Package ‘configural’

February 19, 2019

Type Package
Title Multivariate Profile Analysis
Version 0.1.1
Date 2019-01-20

BugReports https://github.com/bwiernik/configural/issues


License GPL (>= 3)
Depends R (>= 3.4.0)
Encoding UTF-8
LazyData true
Imports stats, dplyr, crayon, cli, stringr
RoxygenNote 6.1.1
NeedsCompilation no

Author Brenton M. Wiernik [aut, cre]
Maintainer Brenton M. Wiernik <brenton@wiernik.org>
Repository CRAN
Date/Publication 2019-02-19 18:10:09 UTC

R topics documented:

configural-package ............................................. 2
adjust_Rsq .................................................... 3
configural_news ............................................... 4
cor_covariance ............................................... 4
cor_covariance_meta ....................................... 5
cor_labels ...................................................... 6
cpa_mat .......................................................... 7
Description

Overview of the \texttt{configural} package.

Details

The \texttt{configural} package provides tools for conducting configural and profile analyses. It currently supports criterion profile analysis (Davison & Davenport, 2002) and meta-analytic criterion profile analysis (Wiernik et al., 2019). Functions are provided to calculate criterion patterns and CPA variance decomposition, as well as for computing confidence intervals, shrinkage corrections, and fungible patterns.

Author(s)

\textbf{Maintainer}: Brenton M. Wiernik <brenton@wiernik.org>

See Also

Useful links:

- Report bugs at \url{https://github.com/bwiernik/configural/issues}
**adjust_Rsq**

Adjust a regression model R-squared for overfitting

### Description

Estimate shrinkage for regression models

### Usage

```r
adjust_rsq(rsq, n, p, adjust = c("fisher", "pop", "cv"))
```

### Arguments

- `rsq`: Observed model R-squared
- `n`: Sample size
- `p`: Number of predictors
- `adjust`: Which adjustment to apply. Options are "fisher" for the Adjusted R-squared method used in `stats::lm()`, "pop" for the positive-part Pratt estimator of the population R-squared, and "cv" for the Browne/positive-part Pratt estimator of the cross-validity R-squared. Based on Shieh (2008), these are the estimators for the population and cross-validity R-squared values that show the least bias with a minimal increase in computational complexity.

### Value

An adjusted R-squared value.

### References


### Examples

```r
adjust_rsq(.55, 100, 6, adjust = "pop")
```
configural_news

Retrieve the NEWS file for the configural package

Description

This function gives a shortcut to the utils::news(package = "configural") function and displays configural’s NEWS file, which contains version information, outlines additions and changes to the package, and describes other updates.

Usage

configural_news()  

Examples

configural_news()

cor_covariance

Calculate the asymptotic sampling covariance matrix for the unique elements of a correlation matrix

Description

Calculate the asymptotic sampling covariance matrix for the unique elements of a correlation matrix

Usage

cor_covariance(r, n)

Arguments

r  
A correlation matrix

n  
The sample size

Value

The asymptotic sampling covariance matrix

Author(s)

Based on an internal function from the fungible package by Niels Waller

References

**cor_covariance_meta**  

**Examples**

```r
cor_covariance(matrix(c(1, .2, .3, .2, 1, .3, .3, 1), ncol = 3), 100)
```

---

**Estimate the asymptotic sampling covariance matrix for the unique elements of a meta-analytic correlation matrix**

**Description**

Estimate the asymptotic sampling covariance matrix for the unique elements of a meta-analytic correlation matrix.

**Usage**

```r
cor_covariance_meta(r, n, sevar, source = NULL, rho = NULL, sevar_rho = NULL, n_overlap = NULL)
```

**Arguments**

- `r`: A meta-analytic matrix of observed correlations (can be full or lower-triangular).
- `n`: A matrix of total sample sizes for the meta-analytic correlations in `r` (can be full or lower-triangular).
- `sevar`: A matrix of estimated sampling error variances for the meta-analytic correlations in `r` (can be full or lower-triangular).
- `source`: A matrix indicating the sources of the meta-analytic correlations in `r` (can be full or lower-triangular). Used to estimate overlapping sample size for correlations when `n_overlap` is `NULL`.
- `rho`: A meta-analytic matrix of corrected correlations (can be full or lower-triangular).
- `sevar_rho`: A matrix of estimated sampling error variances for the meta-analytic corrected correlations in `rho` (can be full or lower-triangular).
- `n_overlap`: A matrix indicating the overlapping sample size for the unique (lower triangular) values in `r` (can be full or lower-triangular). Values must be arranged in the order returned by `cor_labels(colnames(R))`.

**Details**

If both `source` and `n_overlap` are `NULL`, it is assumed that all meta-analytic correlations come from the same source.

**Value**

The estimated asymptotic sampling covariance matrix.
Reference

Nel, D. G. (1985). A matrix derivation of the asymptotic covariance matrix of sample correlation coefficients. *Linear Algebra and Its Applications*, 67, 137–145. [https://doi.org/10/c7jmg](https://doi.org/10/c7jmg)


Examples

```r
cor_covariance_meta(r = mindfulness$r, n = mindfulness$n, 
sevar = mindfulness$sevar_r, source = mindfulness$source)
```

### cor_labels

`cor_labels` is a function that generates labels for correlations from a vector of variable names.

**Generate labels for correlations from a vector of variable names**

#### Description

This function returns a vector of labels for the unique correlations between pairs of variables from a supplied vector of variable names.

#### Usage

```r
cor_labels(var_names)
```

#### Arguments

- `var_names`: A character vector of variable names

#### Value

A vector of correlation labels

#### Examples

```r
cor_labels(colnames(mindfulness$r))
```
Conduct criterion profile analysis using a correlation matrix

**Description**

Conduct criterion profile analysis using a correlation matrix

**Usage**

```r
cpa_mat(formula, cov_mat, n = Inf, se_var_mat = NULL, se_beta_method = c("normal", "lm"), adjust = c("fisher", "pop", "cv"), conf_level = 0.95, ...)
```

**Arguments**

- `formula`: Regression formula with a single outcome variable on the left-hand side and one or more predictor variables on the right-hand side (e.g., `Y ~ X1 + X2`).
- `cov_mat`: Correlation matrix containing the variables to be used in the regression.
- `n`: Sample size to be used in calculating adjusted R-squared and, if `se_var_mat` is NULL, standard errors.
- `se_var_mat`: Optional. The sampling error covariance matrix among the unique elements of `cov_mat`. Used to calculate standard errors. If not supplied, the sampling covariance matrix is calculated using `n`.
- `se_beta_method`: Method to use to estimate the standard errors of standardized regression (beta) coefficients. Current options include "normal" (use the Jones-Waller, 2015, normal-theory approach) and "lm" (estimate standard errors using conventional regression formulas).
- `adjust`: Method to adjust R-squared for overfitting. See `adjust_Rsq` for details.
- `conf_level`: Confidence level to use for confidence intervals.
- `...`: Additional arguments.

**Value**

An object of class "cpa" containing the criterion pattern vector and CPA variance decomposition

**References**


Disorders

Examples

```r
sevar <- cor_covariance_meta(mindfulness$r, mindfulness$n, mindfulness$sevar_r, mindfulness$source)
cpa_mat(mindfulness ~ ES + A + C + Ex + O,
  cov_mat = mindfulness$rho,
  n = harmonic_mean(vechs(mindfulness$n)),
  se_var_mat = sevar,
  adjust = "pop")
```

<table>
<thead>
<tr>
<th>disorders</th>
<th>Meta-analytic correlations among Big Five personality traits and psychological disorders</th>
</tr>
</thead>
</table>

Description

Big Five intercorrelations from Davies et al. (2015). Big Five–psychological disorder correlations from Kotov et al. (2010). Note that there were several duplicate or missing values in the reported data table in the published article. These results are based on corrected data values.

Usage

```r
data(disorders)
```

Format

list with entries `r` (mean observed correlations), `rho` (mean corrected correlations), `n` (sample sizes), `sevar_r` (sampling error variances for mean observed correlations), `sevar_rho` (sampling error variances for mean corrected correlations), and `source` (character labels indicating which meta-analytic correlations came from the same source)

References

Davies, S. E., Connelly, B. L., Ones, D. S., & Birkland, A. S. (2015). The general factor of personality: The “Big One,” a self-evaluative trait, or a methodological gnat that won’t go away? *Personality and Individual Differences, 81*, 13–22. [https://doi.org/10/bc98](https://doi.org/10/bc98)


Examples

```r
data(disorders)
```
### fungible

**Locate extrema of fungible weights for regression and related models**

#### Description

Generates fungible regression weights (Waller, 2008) and related results using the method by Waller and Jones (2010).

#### Usage

```r
fungible(object, theta = 0.005, Nstarts = 1000, MaxMin = c("min", "max"), silent = FALSE, ...)
```

#### Arguments

- **object**: A fitted model object. Currently supported classes are: "cpa"
- **theta**: A vector of values to decrement from R-squared to compute families of fungible coefficients.
- **Nstarts**: Maximum number of (max) minimizations from random starting configurations.
- **MaxMin**: Should the cosine between the observed and alternative weights be maximized ("max") to find the maximally similar coefficients or minimized ("min") to find the maximally dissimilar coefficients?
- **silent**: Should current optimization values be printed to the console (FALSE) or suppressed (TRUE)?
- **...**: Additional arguments

#### Value

A list containing the alternative weights and other fungible weights estimation parameters

#### Author(s)

Niels Waller, Jeff Jones, Brenton M. Wiernik. Adapted from fungible::fungibleExtrema().

#### References


Examples

```r
mind <- cpa_mat(mindfulness ~ ES + A + C + Ex + O,
    cov_mat = mindfulness$r,
    n = harmonic_mean(vechs(mindfulness$n)),
    se_var_mat = cor_covariance_meta(mindfulness$r,
        mindfulness$n,
        mindfulness$sevar_r,
        mindfulness$source),

    adjust = "pop")

mind_fung <- fungible(mind, Nstarts = 100)
```

fungible.cpa

Locate extrema of fungible criterion profile patterns

Description

Identify maximally similar or dissimilar criterion patterns in criterion profile analysis

Usage

```r
## S3 method for class 'cpa'
fungible(object, theta = 0.005, Nstarts = 1000,
    MaxMin = c("min", "max"), silent = FALSE, ...)
```

Arguments

- **object**: A fitted model object of class "cpa".
- **theta**: A vector of values to decrement from R-squared to compute families of fungible coefficients.
- **Nstarts**: Maximum number of (max) minimizations from random starting configurations.
- **MaxMin**: Should the cosine between the observed and alternative weights be maximized ("max") to find the maximally similar coefficients or minimized ("min") to find the maximally dissimilar coefficients?
- **silent**: Should current optimization values be printed to the console (FALSE) or suppressed (TRUE)?
- **...**: Additional arguments

Value

A list containing the alternative weights and other fungible weights estimation parameters

References

fungible.lm

Examples

```r
mind <- cpa_mat(mindfulness ~ ES + A + C + Ex + O,
    cov_mat = mindfulness$r,
    n = harmonic_mean(vechs(mindfulness$n)),
    se_var_mat = cor_covariance_meta(mindfulness$r,
        mindfulness$n,
        mindfulness$sevar_r,
        mindfulness$source),
    adjust = "pop")
mind_fung <- fungible(mind, Nstarts = 100)
```

fungible.lm  Locate extrema of fungible OLS regression weights

Description

Identify maximally similar or dissimilar sets of fungible standardized regression coefficients from an OLS regression model

Usage

```r
## S3 method for class 'lm'
fungible(object, theta = 0.005, Nstarts = 1000,
    MaxMin = c("min", "max"), silent = FALSE, ...)
```

Arguments

- `object` A fitted model object of class "lm" or "summary.lm".
- `theta` A vector of values to decrement from R-squared to compute families of fungible coefficients.
- `Nstarts` Maximum number of (max) minimizations from random starting configurations.
- `MaxMin` Should the cosine between the observed and alternative weights be maximized ("max") to find the maximally similar coefficients or minimized ("min") to find the maximally dissimilar coefficients?
- `silent` Should current optimization values be printed to the console (FALSE) or suppressed (TRUE)?
- `...` Additional arguments

Value

A list containing the alternative weights and other fungible weights estimation parameters

References

Examples

```r
lm_mtcars <- lm(mpg ~ cyl + disp + hp + drat + wt + qsec + vs + am + gear + carb, data = mtcars)
lm_mtcars_fung <- fungible(lm_mtcars, Nstarts = 100)
```

---

**harmonic_mean**

Find the harmonic mean of a vector, matrix, or columns of a data.frame

---

Description

The harmonic mean is merely the reciprocal of the arithmetic mean of the reciprocals.

Usage

```r
harmonic_mean(x, na.rm = TRUE, zero = TRUE)
```

Arguments

- `x`: A vector, matrix, or data.frame
- `na.rm`: Logical. If TRUE, remove NA values before processing
- `zero`: Logical, If TRUE, if there are any zeroes, return 0, else, return the harmonic mean of the non-zero elements

Value

The harmonic mean of `x`

Author(s)

Adapted from `psych::harmonic.mean()` by William Revelle

Examples

```r
harmonic_mean(1:10)
```
**Description**


**Usage**

data(hrm)

**Format**

- list with entries `r` (mean observed correlations), `rho` (mean corrected correlations), `n` (sample sizes), `sevar_r` (sampling error variances for mean observed correlations), `sevar_rho` (sampling error variances for mean corrected correlations), and `source` (character labels indicating which meta-analytic correlations came from the same source)

**References**


**Examples**

data(hrm)

---

**Description**

Self-rated job characteristics intercorrelations and correlations with other-rated job performance and self-rated job satisfaction from Humphrey et al. (2007).

**Usage**

data(jobchar)
Format

list with entries r (mean observed correlations), rho (mean corrected correlations), n (sample sizes), sevar_r (sampling error variances for mean observed correlations), sevar_rho (sampling error variances for mean corrected correlations), and source (character labels indicating which meta-analytic correlations came from the same source)

References


Examples

data(jobchar)

```
mindfulness
```

Description


Usage

data(mindfulness)

Format

list with entries r (mean observed correlations), rho (mean corrected correlations), n (sample sizes), sevar_r (sampling error variances for mean observed correlations), sevar_rho (sampling error variances for mean corrected correlations), and source (character labels indicating which meta-analytic correlations came from the same source)

References


Examples

data(mindfulness)

---

prejudice

Correlations between study design moderators and effect sizes for prejudice reduction following intergroup contact

Description

Correlations among study design moderators and study design moderator–observed prejudice reduction effect sizes from Pettigrew and Tropp (2008). Note that correlations with effect size have been reverse-coded so that a positive correlation indicates that a higher level of the moderator is associated with larger prejudice reduction.

Usage

data(prejudice)

Format

list with entries \( r \) (observed correlations among moderators) and \( k \) (number of samples in meta-analysis)

References


Examples

data(prejudice)

---

team

Meta-analytic correlations among team processes and team effectiveness

Description

Team process intercorrelations and team process–team performance/affect correlations from LePine et al. (2008).

Usage

data(team)
Format

list with entries r (mean observed correlations), rho (mean corrected correlations), n (sample sizes), sevar_r (sampling error variances for mean observed correlations), sevar_rho (sampling error variances for mean corrected correlations), and source (character labels indicating which meta-analytic correlations came from the same source)

Details

Note that LePine et al. (2008) did not report confidence intervals, sampling error variances, or heterogeneity estimates for correlations among team processes; included sampling error variances in this list are based on total sample size only and do not include uncertainty stemming from any effect size heterogeneity.

References


Examples

data(team)

typical

calculation

var_error_cpa 

Description

Estimate the sampling error variance for criterion profile analysis parameters

Usage

var_error_cpa(Rxx, rxy, n, se_var_mat = "normal", adjust = c("fisher", "pop", "cv"))

Arguments

Rxx 
An intercorrelation matrix among the predictor variables

rxy 
A vector of predictor–criterion correlations

n 
The sample size

se_var_mat 
The method used to calculate the sampling covariance matrix for the unique elements of Rxx and rxy. Can be "normal" to estimate the covariance matrix using "n" or a matrix of sampling covariance values.

adjust 
Method to adjust R-squared for overfitting. See adjust_Rsq for details.
Value

A list containing sampling covariance matrices or sampling error variance estimates for CPA parameters.

Examples

```r
## Not run:
var_error_cpa(mindful_rho[1:5, 1:5], mindful_rho[1:5, 6], n = 17060)
## End(Not run)
```

---

**vech**  
Vectorize a matrix

Description

cvec returns the column-wise vectorization of an input matrix (stacking the columns on one another). rvec returns the row-wise vectorization of an input matrix (concatenating the rows after each other). vech returns the column-wise half-vectorization of an input matrix (stacking the lower triangular elements of the matrix, including the diagonal). vechs returns the strict column-wise half-vectorization of an input matrix (stacking the lower triangular elements of the matrix, excluding the diagonal). All functions return the output as a vector.

Usage

```r
vech(x)
vechs(x)
cvec(x)
rvec(x)
```

Arguments

- **x**  
  A matrix

Value

A vector of values

Author(s)

Based on functions from the OpenMx package
**Examples**

```r
cvec(matrix(1:9L SL SII
rvec(matrix(1:9L SL SII
vech(matrix(1:9L SL SII
vechs(matrix(1:9L SL SII
vechs(matrix(1:1RL SL TII
vechRfull
```

---

**vech2full**  
*Inverse vectorize a matrix*

---

**Description**

These functions return the symmetric matrix that produces the given half-vectorization result.

**Usage**

```r
vech2full(x)
vechs2full(x, diagonal = 1)
```

**Arguments**

- `x`  
  A vector

- `diagonal`  
  A value or vector of values to enter on the diagonal for `vechs2full` (default = 1)

**Details**

The input consists of a vector of the elements in the lower triangle of the resulting matrix (for `vech2full`, including the elements along the diagonal of the matrix, as a column vector), filled column-wise. For `vechs2full`, the diagonal values are filled as 1 by default, alternative values can be specified using the `diag` argument. The inverse half-vectorization takes a vector and reconstructs a symmetric matrix such that `vech2full(vech(x))` is identical to `x` if `x` is symmetric.

**Value**

A symmetric matrix

**Author(s)**

Based on functions from the OpenMx package

**Examples**

```r
vech2full(c(1, 2, 3, 5, 6, 9))
vechs2full(c(2, 3, 6), diagonal = 0)
```
wt_cov

**Compute weighted covariances**

**Description**

Compute the weighted covariance among variables in a matrix or between the variables in two separate matrices/vectors.

**Usage**

```r
wt_cov(x, y = NULL, wt = NULL, as_cor = FALSE, use = c("everything", "listwise", "pairwise"), unbiased = TRUE, df_type = c("count", "sum_wts"))

wt_cor(x, y = NULL, wt = NULL, use = "everything")
```

**Arguments**

- `x`: Vector or matrix of x variables.
- `y`: Vector or matrix of y variables
- `wt`: Vector of weights
- `as_cor`: Logical scalar that determines whether the covariances should be standardized (TRUE) or unstandardized (FALSE).
- `use`: Method for handling missing values. "everything" uses all values and does not account for missingness, "listwise" uses only complete cases, and "pairwise" uses pairwise deletion.
- `unbiased`: Logical scalar determining whether variance should be unbiased (TRUE) or maximum-likelihood (FALSE).
- `df_type`: Character scalar determining whether the degrees of freedom for unbiased estimates should be based on numbers of cases (n - 1; "count"; default) or squared sums of weights (1 - sum(w^2); "sum_wts")

**Value**

Scalar, vector, or matrix of covariances.

**Author(s)**

Jeffrey A. Dahlke
Examples

\begin{verbatim}
wt_cov(x = c(1, 0, 2), y = c(1, 2, 3), wt = c(1, 2, 2), as_cor = FALSE, use = "everything")
wt_cov(x = c(1, 0, 2), y = c(1, 2, 3), wt = c(1, 2, 2), as_cor = TRUE, use = "everything")
wt_cov(x = cbind(c(1, 0, 2), c(1, 2, 3)), wt = c(1, 2, 2), as_cor = FALSE, use = "everything")
wt_cov(x = cbind(c(1, 0, 2), c(1, 2, 3)), wt = c(1, 2, 2), as_cor = TRUE, use = "everything")
wt_cov(x = cbind(c(1, 0, RL na), c(1, RL SI)), wt = c(1, 2, RL 1), as_cor = FALSE, use = "listwise")
wt_cov(x = cbind(c(1, 0, 2, NA), c(1, 2, 3, 3)), wt = c(1, 2, 2, 1), as_cor = TRUE, use = "listwise")
\end{verbatim}

wt_dist

Weighted descriptive statistics for a vector of numbers

Description

Compute the weighted mean and variance of a vector of numeric values. If no weights are supplied, defaults to computing the unweighted mean and the unweighted maximum-likelihood variance.

Usage

\begin{verbatim}
wt_dist(x, wt = rep(1, length(x)), unbiased = TRUE, df_type = c("count", "sum_wts"))
wt_mean(x, wt = rep(1, length(x)))
wt_var(x, wt = rep(1, length(x)), unbiased = TRUE, df_type = c("count", "sum_wts"))
\end{verbatim}

Arguments

- **x**: Vector of values to be analyzed.
- **wt**: Weights associated with the values in x.
- **unbiased**: Logical scalar determining whether variance should be unbiased (TRUE) or maximum-likelihood (FALSE).
- **df_type**: Character scalar determining whether the degrees of freedom for unbiased estimates should be based on numbers of cases ("count"; default) or sums of weights ("sum_wts").

Details

The weighted mean is computed as

\[
\bar{x}_w = \frac{\sum_{i=1}^{k} x_i w_i}{\sum_{i=1}^{k} w_i}
\]

where \( x \) is a numeric vector and \( w \) is a vector of weights.
The weighted variance is computed as

$$var_w(x) = \frac{\sum_{i=1}^{k} (x_i - \bar{x}_w)^2 w_i}{\sum_{i=1}^{k} w_i}$$

and the unbiased weighted variance is estimated by multiplying $var_w(x)$ by $\frac{k}{k-1}$.

**Value**

A weighted mean and variance if weights are supplied or an unweighted mean and variance if weights are not supplied.

**Author(s)**

Jeffrey A. Dahlke

**Examples**

```r
wt_dist(x = c(.1, .3, .5), wt = c(100, 200, 300))
wt_mean(x = c(.1, .3, .5), wt = c(100, 200, 300))
wt_var(x = c(.1, .3, .5), wt = c(100, 200, 300))
```

---

**Quadratic form matrix product**

**Description**

Calculate the quadratic form

$$Q = x'Ax$$

**Usage**

```r
A %&% x
```

**Arguments**

- `A` A square matrix
- `x` A vector or matrix

**Value**

The quadratic product

**Examples**

```r
diag(5) %&% 1:5
```
Index

*Topic **datasets**
- disorders, 8
- hrm, 13
- jobchar, 13
- mindfulness, 14
- prejudice, 15
- team, 15

*Topic **univar**
- wt_dist, 20

adjust_Rsq, 3, 7, 16

cconfigural (configural-package), 2
- configural-package, 2
- configural_news, 4
- cor_covariance, 4
- cor_covariance_meta, 5
- cor_labels, 6
- cpa_mat, 7
- cvec (vech), 17

disorders, 8

fungible, 9
- fungible.cpa, 10
- fungible.lm, 11
- fungible::fungibleExtrema, 9

harmonic_mean, 12
- hrm, 13
- jobchar, 13
- mindfulness, 14
- prejudice, 15
- psych::harmonic.mean, 12
- rvec (vech), 17
- stats::lm, 3

var_error_cpa, 16
- vech, 17
- vech2full, 18
- vechs (vech), 17
- vechs2full (vech2full), 18
- wt_cor (wt_cov), 19
- wt_cov, 19
- wt_dist, 20
- wt_mean (wt_dist), 20
- wt_var (wt_dist), 20