" which does the normalization before calling Promax in GPArotation.

fm — Factoring method fm="minres" (default) will do a minimum residual as will fm="uls". Both of these use a first derivative. fm="ols" differs very slightly from "minres" in that it minimizes the entire residual matrix using an OLS procedure but uses the empirical first derivative. This will be slower. fm="wls" will do a weighted least squares (WLS) solution, fm="gls" does a generalized weighted least squares (GLS), fm="pa" will do the principal factor solution, fm="ml" will do a maximum likelihood factor analysis. fm="minchi" will minimize the sample size weighted chi square when treating pairwise correlations with different number of subjects per pair. fm ="minrank" will do a minimum rank factor analysis. "old.min" will do minimal residual the way it was done prior to April, 2017 (see discussion below). fm="alpha" will do alpha factor analysis as described in Kaiser and Coffey (1965)

squared — TRUE if matrix is squared (such as adjacency matrices), FALSE otherwise

Details

If finds any individual communality below the optimal value of 0.5 then removes the lowest communality value variable until no more variable has not-optimal communality values.

Value

A list with

1. `df` - A dataframe that has reached its optimal solution in terms of KMO values
2. `removed` - A list of removed variables ordered by the first to last removed during the procedure
3. `loadings` - A table with the communalities loadings from the variables final iteration
4. `results` - Results of the final iteration of either the `principal` or `fa` functions from `psych` package

See Also

`principal` the PCA function from psych and `fa` the Factor Analysis function from psych

Examples

```
set.seed(123)
df <- as.data.frame(matrix(rnorm(100*10, 1, .5), ncol=10))
communalities_optimal_solution(df, nfactors = 2,type = "principal", squared = FALSE)
```

`kmo` *Calculates the Kayser-Meyer-Olkin (KMO)*

Description

`kmo()` handles both positive definite and not-positive definite matrix by employing the *Moore-Penrose* inverse (pseudoinverse)

Usage

```
kmo(x, squared = TRUE)
```

Arguments

`x`	a matrix or dataframe
`squared`	TRUE if matrix is squared (such as adjacency matrices), FALSE otherwise

Value

A list with

1. `overall` - Overall KMO value
2. `individual` - Individual KMO's dataframe
3. `AIS` - Anti-image Covariance Matrix
4. `AIR` - Anti-image Correlation Matrix

Examples

```
set.seed(123)
df <- as.data.frame(matrix(rnorm(100*10, 1, .5), ncol=10))
kmo(df, squared = FALSE)
```

`kmo_optimal_solution` *Calculates the Optimal Solution for Kayser-Meyer-Olkin (KMO) in a Dataframe*

Description

`kmo_optimal_solution()` call upon the `kmo` function to iterate over the variables of a dataframe.

Usage

```
kmo_optimal_solution(df, squared = TRUE)
```

Arguments

`df`	a dataframe with only `int` or `num` type of variables
`squared`	TRUE if matrix is squared (such as adjacency matrices), FALSE otherwise

Details

If finds any individual KMO's below the optimal value of 0.5 then removes the lowest KMO value variable until no more variable has not-optimal KMO values.

Value

A list with

1. `df` - A dataframe that has reached its optimal solution in terms of KMO values
2. `removed` - A list of removed variables ordened by the first to last removed during the procedure
3. `kmo_results` - Results of the final iteration of the `kmo` function

See Also

`kmo` for kmo computation function

Examples

```
set.seed(123)
df <- as.data.frame(matrix(rnorm(100*10, 1, .5), ncol=10))
kmo_optimal_solution(df, squared = FALSE)
```

Index