Package ‘ccpsyc’

May 11, 2022

Type Package
Title Methods for Cross-Cultural Psychology
Version 0.2.6
Description With the development of new cross-cultural methods this package is intended to combine multiple functions automating and simplifying functions providing a unified analysis approach for commonly employed methods.
License GPL-3
Depends R (>= 4.0.0)
Imports magrittr, dplyr, lavaan, readr, MCMCpack, psych, ufs, xlsx,
tibble, rlang, RcppAlgos, tidyr
Suggests knitr, rmarkdown, testthat (>= 3.0.0)
Config/testthat/edition 3
Encoding UTF-8
LazyData true
RoxygenNote 7.1.1
NeedsCompilation no
Author Johannes Karl [aut, cre]
Maintainer Johannes Karl <johannes.a.karl@gmail.com>
Repository CRAN
Date/Publication 2022-05-11 05:50:06 UTC

R topics documented:

  boot_inv_eff .................................................. 2
  clearing_fa .................................................. 3
  dMACS .......................................................... 4
  equival .......................................................... 4
  example ......................................................... 5
  format_boot_inv_eff .......................................... 6
  gamma_hat_scaled ............................................. 7
boot_inv_eff

Bootstrapped pairwise differences in psychometric function of groups.

Description

Bootstrapped pairwise differences in psychometric function of groups.

Usage

```r
boot_inv_eff(
  n,  
n_sample,  
  df,  
  items,  
group,  
eff_sizes = c("SDI2", "UDI2", "WSDI", "WUDI", "dmacs"),  
  seed = 2711
)
```

Arguments

- `n`: Number of bootstraps
- `n_sample`: Number of participants to sample
- `df`: Data to resample
- `items`: Items to resample for the model as vector of strings
- `group`: String variable indicating grouping variable
- `eff_sizes`: Effect sizes to be returned
- `seed`: Seed for replicability

Value

Returns a dataframe with the bootstrapped effect sizes based on the invariance_eff function in this package for two country comparisons.
Examples

two_country <- dplyr::filter(example, country %in% c("NZ", "BRA"))
boot_inv_eff(n = 10,
  n_sample = 200, df = two_country, group = "country",
  items = paste0("voice",1:3, "m"))

---

clearing_fa  Function to quickly organize and clear psych factor loadings

Description

Function to quickly organize and clear psych factor loadings

Usage

clearing_fa(
  psych_fa,
  cutoff = 0.4,
  dbl_dist = 0.2,
  key_file = NULL,
  cleaned = TRUE
)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>psych_fa</td>
<td>Output from the psych package, can be either fa or principal with at least two dimensions</td>
</tr>
<tr>
<td>cutoff</td>
<td>Desired cutoff below which loadings are omitted defaults to .40</td>
</tr>
<tr>
<td>dbl_dist</td>
<td>Desired distance between highest and second highest loading for an item to remove double loadings, defaults to .20</td>
</tr>
<tr>
<td>key_file</td>
<td>Optional: Either a .csv or .xlsx file with at least two columns: 1 labeled item containing the item labels as in the data frame, 2 a column labeled wording containing the item wording.</td>
</tr>
<tr>
<td>cleaned</td>
<td>If true (default), only the cleaned solution with a message for descriptive stats are returned. If false the function returns a list of data frames one cleaned and one showing all in-between steps</td>
</tr>
</tbody>
</table>

Value

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>clean</td>
<td>This column contains the assignment after removing NAs and double loadings</td>
</tr>
<tr>
<td>dir</td>
<td>This column contains the direction (positive or negative) of the highest loading.</td>
</tr>
</tbody>
</table>
Examples

```r
library(psych)
fa_solution <- fa(example[c(paste0("help", 1:6, "m"), c(paste0("voice", 1:5, "m"))], nfacors = 2)
clearing_fa(fa_solution)
```

---

**dMACS**

*Computes dMACS*

---

**Description**

Computes dMACS

**Usage**

```r
dMACS(fit.cfa, group1, group2)
```

**Arguments**

- `fit.cfa`: Lavaan output object with two groups and a single factor.
- `group1`: String for first group in the grouping factor.
- `group2`: String for second group in the grouping factor.

**Value**

Returns dMACS for each item.

**Examples**

```r
dMACS
```

---

**equival**

*One-step equivalence testing The function allows for a simple one step test of configural, metric, and scalar equivalence between multiple groups.*

---

**Description**

One-step equivalence testing The function allows for a simple one step test of configural, metric, and scalar equivalence between multiple groups.

**Usage**

```r
equival(x, dat, group, standart_lv = TRUE, orthog = TRUE, estim = "MLM")
```
Arguments

x    CFA model identical to models provided to lavaan.
dat  A data frame or tibble containing the raw data for the specified model.
group A character string that indicates the column of dat that contains the grouping variable. e.g "country"
standart_lv A boolean that indicates whether the latent variables should be standardised.
ortog A boolean that indicates whether the latent variables should be orthogonal.
estim A string indicating the estimator to be used MLM for complete data and MLR for incomplete data. Defaults to MLM

Value

Returns a data frame with the fit indices for each model and delta values comparing the different levels of equivalence. For a step by step interpretation see.

Examples

```r
model <- "voice =~ voice1m + voice2m + voice3m
             help =~ help1m + help2m + help3m"
equiv(x = model, dat = example, group = "country")
```

example  Help and Voice Behavior in different countries

Description

Help and Voice Behavior in different countries

Usage

example

Format

A data frame with 5201 rows and 13 variables:

country Country of sample
help1m First Help Item
help2m Second Help Item
help3m Third Help Item
help4m Fourth Help Item
help5m Fifth Help Item
help6m  Sixth Help Item
help7m  Seventh Help Item
voice1m  First Voice Item
voice2m  Second Voice Item
voice3m  Third Voice Item
voice4m  Fourth Voice Item
voice5m  Fifth Voice Item
...

Source


format_boot_inv_eff  Improving boot effectsize output

Description

Improving boot effectsize output

Usage

format_boot_inv_eff(x)

Arguments

x  The output of a bootstrapped invariance effect call

Value

A formatted tibble with all effect sizes reported by boot_inv_eff from this package and significant determined by 95% CIs either crossing 0 or .30

Examples

two_country <- dplyr::filter(example, country %in% c("NZ", "BRA"))
boot_inv_eff_result <- boot_inv_eff(n = 10,
  n_sample = 200, df = two_country, group = "country",
  items = paste0("voice",1:3, "m"))
format_boot_inv_eff(boot_inv_eff_result)
**gamma_hat_scaled**  
*Gamma Hat from MLM fitted lavaan object*

**Description**
Gamma Hat from MLM fitted lavaan object

**Usage**

```r
gamma_hat_scaled(object)
```

**Arguments**

- `object`  
  A lavaan output object that was fitted with a MLM estimator

---

**invariance_eff**  
*Invariance Effect Sizes*

**Description**
Invariance Effect Sizes

**Usage**

```r
invariance_eff(
  df,  
  items,  
  group,  
  nodewidth = 0.01,  
  intercept_fix = 1,  
  lowerLV = -10,  
  upperLV = 10,  
  ...
)
```

**Arguments**

- `df`  
  Multi-group dataframe
- `items`  
  vector of items for the target construct
- `group`  
  string defining grouping variable
- `nodewidth`  
  Steps tested
- `intercept_fix`  
  Which item should have a fixed intercept defaults to the first item
- `lowerLV`  
  Lower range of latent variable tested
- `upperLV`  
  Upper range of latent variable tested
- `...`  
  Passes on to lavaan CFA functions
Value

Returns a dataframe with a row for each item comprising the uni-factorial solution and one column for each invariance effect size. A detailed interpretation of each effect size is provided in Gunn et al. (2019).

---

lavTestScore.clean  

Get more comprehensible output from lavTestScore

Description

Get more comprehensible output from lavTestScore

Usage

lavTestScore.clean(lavaan.fit, ndigit = 3, ...)

Arguments

- lavaan.fit: Model fitted with lavaan
- ndigit: Defines the rounding
- ...: Arguments passed to lavTestScore

Value

Returns a dataframe which contains one row for each constrained parameter in the model together with a chi-square test indicating whether the parameter significantly differs between groups. This is a cleaned version identical to lavTestScore.

Author(s)

Maksim Rudnev

---

mg_rel_table  

Multi-group reliability table

Description

Multi-group reliability table

Usage

mg_rel_table(df_s, measure_list, group, digitn = 3, seed = 2711)
Arguments

- df_s: The full dataframe with all groups and items.
- measure_list: A named list of vectors containing the item names. The format should be `list(measure_name1 = c('Item1', 'Item2', 'Item3'), measure_name2 = c('Item1', 'Item2', 'Item3'))`
- group: Grouping variable in the dataset as string for example "country"
- digitn: Controls the amount of digits shown in the output
- seed: Seed for the bootstrapped confidence intervals

Value

Returns a formatted dataframe with the reliability of all constructs by group

---

**MNCI**

*Non-Centrality Index*

**Description**

Non-Centrality Index

**Usage**

MNCI(object)

**Arguments**

- object: A lavaan object that was fitted with a MLM estimator/

---

**multi_group_eff**

*Pairwise Effect sizes of similarities and difference in the psychometric structure between multiple groups*

**Description**

Pairwise Effect sizes of similarities and difference in the psychometric structure between multiple groups

**Usage**

multi_group_eff(
  df,
  group,
  items,
  eff_sizes = c("SDI2", "UDI2", "WSDI", "WUDI", "dmacs")
)
Arguments

df    Multi-Group data frame

group    String variable indicating the grouping variable

items    Vector of strings indicating items for the uni-factorial construct

eff_sizes    Effect sizes to be returned

Value

The function returns a list of dataframes with the first reporting the averaged results per item and the second reporting the pairwise comparisons.

Examples

```r
example_s <- dplyr::filter(example, country %in% c("NZ", "BRA"))
multi_group_eff(df = example, group = "country", items = paste0("voice", 1:3, "m"))
```

---

**pancultural**

*Creating a Pan-Cultural Loading Matrix*

Description

Creating a Pan-Cultural Loading Matrix

Usage

```r
pancultural(df, group, nfactors)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>df</td>
<td>A data frame contains the variables for the exploratory factor analysis and the grouping variable.</td>
</tr>
<tr>
<td>group</td>
<td>The name of the column that contains the grouping supplied as a string.</td>
</tr>
<tr>
<td>nfactors</td>
<td>The number of factors expected.</td>
</tr>
</tbody>
</table>

Value

returns a Pan-Cultural loading matrix.

Examples

```r
pancultural(example, "country", 5)
```
**prost**  
*Procrustes rotation function, returning Tucker’s Phi*

**Description**

Procrustes rotation function, returning Tucker’s Phi

**Usage**

```r
prost(loading, norm, rotated = FALSE, digits = 2)
```

**Arguments**

- **loading**: A correlation matrix to be rotated towards a target
- **norm**: A correlation matrix that is the goal of the rotation
- **rotated**: A TRUE/FALSE operator indicating if the rotated matrix should be returned in addition to Tucker’s Phi
- **digits**: The number of digits to be displayed in the output, defaults to 2

**Value**

Returns Tuckers Phi evaluating the congruence of the loading matrix to the normative matrix

---

**release_bonferroni**  
*Examining chisquare improvement if paths are unconstrained. The function returns the paths to be unconstrained based on chisquare change. Adjusted P-values are calculated based on iterative Bonferroni corrections.*

**Description**

Examening chisquare improvement if paths are unconstrained. The function returns the paths to be unconstrained based on chisquare change. Adjusted P-values are calculated based on iterative Bonferroni corrections.

**Usage**

```r
release_bonferroni(lavaan.fit, ndigit = 3, exp_p = 0.05, ...)
```

**Arguments**

- **lavaan.fit**: Model fitted with lavaan
- **ndigit**: Number of digits to round chi and p to
- **exp_p**: Expected p-value
- **...**: Arguments passed to lavTestScore
Value

Returns a dataframe representing a Bonferroni corrected version of `lavTestScore.clean`.

Author(s)

Maksim Rudnev

---

**splitgroup**  
_Split by groups_

**Description**

Split by groups

**Usage**

```r
splitgroup(df, group, named = FALSE, name.list = NA)
```

**Arguments**

- `df`  
  Dataframe
- `group`  
  Variable from the dataset that defines the groups
- `named`  
  TRUE/FALSE argument wheter the resulting list should be named
- `name.list`  
  Supply a list of names same length as number of groups

**Value**

Returns a list of dataframes with only the selected groups
Index

* datasets
  example, 5
boot_inv_eff, 2
clearing_fa, 3
dMACS, 4
equival, 4
example, 5
format_boot_inv_eff, 6
gamma_hat_scaled, 7
invariance_eff, 7
lavTestScore, 8
lavTestScore.clean, 8, 12
mg_rel_table, 8
MNCI, 9
multi_group_eff, 9

pancultural, 10
prost, 11
release_bonferroni, 11
splitgroup, 12