Package ‘metaSEM’

Type Package
Title Meta-Analysis using Structural Equation Modeling
Version 1.3.1
Date 2023-08-08
Depends R (>= 3.4.0), OpenMx
Imports Matrix, MASS, ellipse, graphics, stats, utils, mvtnorm,
       numDeriv, lavaan
Suggests metafor, semPlot, R.rsp, testthat, matrixcalc
VignetteBuilder R.rsp
Maintainer Mike Cheung <mikewlcheung@nus.edu.sg>
Description A collection of functions for conducting meta-analysis using a
structural equation modeling (SEM) approach via the ‘OpenMx’ and
‘lavaan’ packages. It also implements various procedures to
perform meta-analytic structural equation modeling on the
correlation and covariance matrices, see Cheung (2015)
License GPL (>= 2)
LazyLoad yes
LazyData yes
ByteCompile yes
URL https://github.com/mikewlcheung/metasem
BugReports https://github.com/mikewlcheung/metasem/issues
NeedsCompilation no
Author Mike Cheung [aut, cre] (<https://orcid.org/0000-0003-0113-0758>)
Repository CRAN
Date/Publication 2023-08-09 18:50:02 UTC
### R topics documented:

- `metaSEM-package` .................................................. 4
- `Aloe14` .......................................................... 5
- `anova` ........................................................... 7
- `as.mxAlgebra` ....................................................... 8
- `as.mxMatrix` ......................................................... 10
- `as.symMatrix` ......................................................... 12
- `asyCov` ............................................................ 13
- `BCG` ............................................................... 15
- `bdiagMat` .......................................................... 16
- `bdiagRep` .......................................................... 17
- `Becker09` .......................................................... 18
- `Becker83` .......................................................... 22
- `Becker92` .......................................................... 23
- `Becker94` .......................................................... 25
- `Berkey98` .......................................................... 27
- `Boer16` ............................................................. 29
- `bootuniR1` .......................................................... 30
- `bootuniR2` .......................................................... 31
- `Bornmann07` ....................................................... 32
- `calEffSizes` ........................................................ 34
- `checkRAM` ........................................................ 36
- `Cheung00` ........................................................ 37
- `Cheung09` ........................................................ 39
- `coef` ............................................................... 40
- `Cooke16` .......................................................... 42
- `Cooper03` ........................................................ 44
- `Cor2DataFrame` ..................................................... 45
- `create.Fmatrix` .................................................... 47
- `create.modMatrix` ................................................ 48
- `create.mxMatrix` ................................................ 49
- `create.mxModel` .................................................. 50
- `create.Tau2` ....................................................... 52
- `create.V` .......................................................... 53
- `create.vechsR` .................................................... 54
- `Diag` .............................................................. 56
- `Digman97` ........................................................ 57
- `Gleser94` ........................................................ 60
- `Gnambs18` ........................................................ 61
- `HedgesOlkin85` .................................................. 62
- `homoStat` ........................................................ 63
- `Hox02` .............................................................. 65
- `Hunter83` ........................................................ 66
- `impliedR` .......................................................... 69
- `indirectEffect` .................................................... 72
- `is.pd` ............................................................... 73
- `issp05` ............................................................ 74
<table>
<thead>
<tr>
<th>R topics documented:</th>
</tr>
</thead>
<tbody>
<tr>
<td>issp89</td>
</tr>
<tr>
<td>Jaramillo05</td>
</tr>
<tr>
<td>Kalakan96</td>
</tr>
<tr>
<td>lavaan2RAM</td>
</tr>
<tr>
<td>list2matrix</td>
</tr>
<tr>
<td>Mak09</td>
</tr>
<tr>
<td>Mathieu15</td>
</tr>
<tr>
<td>matrix2bdiag</td>
</tr>
<tr>
<td>meta</td>
</tr>
<tr>
<td>meta2semPlot</td>
</tr>
<tr>
<td>meta3L</td>
</tr>
<tr>
<td>Nam03</td>
</tr>
<tr>
<td>Nohe15</td>
</tr>
<tr>
<td>Norton13</td>
</tr>
<tr>
<td>osmasem</td>
</tr>
<tr>
<td>osmasemR2</td>
</tr>
<tr>
<td>osmasemSRMR</td>
</tr>
<tr>
<td>pattern.n</td>
</tr>
<tr>
<td>pattern.na</td>
</tr>
<tr>
<td>plot</td>
</tr>
<tr>
<td>print</td>
</tr>
<tr>
<td>rCor</td>
</tr>
<tr>
<td>readData</td>
</tr>
<tr>
<td>reml</td>
</tr>
<tr>
<td>reml3L</td>
</tr>
<tr>
<td>rerun</td>
</tr>
<tr>
<td>Roorda11</td>
</tr>
<tr>
<td>Scalco17</td>
</tr>
<tr>
<td>smdMES</td>
</tr>
<tr>
<td>smdMTS</td>
</tr>
<tr>
<td>Stadler15</td>
</tr>
<tr>
<td>summary</td>
</tr>
<tr>
<td>Tenenbaum02</td>
</tr>
<tr>
<td>tssem1</td>
</tr>
<tr>
<td>tssemParaVar</td>
</tr>
<tr>
<td>uniR1</td>
</tr>
<tr>
<td>uniR2</td>
</tr>
<tr>
<td>vanderPol17</td>
</tr>
<tr>
<td>VarCorr</td>
</tr>
<tr>
<td>vcov</td>
</tr>
<tr>
<td>vec2symMat</td>
</tr>
<tr>
<td>wls</td>
</tr>
<tr>
<td>wvs94a</td>
</tr>
<tr>
<td>wvs94b</td>
</tr>
</tbody>
</table>
Description

A collection of functions for conducting meta-analysis using a structural equation modeling (SEM) approach via the 'OpenMx' and 'lavaan' packages. It also implements various procedures to perform meta-analytic structural equation modeling on the correlation and covariance matrices.

Details

- **Package:** metaSEM
- **Type:** Package
- **Version:** 1.3.1
- **Date:** 2023-08-08
- **License:** GPL (>=2)
- **LazyLoad:** yes

Author(s)

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>
Maintainer: Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

References


---

**Aloe14**

*Multivariate effect sizes between classroom management self-efficacy (CMSE) and other variables reported by Aloe et al. (2014)*

**Description**

This study reports sixteen studies on the effect sizes (correlation coefficients) between CMSE and emotional exhaustion (EE), depersonalization (DP), and (lowered) personal accomplishment (PA) reported by Aloe et al. (2014).

**Usage**

`data("Aloe14")`
A data frame with 16 observations on the following 14 variables.

Study a factor with levels Betoret Brouwers & Tomic Bumen Chang Durr Evers et al. Friedman Gold Huk Kress Kumarakulasingam Martin et al. Ozdemir Skaalvik and Skaalvik Williams

Year Year of publication
EE Emotional exhaustion
DP Depersonalization
PA (Lowered) personal accomplishment
V_EE Sampling variance of emotional exhaustion
V_DP Sampling variance of depersonalization
V_PA Sampling variance of (lowered) personal accomplishment
C_EE_DP Sampling covariance between EE and DP
C_EE_PA Sampling covariance between EE and PA
C_DP_PA Sampling covariance between DP and PA
Publication_type Either Dissertation or Journal
Percentage_females Percentage of females in the study
Years_experience Average years of experience

Source

Examples
## Not run:
data(Aloe14)

## Random-effects meta-analysis
meta1 <- meta(cbind(EE,DP,PA),
              cbind(V_EE, C_EE_DP, C_EE_PA, V_DP, C_DP_PA, V_PA),
data=Aloe14)

## Remove error code
meta1 <- rerun(meta1)

## Extract the coefficients for the variance component of the random effects
coef1 <- coef(meta1, select="random")

## Convert it into a symmetric matrix by row major
my.cov <- vec2symMat(coef1, byrow=TRUE)

## Convert it into a correlation matrix
anova

```r
cov2cor(my.cov)

## Plot the multivariate effect sizes
plot(meta1)

## End(Not run)
```

---

anova

**Compare Nested Models with Likelihood Ratio Statistic**

### Description

It compares nested models with the likelihood ratio statistic from various objects. It is a wrapper of `mxCompare`.

### Usage

```r
## S3 method for class 'wls'
anova(object, ..., all=FALSE)

## S3 method for class 'meta'
anova(object, ..., all=FALSE)

## S3 method for class 'meta3LFIML'
anova(object, ..., all=FALSE)

## S3 method for class 'reml'
anova(object, ..., all=FALSE)

## S3 method for class 'osmasem'
anova(object, ..., all=FALSE)
```

### Arguments

- `object` An object or a list of objects of various classes. It will be passed to the base argument in `mxCompare`.
- `...` An object or a list of objects of various classes. It will be passed to the comparison argument in `mxCompare`.
- `all` A Boolean value on whether to compare all bases with all comparisons. It will be passed to the all argument in `mxCompare`.

### Value

A table of comparisons between the models in base and comparison.

### Note

When the objects are class `wls`, the degrees of freedom in the base and comparison models are incorrect, while the degrees of freedom of the difference between them is correct. If users want to obtain the correct degrees of freedom in the base and comparison models, they may individually apply the `summary` function on the base and comparison models.
Author(s)
Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

Examples

```r
## Test the significance of a predictor with likelihood ratio test
## Model0: No predictor
model0 <- meta(y=yi, v=vi, data=Hox02, model.name="No predictor")

## Model1: With a predictor
model1 <- meta(y=yi, v=vi, x=weeks, data=Hox02, model.name="One predictor")

## Compare these two models
anova(model1, model0)
```

---

**as.mxAlgebra**

Convert a Character Matrix into MxAlgebra-class

**Description**

It converts a character matrix into `mxAlgebra` object.

**Usage**

```r
as.mxAlgebra(x, startvalues=NULL, name="X")
```

**Arguments**

- `x`: A character or numeric matrix, which consists of valid operators in `mxAlgebra`.
- `startvalues`: A list of starting values of the free parameters. If it is not provided, all free parameters are assumed 0.
- `name`: A character string of the names of the objects based on.

**Details**

Suppose the name argument is "X", the output is a list of the following elements.

**Value**

- `mxalgebra`: An `mxAlgebra` object.
- `parameters`: A column vector `mxMatrix` of the free parameters.
- `list`: A list of `mxMatrix` to form the `mxAlgebra` object.

**Author(s)**

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>
See Also

as.mxMatrix, mxAlgebra

Examples

## a, b, and c are free parameters
(A1 <- matrix(c(1, "a*b", "a^b", "exp(c)"), ncol=2, nrow=2))
## [,1] [,2]
## [1,] "1" "a^b"
## [2,] "a*b" "exp(c)"
A <- as.mxAlgebra(A1, startvalues=list(a=1, b=2), name="A")

## An object of mxAlgebra
A$mxalgebra
## mxAlgebra 'A'
## $formula: rbind(cbind(A1_1, A1_2), cbind(A2_1, A2_2))
## $result: (not yet computed) <0 x 0 matrix>
## dimnames: NULL

## A matrix of parameters
A$parameters
## FullMatrix 'Avars'

## $labels
## [,1]
## [1,] "a"
## [2,] "b"
## [3,] "c"

## $values
## [,1]
## [1,] 1
## [2,] 2
## [3,] 0

## $free
## [,1]
## [1,] TRUE
## [2,] TRUE
## [3,] TRUE

## $lbound: No lower bounds assigned.

## $ubound: No upper bounds assigned.

## A list of matrices of elements for the mxAlgebra
A$list
## $A1_1
## mxAlgebra 'A1_1'
## $formula: 1
As.mxMatrix

Convert a Matrix into MxMatrix-class

Description

It converts a matrix into MxMatrix-class via mxMatrix.

Usage

as.mxMatrix(x, name, ...)

Arguments

x A character or numeric matrix. If x is not a matrix, as.matrix(x) is applied first.
name An optional character string as the name of the MxMatrix object created by mxModel function. If the name is missing, the name of x will be used.
... Further arguments to be passed to mxMatrix. It should be noted that type, nrow, ncol, values, free, name and labels will be created automatically. Thus, these arguments except labels should be avoided in ...

Details

If there are non-numeric values in x, they are treated as the labels of the parameters. If a "*" is present, the numeric value on the left-hand side will be treated as the starting value for a free parameter. If an "@" is present, the numeric value on the left-hand side will be considered as the value for a fixed parameter. If it is a matrix of numeric values, there are no free parameters in the output matrix.
Value

A **MxMatrix-class** object with the same dimensions as x

Author(s)

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

See Also

mxMatrix, create.mxMatrix, create.Fmatrix, checkRAM, lavaan2RAM, as.symMatrix

Examples

```r
## a and b are free parameters with starting values and labels
(a1 <- matrix(c(1:4, "5*a", 6, "7*b", 8, 9), ncol=3, nrow=3))
# [,1] [,2] [,3]
# [1,] "1" "4" "7*b"
# [2,] "2" "5*a" "8"
# [3,] "3" "6" "9"

a1 <- as.mxMatrix(a1)

## a and b are fixed parameters without any labels, name="new2"
(a2 <- matrix(1:9, ncol=3, nrow=3))
# [,1] [,2] [,3]
# [1,] 1 4 7
# [2,] 2 5 8
# [3,] 3 6 9

new2 <- as.mxMatrix(a2, name="new2")

## Free parameters without starting values
(a3 <- matrix(c(1:4, "*a", 6, "*b", 8, 9), ncol=3, nrow=3))
# [,1] [,2] [,3]
# [1,] "1" "4" "*b"
# [2,] "2" "*a" "8"
# [3,] "3" "6" "9"

a3 <- as.mxMatrix(a3, lbound=0)

## A free parameter without label
(a4 <- matrix(c(1:4, "5*", 6, "7*b", 8, 9), ncol=3, nrow=3))
# [,1] [,2] [,3]
# [1,] "1" "4" "7*b"
# [2,] "2" "5*" "8"
# [3,] "3" "6" "9"

a4 <- as.mxMatrix(a4)

## Convert a scalar into mxMatrix object
## "name" is required as "3*a" is not a valid name.
(a5 <- as.mxMatrix("3*a", name="a5"))
```
### Free and fixed parameters

```r
(a6 <- matrix(c(1, "2*a", "3@b", 4), ncol=2, nrow=2))

as_mxMatrix(a6)
```

---

**as.symMatrix**

*Convert a Character Matrix with Starting Values to a Character Matrix without Starting Values*

**Description**

It converts a character matrix with starting values to a character matrix without the starting values.

**Usage**

```r
as.symMatrix(x)
```

**Arguments**

- `x`: A character or numeric matrix or a list of character or numeric matrices.

**Details**

If there are non-numeric values in `x`, they are treated as the labels of the free parameters. If a "*" is present, the numeric value on the left-hand side will be treated as the starting value for a free parameter or a fixed value for a fixed parameter. If it is a matrix of numeric values, there are no free parameters in the output matrix. This function removes the starting values and "*" in the matrices.

**Value**

A character matrix.

**Author(s)**

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

**See Also**

`as.mxMatrix`
# Examples

```r
# a and b are free parameters with starting values and labels
(a1 <- matrix(c(1:4, "5*a", 6, "7*b", 8, 9), ncol=3, nrow=3))
# [,1] [,2] [,3]
# [1,] "1" "4" "7*b"
# [2,] "2" "5*a" "8"
# [3,] "3" "6" "9"
```

(as.symMatrix(a1))

```r
# [,1] [,2] [,3]
# [1,] "1" "4" "b"
# [2,] "2" "a" "8"
# [3,] "3" "6" "9"
```

---

## asyCov

**Compute Asymptotic Covariance Matrix of a Correlation/Covariance Matrix**

### Description

It computes the asymptotic sampling covariance matrix of a correlation/covariance matrix under the assumption of multivariate normality.

### Usage

```r
asyCov(x, n, cor.analysis = TRUE, as.matrix = TRUE, 
acov=c("weighted", "individual", "unweighted"), ...)
asyCovOld(x, n, cor.analysis = TRUE, dropNA = FALSE, as.matrix = TRUE, 
acov=c("individual", "unweighted", "weighted"), 
suppressWarnings = TRUE, silent = TRUE, run = TRUE, ...)
```

### Arguments

- **x** A correlation/covariance matrix or a list of correlation/covariance matrices. NA on the variables or other values defined in `na.strings` will be removed before the analysis. Note that it only checks the diagonal elements of the matrices. If there are missing values, make sure that the diagonals are coded with NA or values defined in `na.strings`.
- **n** Sample size or a vector of sample sizes
- **cor.analysis** Logical. The output is either a correlation or covariance matrix.
- **dropNA** Logical. If it is `TRUE`, the resultant dimensions will be reduced by dropping the missing variables. If it is `FALSE`, the resultant dimensions are the same as the input by keeping the missing variables.
- **as.matrix** Logical. If it is `TRUE` and `x` is a list of correlation/covariance matrices with the same dimensions, the asymptotic covariance matrices will be column vectorized and stacked together. If it is `FALSE`, the output will be a list of asymptotic covariance matrices. Note that if it is `TRUE`, `dropNA` will be `FALSE` automatically. This option is useful when passing the asymptotic covariance matrices to `meta`.
If it is individual (the default), the sampling variance-covariance matrices are calculated based on the individual correlation/covariance matrix. If it is either unweighted or weighted, the average correlation/covariance matrix is calculated based on the unweighted or weighted mean with the sample sizes. The average correlation/covariance matrix is used to calculate the sampling variance-covariance matrices.

**Value**

An asymptotic covariance matrix of the vectorized correlation/covariance matrix or a list of these matrices. If `as.matrix` = TRUE and `x` is a list of matrices, the output is a stacked matrix.

**Note**

Before 1.2.6, `asyCov` used an SEM approach based on Cheung and Chan (2004). After 1.2.6, `asyCov` was rewritten based on Olkin and Siotani (1976) for correlation matrix and Yuan and Bentler (2007, p. 371) for covariance matrix. Arguments such as `dropNA`, `suppressWarnings`, `silent`, and `run` were dropped. The original version was renamed to `asyCovOld` for compatibility.

**Author(s)**

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

**References**


**Examples**

```r
## Not run:
C1 <- matrix(c(1,0.5,0.4,0.5,1,0.2,0.4,0.2,1), ncol=3)
asyCov(C1, n=100)

## Data with missing values
C2 <- matrix(c(1,0.4,NA,0.4,1,NA,NA,NA,NA), ncol=3)
C3 <- matrix(c(1,0.2,0.2,1), ncol=2)
```
## Output is a stacked matrix of asymptotic covariance matrices
asyCov(list(C1,C2), n=c(100,50), as.matrix=TRUE)

## Output is a stacked matrix of asymptotic covariance matrices
asyCov(list(C3,C3), n=c(100,50), as.matrix=TRUE)

## Output is a list of asymptotic covariance matrices using the old version
asyCovOld(list(C1,C2,C3), n=c(100,50,50), dropNA=TRUE, as.matrix=FALSE)

## End(Not run)

---

**BCG**

*Dataset on the Effectiveness of the BCG Vaccine for Preventing Tuberculosis*

### Description

This dataset includes 13 studies on the effectiveness of the Bacillus Calmette-Guerin (BCG) vaccine for preventing tuberculosis (see van Houwelingen, Arends, & Stijnen (2002) for details).

### Usage

```r
data(BCG)
```

### Details

A list of data with the following structure:

- **Trial**: Number of the trials
- **Author**: Authors of the original studies
- **Year**: Year of publication
- **VD**: Vaccinated group with disease
- **VWD**: Vaccinated group without the disease
- **NVD**: Not vaccinated group with disease
- **NVWD**: Not vaccinated group without the disease
- **Latitude**: Geographic latitude of the place where the study was done
- **Allocation**: Method of treatment allocation
- **ln_OR**: Natural logarithm of the odds ratio: \( \log((VD/VWD)/(NVD/NVWD)) \)
- **v_ln_OR**: Sampling variance of ln_OR: \( 1/VD+1/VWD+1/NVD+1/NVWD \)
- **ln_Odd_V**: Natural logarithm of the odds of the vaccinated group: \( \log(VD/VWD) \)
- **ln_Odd_NV**: Natural logarithm of the odds of the not vaccinated group: \( \log(NVD/NVWD) \)
- **v_ln_Odd_V**: Sampling variance of ln_Odd_V: \( 1/VD+1/VWD \)
- **cov_V_NV**: Sampling covariance between ln_Odd_V and ln_Odd_NV: It is always 0
- **v_ln_Odd_NV**: Sampling variance of ln_Odd_NV: \( 1/NVD+1/NVWD \)
Source

References


Examples
```r
data(BCG)

## Univariate meta-analysis on the log of the odds ratio
summary( meta(y=ln_OR, v=v_ln_OR, data=BCG,
            x=cbind(scale(Latitude,scale=FALSE),
                   scale(Year,scale=FALSE)) ) )

## Multivariate meta-analysis on the log of the odds
## The conditional sampling covariance is 0
bcg <- meta(y=cbind(ln_Odd_V, ln_Odd_NV), data=BCG,
            v=cbind(v_ln_Odd_V, cov_V_NV, v_ln_Odd_NV))
summary(bcg)
plot(bcg)
```

---

**bdiagMat**

*Create a Block Diagonal Matrix*

**Description**
It creates a block diagonal matrix from a list of numeric or character matrices.

**Usage**
bdiagMat(x)

**Arguments**

- **x** A list of numeric or character matrices (or values)
bdiagRep

Create a Block Diagonal Matrix by Repeating the Input

Description

It creates a block diagonal matrix by repeating the input matrix several times.

Usage

bdiagRep(x, times)

Arguments

x A numeric or character matrix (or values)
times Number of times of x to be repeated
Value

A numeric or character block diagonal matrix

Author(s)

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

See Also

bdiagMat, matrix2bdiag

Examples

```r
## Block diagonal matrix of numerics
bdiagRep( matrix(1:4, nrow=2, ncol=2), 2 )
# [1,] 1 3 0 0
# [2,] 2 4 0 0
# [3,] 0 0 1 3
# [4,] 0 0 2 4

## Block diagonal matrix of characters
bdiagRep( matrix(letters[1:4], nrow=2, ncol=2), 2 )
# [1,] "a" "c" "0" "0"
# [2,] "b" "d" "0" "0"
# [3,] "0" "0" "a" "c"
# [4,] "0" "0" "b" "d"
```

Description

This dataset includes ten studies on the relationships between CSAI subscales and sports behavior. The original data were used in Craft et al. (2003), whereas a subset of them was illustrated in Becker (2009).

Usage

data("Becker09")

Details

A list of data with the following structure:

- **data** A list of 4x4 correlation matrices. The variables are *Performance*, *Cognitive*, *Somatic*, and *Self_confidence*
- **n** A vector of sample sizes
- **Type_of_sport** Samples based on *Individual* or *Team*
Source

References

Examples
```r
## Not run:
data(Becker09)

#### Fixed-effects model
## First stage analysis
fixed1 <- tssem1(Becker09$data, Becker09$n, method="FEM")
summary(fixed1)

## Prepare a regression model using create.mxMatrix()
A1 <- create.mxMatrix(c(0, "0.1*Cog2Per", "0.1*SO2Per", "0.1*SC2Per",
0, 0, 0, 0,
0, 0, 0, 0,
0, "0.1*Cog2SC", "0.1*SO2SC", 0),
type="Full", byrow=TRUE, ncol=4, nrow=4,
as.mxMatrix=FALSE)

## This step is not necessary but it is useful for inspecting the model.
dimnames(A1)[[1]] <- dimnames(A1)[[2]] <- c("Performance", "Cognitive",
"Somatic", "Self_confidence")

## Display A1
A1

S1 <- create.mxMatrix(c("0.1*var_Per",
0, 1,
0, "0.1*cor", 1,
0, 0, 0,"0.1*var_SC"), byrow=TRUE, type="Symm",
as.mxMatrix=FALSE)

## This step is not necessary but it is useful for inspecting the model.
dimnames(S1)[[1]] <- dimnames(S1)[[2]] <- c("Performance", "Cognitive",
"Somatic", "Self_confidence")

## Display S1
S1
```

Performance ~ Cog2Per*Cognitive + SO2Per*Somatic + SC2Per*Self_confidence
Self_confidence ~ Cog2SC*Cognitive + SO2SC*Somatic
## Fix the variances of Cog and SO at 1
Cognitive ~~ 1*Cognitive
Somatic ~~ 1*Somatic
## Label the correlation between Cog and SO
Cognitive ~~ cor*Somatic
## Label the error variances of Per and SC
Performance ~~ var_Per*Performance
Self_confidence ~~ var_SC*Self_confidence"

## Display the model
plot(model, layout="spring")

RAM <- lavaan2RAM(model, obs.variables=c("Performance", "Cognitive", "Somatic", "Self_confidence"))
RAM

A1 <- RAM$A
S1 <- RAM$S

################################################################################
## Second stage analysis
fixed2 <- tssem2(fixed1, Amatrix=A1, Smatrix=S1, diag.constraints=TRUE, intervals.type="LB", model.name="TSSEM2 Becker09", mx.algebras=list( Cog=mxAlgebra(Cog2SC*SC2Per, name="Cog"), SO=mxAlgebra(SO2SC*SC2Per, name="SO"), Cog_SO=mxAlgebra(Cog2SC*SC2Per+SO2SC*SC2Per, name="Cog_SO")))

summary(fixed2)

## Display the model with the parameter estimates
plot(fixed2, layout="spring")

#### Fixed-effects model: with type of sport as cluster
## First stage analysis
cluster1 <- tssem1(Becker09$data, Becker09$n, method="FEM", cluster=Becker09$Type_of_sport)
summary(cluster1)

## Second stage analysis
cluster2 <- tssem2(cluster1, Amatrix=A1, Smatrix=S1, diag.constraints=TRUE, intervals.type="LB", model.name="TSSEM2 Becker09", mx.algebras=list( Cog=mxAlgebra(Cog2SC*SC2Per, name="Cog"), SO=mxAlgebra(SO2SC*SC2Per, name="SO"), Cog_SO=mxAlgebra(Cog2SC*SC2Per+SO2SC*SC2Per, name="Cog_SO")))

summary(cluster2)

## Convert the model to semPlotModel object with 2 plots
## Use the short forms of the variable names
my.plots <- lapply(X=cluster2, FUN=meta2semPlot, manNames=c("Per","Cog","SO","SC") )
## Load the library
library("semPlot")

## Setup two plots
layout(t(1:2))
## The labels are overlapped. We may modify it by using layout="spring"
semPaths(my.plots[[1]], whatLabels="est", nCharNodes=10, color="orange",
layout="spring", edge.label.cex=0.8)
title("Individual sport")
semPaths(my.plots[[2]], whatLabels="est", nCharNodes=10, color="skyblue",
layout="spring", edge.label.cex=0.8)
title("Team sport")

#### Random-effects model
#### First stage analysis
random1 <- tssem1(Becker09$data, Becker09$n, method="REM", RE.type="Diag")
summary(random1)

## Second stage analysis
random2 <- tssem2(random1, Amatrix=A1, Smatrix=S1, diag.constraints=TRUE,
intervals.type="LB", model.name="TSSEM2 Becker09",
mx.algebras=list( Cog=mxAlgebra(Cog2SC*SC2Per, name="Cog"),
SO=mxAlgebra(SO2SC*SC2Per, name="SO"),
Cog_SO=mxAlgebra(Cog2SC*SC2Per+SO2SC*SC2Per,
name="Cog_SO")))

summary(random2)

## Display the model
plot(random2, what="path", layout="spring")

## Display the model with the parameter estimates
plot(random2, layout="spring", color="yellow")

#### Univariate r approach
#### First stage of the analysis
uni1 <- uniR1(Becker09$data, Becker09$n)

#### Second stage of the analysis using OpenMx
#### Regression paths
model2 <- "## Regression paths
Performance ~ Cog2Per*Cognitive + SO2Per*Somatic + SC2Per*Self_confidence
Self_confidence ~ Cog2SC*Cognitive + SO2SC*Somatic
## Provide starting values for Cog and SO
Cognitive ~~ start(1)*Cognitive
Somatic ~~ start(1)*Somatic
## Label the correlation between Cog and SO
Cognitive ~~ cor*Somatic
## Label the error variances of Per and SC
Performance ~~ var_Per*Performance
Self_confidence ~~ var_SC*Self_confidence"

RAM2 <- lavaan2RAM(model2, obs.variables=c("Performance", "Cognitive",
"Self_confidence", "Cognitive", "Somatic")

## Display the model
plot(model2, what="RAM", layout="spring")

summary(random2)

## Display the model with the parameter estimates
plot(random2, layout="spring", color="yellow")
Becker83

Studies on Sex Differences in Conformity Reported by Becker (1983)

Description

The data set includes studies on sex differences in conformity using the fictitious norm group paradigm reported by Becker (1983).

Usage

data(Becker83)

Details

The variables are:

<table>
<thead>
<tr>
<th>study</th>
<th>study number</th>
</tr>
</thead>
<tbody>
<tr>
<td>di</td>
<td>Standardized mean difference</td>
</tr>
<tr>
<td>vi</td>
<td>Sampling variance of the effect size</td>
</tr>
<tr>
<td>percentage</td>
<td>Percentage of male authors</td>
</tr>
<tr>
<td>items</td>
<td>Number of items</td>
</tr>
</tbody>
</table>

Source


References

**Examples**

```r
data(Becker83)

## Random-effects meta-analysis
summary( meta(y=di, v=vi, data=Becker83) )

## Mixed-effects meta-analysis with log(items) as the predictor
summary( meta(y=di, v=vi, x=log(items), data=Becker83) )
```

---

**Description**

This data set includes six studies of correlation matrices reported by Becker (1992; 1995).

**Usage**

```r
data(Becker92)
```

**Details**

A list of data with the following structure:

- `data` A list of 6 studies of correlation matrices. The variables are *Math* (math aptitude), *Spatial* (spatial ability), and *Verbal* (verbal ability)
- `n` A vector of sample sizes

**Source**


**Examples**

```r
## Not run:
data(Becker92)

#### Fixed-effects model
## First stage analysis
## Replicate Becker's (1992) analysis using 4 studies only
fixed1 <- tssem1(Becker92$data[1:4], Becker92$n[1:4], method="FEM")
summary(fixed1)

## Prepare a regression model using create.mxMatrix()
```
A1 <- create.mxMatrix(c(0,0,0,"0.2*Spatial2Math", 0,0,"0.2*Verbal2Math",0,0), type="Full", ncol=3, nrow=3, as.mxMatrix=FALSE)
var.names <- c("Math_aptitude","Spatial","Verbal")

## This step is not necessary but it is useful for inspecting the model.
## dimnames(A1)[[1]] <- dimnames(A1)[[2]] <- var.names

## Display A1
A1

S1 <- create.mxMatrix(c("0.2*ErrorVarMath",0,0,1,"0.2*CorSpatialVerbal",1),
type="Symm", as.mxMatrix=FALSE)

## This step is not necessary but it is useful for inspecting the model.
## dimnames(S1)[[1]] <- dimnames(S1)[[2]] <- var.names

## Display S1
S1

## Alternative model specification in lavaan model syntax
model <- "## Regression paths
Math ~ Spatial2Math*Spatial + Verbal2Math*Verbal
Spatial ~~ CorSpatialVerbal*Verbal
## Fix the variances of Spatial and Verbal at 1
Spatial ~~ 1*Spatial
Verbal ~~ 1*Verbal
## Label the error variance of Math
Math ~~ ErrorVarMath*Math + start(0.2)*Math"

## Display the model
plot(model)

RAM <- lavaan2RAM(model, obs.variables=c("Math", "Spatial", "Verbal"))

## Fixed-effects model: Second stage analysis
## Two equivalent versions to calculate the R2 and its 95% LBCI
fixed2 <- tssem2(fixed1, RAM=RAM, intervals.type="LB",
mx.algebras=list(R1=mxAlgebra(Spatial2Math^2+Verbal2Math^2+2*CorSpatialVerbal*Spatial2Math*Verbal2Math, name="R1"),
R2=mxAlgebra(One-Smatrix[1,1], name="R2"),
One=mxMatrix("Iden", ncol=1, nrow=1, name="One")))

summary(fixed2)

## Display the model with the parameter estimates
plot(fixed2)

## Random-effects model
```
## First stage analysis
## No random effects for off-diagonal elements
random1 <- tssem1(Becker92$data, Becker92$n, method="REM", RE.type="Diag")
summary(random1)

## Random-effects model: Second stage analysis
random2 <- tssem2(random1, RAM=RAM)
summary(random2)

## Display the model with the parameter estimates
plot(random2, color="yellow")

#### Similar to conventional fixed-effects GLS approach
## First stage analysis
## No random effects
## Replicate Becker's (1992) analysis using 4 studies only
gls1 <- tssem1(Becker92$data[1:4], Becker92$n[1:4], method="REM", RE.type="Zero", model.name="Fixed effects GLS Stage 1")
summary(gls1)

gls2 <- tssem2(gls1, RAM=RAM, model.name="Fixed effects GLS Stage 2")
summary(gls2)

## End(Not run)
```

---

**Becker94**

*Five Studies of Ten Correlation Matrices reported by Becker and Schram (1994)*

**Description**

This data set includes five studies of ten correlation matrices reported by Becker and Schram (1994).

**Usage**

```r
data(Becker94)
```

**Details**

A list of data with the following structure:

- **data** A list of 10 correlation matrices. The variables are *Math* (math aptitude), *Spatial* (spatial ability), and *Verbal* (verbal ability)
- **n** A vector of sample sizes
- **gender** *Females* or *Males* samples
Source

Examples

```r
## Not run:
data(Becker94)

#### Fixed-effects model
## First stage analysis
fixed1 <- tssem1(Becker94$data, Becker94$n, method="FEM")
summary(fixed1)

## Prepare a regression model using create.mxMatrix()
## A1 <- create.mxMatrix(c(0,0,0,"0.2*Spatial2Math",
## 0,0,"0.2*Verbal2Math",0,0), type="Full",
## ncol=3, nrow=3, name="A1")
## S1 <- create.mxMatrix(c("0.2*ErrorVarMath",0,0,1,
## "0.2*CorBetweenSpatialVerbal",1),
## type="Symm", name="S1")

## An alternative method to create a regression model with the lavaan syntax
model <- "## Regression model
Math ~ Spatial2Math*Spatial + Verbal2Math*Verbal
## Error variance of Math
Math ~~ ErrorVarMath*Math
## Variances of Spatial and Verbal fixed at 1.0
Spatial ~~ 1*Spatial
Verbal ~~ 1*Verbal
## Correlation between Spatial and Verbal
Spatial ~~ CorBetweenSpatialVerbal*Verbal"

## Display the model
plot(model)

RAM <- lavaan2RAM(model, obs.variables=c("Math", "Spatial", "Verbal"))
RAM

## Second stage analysis
## A1 <- RAM$A
## S1 <- RAM$S
## fixed2 <- tssem2(fixed1, Amatrix=A1, Smatrix=S1, intervals.type="LB")

## Display the model with the parameter estimates
plot(fixed2)

#### Fixed-effects model: with gender as cluster
```
## First stage analysis

```r
cluster1 <- tssem1(Becker94$data, Becker94$n, method="FEM", cluster=Becker94$gender)
summary(cluster1)
```

## Second stage analysis

```r
cluster2 <- tssem2(cluster1, RAM=RAM, intervals.type="LB")
summary(cluster2)
```

#### Conventional fixed-effects GLS approach

## First stage analysis

## No random effects

## Replicate Becker's (1992) analysis using 4 studies only

```r
gls1 <- tssem1(Becker92$data[1:4], Becker92$n[1:4], method="REM", RE.type="Zero",
               model.name="Fixed effects GLS Stage 1")
summary(gls1)
```

## Fixed-effects GLS model: Second stage analysis

```r
gls2 <- tssem2(gls1, RAM=RAM, intervals.type="LB",
               model.name="Fixed effects GLS Stage 2")
summary(gls2)
```

## End(Not run)

---

**Five Published Trails from Berkey et al. (1998)**

**Description**

The data set includes five published trials, reported by Berkey et al. (1998), comparing surgical and non-surgical treatments for medium-severity periodontal disease, one year after treatment.

**Usage**

data(Berkey98)

**Details**

The variables are:

- **trial** Trial number
- **pub_year** Publication year
- **no_of_patients** Number of patients
- **PD** Patient improvements (mm) in *probing depth*
- **AL** Patient improvements (mm) in *attachment level*
- **var_PD** Sampling variance of PD
- **cov_PD_AL** Sampling covariance between PD and AD
- **var_AL** Sampling variance of AL
Source


Examples

```r
## Not run:
data(Berkey98)

#### ML estimation method
## Multivariate meta-analysis
x <- meta(y=cbind(PD, AL), v=cbind(var_PD, cov_PD_AL, var_AL), data=Berkey98)
x <- rerun(x)
summary(x)
plot(x)

## Plot individual studies proportional to the weights
plot(x, study.weight.plot=TRUE)

## Include forest plot from the metafor package
library(metafor)
plot(x, diag.panel=TRUE, main="Multivariate meta-analysis",
 axis.label=c("PD", "AL"))
forest( rma(yi=PD, vi=var_PD, data=Berkey98) )
title("Forest plot of PD")
forest( rma(yi=AL, vi=var_AL, data=Berkey98) )
title("Forest plot of AL")

#### REML estimation method
## Multivariate meta-analysis
summary( reml(y=cbind(PD, AL), v=cbind(var_PD, cov_PD_AL, var_AL),
 data=Berkey98,
 RE.lbound=NA) )

#### REML estimation method
## Multivariate meta-analysis with "publication year-1979" as the predictor
summary( reml(y=cbind(PD, AL), v=cbind(var_PD, cov_PD_AL, var_AL),
 RE.constraints=Diag(c("1e-5*Tau2_1_1", "1e-5*Tau2_2_2")),
 x=scale(pub_year, center=1979), data=Berkey98) )
```
Boer16

Correlation Matrices from Boer et al. (2016)

Description

The data set includes correlation matrices of leader-member exchange in transformational leadership reported by Boer et al. (2016).

Usage

data(Boer16)

Details

A list of data with the following structure:

- **data**: A list of correlation matrices. The variables are LMX (leader-member exchange), TFL (transformational leadership), JS (job satisfaction), OC (organizational commitment), and LE (leader effectiveness).
- **n**: A vector of sample sizes.
- **RelLMX**: The reliability of LMX.
- **RelTFL**: The reliability of TFL.

Source


Examples

```r
## Not run:
## Stage 1 analysis
rand1 <- tssem1(Boer16$data, Boer16$n, method="REM", RE.type="Diag",
                acov="weighted")
summary(rand1)

## Stage 2 analysis
model2a <- 'JS+OC+LE ~ LMX+TFL
LMX ~ TFL
## Variance of TFL is fixed at 1
TFL ~~ 1*TFL
## Correlated residuals
JS ~~ OC
JS ~~ LE
OC ~~ LE'
```
## Display the model
plot(model2a)

RAM2a <- lavaan2RAM(model2a, obs.variables = c("LMX", "TFL", "JS", "OC", "LE"),
A.notation="on", S.notation="with")

rand2a <- tssem2(rand1, Amatrix=RAM2a$A, Smatrix=RAM2a$S)
summary(rand2a)

## Display the model with the parameter estimates
plot(rand2a, layout="spring")

## End(Not run)

---

**bootuniR1**  
*Parametric bootstrap on the univariate R (uniR) object*

### Description
It generates correlation matrices with the parametric bootstrap on the univariate R (uniR) object.

### Usage

```r
bootuniR1(x, Rep, nonPD.pop=c("replace", "nearPD", "accept"))
```

### Arguments

- **x**: An object of class `uniR`
- **Rep**: Number of replications of the parametric bootstrap
- **nonPD.pop**: If it is replace, generated non-positive definite matrices are replaced by generated new ones which are positive definite. If it is nearPD, they are replaced by nearly positive definite matrices by calling `Matrix::nearPD()`. If it is accept, they are accepted.

### Details
This function implements the parametric bootstrap approach suggested by Yu et al. (2016). It is included in this package for research interests. Please refer to Cheung (2018) for the issues associated with this parametric bootstrap approach.

### Value
An object of the generated correlation matrices.

### Author(s)
Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>
References


See Also

*rCor, bootuniR2, Nohe15*

bootuniR2  
Fit Models on the bootstrapped correlation matrices

Description

It fits structural equation models on the bootstrapped correlation matrices.

Usage

`bootuniR2(model, data, n, ...)`

Arguments

- `model` A model in `sem` syntax.
- `data` A list of correlation matrices.
- `n` Sample size in fitting the structural equation models
- `...` Further arguments to be passed to `sem`.

Details

This function fits the lavaan model with the bootstrapped correlation matrices. It implements the parametric bootstrap approach suggested by Yu et al. (2016). It is included in this package for research interests. Please refer to Cheung (2018) for the issues associated with this parametric bootstrap approach.

Value

A list of the fitted object from `sem`.

Author(s)

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>
References


See Also

bootuniR2, tssemParaVar, Nohe15

---

**Bornmann07**

**A Dataset from Bornmann et al. (2007)**

**Description**

A dataset from Bornmann et al. (2007) for three-level meta-analysis.

**Usage**

`data(Bornmann07)`

**Details**

The variables are:

- **ID** ID of the study
- **Study** Study name
- **Cluster** Cluster for effect sizes
- **logOR** Effect size: log odds ratio
- **v** Sampling variance of logOR
- **Year** Year of publication
- **Type** Type of proposal: either *Grant* or *Fellowship*
- **Discipline** Discipline of the proposal: either *Physical sciences, Life sciences/biology, Social sciences/humanities* or *Multidisciplinary*
- **Country** Country of the proposal: either the *United States, Canada, Australia, United Kingdom* or *Europe*

**Source**

References


Examples

```r
## Not run:
data(Bornmann07)

#### ML estimation method
## No predictor
summary( meta3L(y=logOR, v=v, cluster=Cluster, data=Bornmann07) )

## Type as a predictor
## Grant: 0
## Fellowship: 1
summary( meta3L(y=logOR, v=v, x=(as.numeric(Type)-1),
cluster=Cluster, data=Bornmann07) )

## Centered Year as a predictor
summary( meta3L(y=logOR, v=v, x=scale(Year, scale=FALSE),
cluster=Cluster, data=Bornmann07) )

#### REML estimation method
## No predictor
summary( reml3L(y=logOR, v=v, cluster=Cluster, data=Bornmann07) )

## Type as a predictor
## Grants: 0
## Fellowship: 1
summary( reml3L(y=logOR, v=v, x=(as.numeric(Type)-1),
cluster=Cluster, data=Bornmann07) )

## Centered Year as a predictor
summary( reml3L(y=logOR, v=v, x=scale(Year, scale=FALSE),
cluster=Cluster, data=Bornmann07) )

## Handling missing covariates with FIML
## MCAR
## Set seed for replication
set.seed(1000000)

## Copy Bornmann07 to my.df
my.df <- Bornmann07

## "Fellowship": 1; "Grant": 0
my.df$Type_MCAR <- ifelse(Bornmann07$Type=="Fellowship", yes=1, no=0)

## Create 17 out of 66 missingness with MCAR
```
my.df$Type_MCAR[sample(1:66, 17)] <- NA
summary(meta3LFIML(y=logOR, v=v, cluster=Cluster, x2=Type_MCAR, data=my.df))

## MAR
Type_MAR <- ifelse(Bornmann07$Type=="Fellowship", yes=1, no=0)

## Create 27 out of 66 missingness with MAR for cases Year<1996
index_MAR <- ifelse(Bornmann07$Year<1996, yes=TRUE, no=FALSE)
Type_MAR[index_MAR] <- NA

## Include auxiliary variable
summary(meta3LFIML(y=logOR, v=v, cluster=Cluster, x2=Type_MAR, av2=Year, data=my.df))

## End(Not run)

---

**calEffSizes**

*Calculate Effect Sizes using lavaan Models*

**Description**

It calculates effect sizes with Delta Method by formulating the effect sizes as functions of SEM in lavaan.

**Usage**

```
calEffSizes(model, data=NULL, n, Cov, Mean=NULL, group=NULL, lavaan.output=FALSE, ...)
```

**Arguments**

- `model` A lavaan model. Effect sizes are defined as functions of SEM parameters with `:=`.
- `data` A data frame of the observed variables. If it is `NULL`, summary statistics are required.
- `n` Sample sizes
- `Cov` A covariance matrix or a list of covariance matrices.
- `Mean` Optional sample means.
- `group` A character of the variable name in the data frame defining the groups in a multiple group analysis.
- `lavaan.output` If `TRUE`, it returns the fitted object instead of the effect sizes and their sampling covariance matrix.
- `...` Further arguments passed to `sem`.

**Value**

Effect sizes and their sampling covariance matrix or a lavaan fitted object.
Note

The input matrices are treated as covariance matrices unless there are explicit constraints in the model.

Author(s)

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

References


See Also

smdMES, smdMTS

Examples

```r
## Not run:
## Select ATT, Bi, and BEH
obs.vars <- c("BEH", "BI", "ATT")

## Select one study from Cooke16 for illustration
my.cor <- Cooke16$data[[4]][obs.vars, obs.vars]
my.n <- Cooke16$n[4]

## Effect sizes: indirect effect and direct effect
model <- "BEH ~ c*ATT + b*BI
  BI ~ a*ATT
  ## Indirect effect
  Ind := a*b
  Dir := c"

calEffSizes(model=model, n=my.n, Cov=my.cor, lavaan.output=FALSE)

## Return the lavaan fitted model
fit <- calEffSizes(model=model, n=my.n, Cov=my.cor, lavaan.output=TRUE)
lavaan::summary(fit)
lavaan::parameterestimates(fit)

## End(Not run)
```
checkRAM

Check the correctness of the RAM formulation

Description

It provides simple checks on the correctness of the RAM formulation.

Usage

checkRAM(Amatrix, Smatrix, cor.analysis=TRUE)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amatrix</td>
<td>An asymmetric matrix in the RAM specification with \textit{MxMatrix-class}. If it is a matrix, it will be converted into \textit{MxMatrix-class} by the \texttt{as.mxMatrix} function.</td>
</tr>
<tr>
<td>Smatrix</td>
<td>A symmetric matrix in the RAM specification with \textit{MxMatrix-class}. If it is a matrix, it will be converted into \textit{MxMatrix-class} by the \texttt{as.mxMatrix} function.</td>
</tr>
<tr>
<td>cor.analysis</td>
<td>Logical. Analysis of correlation or covariance structure. There are additional checks for cor.analysis=TRUE.</td>
</tr>
</tbody>
</table>

Value

It returns silently if no error has been detected; otherwise, it returns a warning message.

Author(s)

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

See Also

\texttt{as.mxMatrix}, \texttt{lavaan2RAM}

Examples

```r
## Not run:
## Digman97 example
model1 <- "## Factor loadings
Alpha=~A+C+ES
Beta=~E+I
## Factor correlation
Alpha~~Beta"

RAM1 <- lavaan2RAM(model1, obs.variables=c("A","C","ES","E","I"),
                      A.notation="on", S.notation="with")

checkRAM(Amatrix=RAM1$A, Smatrix=RAM1$S)
```
## Hunter83 example

model2 <- "## Regression paths
Job_knowledge ~ A2J*Ability
Work_sample ~ A2W*Ability + J2W*Job_knowledge
Supervisor ~ J2S*Job_knowledge + W2S*Work_sample

## Fix the variance of Ability at 1
Ability ~~ 1*Ability

## Label the error variances of the dependent variables
Job_knowledge ~~ VarE_J*Job_knowledge
Work_sample ~~ VarE_W*Work_sample
Supervisor ~~ VarE_S*Supervisor"

RAM2 <- lavaan2RAM(model2, obs.variables=c("Ability","Job_knowledge", "Work_sample","Supervisor"))

## The model is okay.
checkRAM(Amatrix=RAM2$A, Smatrix=RAM2$S)

## End(Not run)

---

### Description

This data set includes fifty studies of correlation matrices on the theory of planned theory reported by Cheung and Chan (2000).

### Usage

data(Cheung00)

### Details

A list of data with the following structure:

- **data** A list of 50 studies of correlation matrices. The variables are the attitude toward behavior `att`, subjective norm `sn`, behavioral intention `bi`, and behavior `beh`
- **n** A vector of sample sizes

### Note

These studies were extracted from the original data set for illustration purpose. Some samples contained two or more correlation matrices, and only one of them was arbitrarily selected to avoid the problem of dependence. Moreover, studies with less than 3 correlation coefficients were also excluded.
Source

References

Examples
```r
## Not run:
data(Cheung00)

## Variable labels
labels <- colnames(Cheung00$data[[1]])

## Full mediation model
S <- create.mxMatrix(c("1",
                      ".2* cov_att_sn", "1",
                      0, 0, ".2* e_bi",
                      0, 0, 0, ".2* e_beh"),
                      type="Symm", as.mxMatrix=FALSE, byrow=TRUE)
dimnames(S) <- list(labels, labels)
S

A <- matrix(c("0", ".2* att2bi", ".2* sn2bi", "0",
              ".2*bi2beh", "0", ".2*bi2beh", ".2*bi2beh",
              "0", "0", "0", "0"), 
              byrow=TRUE, 4, 4)
dimnames(A) <- list(labels, labels)
A

#### Random-effects model

## Stage 1 analysis
random_1 <- tssem1(Cheung00$data, Cheung00$n, method="REM", RE.type="Diag", 
                    acov="weighted")
summary(random_1)

## Stage 2 analysis
random_2 <- tssem2(random_1, Amatrix=A, Smatrix=S, intervals.type="LB", 
                    diag.constraints=TRUE)
summary(random_2)

## Display the model
plot(random_2, what="path")

## Display the model with the parameter estimates
plot(random_2, color="yellow")
```
## Load the library
library("semPlot")

## End(Not run)

---

### Description

Four studies were selected from the data set used by Cheung and Chan (2005; 2009). Some variables were randomly deleted to illustrate the analysis with missing data.

### Usage

data(Cheung09)

### Details

A list of data with the following structure:

- **data** A list of 4 studies of correlation matrices
- **n** A vector of sample sizes

### Source


### References


### Examples

```r
## Not run:
data(Cheung09)

#### Fixed-effects model: Stage 1 analysis
fixed1 <- tssem1(Cheung09$data, Cheung09$n, method= "FEM")
summary(fixed1)

## Prepare a model implied matrix
## Factor correlation matrix
```
Phi <- create.mxMatrix( c("0.3*corf2f1","0.3*corf3f1","0.3*corf3f2"),
                      type="Stand", as.mxMatrix=FALSE )

## Error variances
Psi <- create.mxMatrix( paste("0.2*e", 1:9, sep=""), type="Diag",
                      as.mxMatrix=FALSE )

## Create Smatrix
S1 <- bdiagMat(list(Psi, Phi))
## dimnames(S1)[[1]] <- dimnames(S1)[[2]] <- c(paste("x",1:9,sep=""),
##                                paste("f",1:3,sep=""))
## S1
S1 <- as.mxMatrix(S1)

## Factor loadings
Lambda <- create.mxMatrix( c("0.3*f1x1","0.3*f1x2","0.3*f1x3",rep(0,9),
                             "0.3*f2x4","0.3*f2x5","0.3*f2x6","0.3*f2x7",
                             rep(0,9),"0.3*f3x8","0.3*f3x9"), type="Full",
                             ncol=3, nrow=9, as.mxMatrix=FALSE )
Zero1 <- matrix(0, nrow=9, ncol=9)
Zero2 <- matrix(0, nrow=3, ncol=12)

## Create Amatrix
A1 <- rbind( cbind(Zero1, Lambda),
            Zero2 )
## dimnames(A1)[[1]] <- dimnames(A1)[[2]] <- c(paste("x",1:9,sep=""),
##                                paste("f",1:3,sep=""))
## A1
A1 <- as.mxMatrix(A1)

## Create Fmatrix
F1 <- create.Fmatrix(c(rep(1,9), rep(0,3)))

#### Fixed-effects model: Stage 2 analysis
fixed2 <- tssem2(fixed1, Amatrix=A1, Smatrix=S1, Fmatrix=F1,
                 intervals.type="LB")
summary(fixed2)

## Display the model
plot(fixed2, what="path")

## Display the model with the parameter estimates
plot(fixed2, latNames=c("f1", "f2", "f3"), edge.label.cex=0.8,
     color="yellow")

## End(Not run)

## Extract Parameter Estimates from various classes.
coef

Description

It extracts the parameter estimates from objects of various classes.
Usage

```r
## S3 method for class 'tssem1FEM'
coef(object, ...)
## S3 method for class 'tssem1FEM.cluster'
coef(object, ...)
## S3 method for class 'tssem1REM'
coef(object, select = c("all", "fixed", "random"), ...)
## S3 method for class 'wls'
coef(object, ...)
## S3 method for class 'wls.cluster'
coef(object, ...)
## S3 method for class 'meta'
coef(object, select = c("all", "fixed", "random"), ...)
## S3 method for class 'meta3LFIML'
coef(object, select = c("all", "fixed", "random", "allX"), ...)
## S3 method for class 'reml'
coef(object, ...)
## S3 method for class 'MxRAMModel'
coef(object, ...)
## S3 method for class 'osmasem'
coef(object, select = c("fixed", "all", "random"), ...)
```

Arguments

- `object` An object returned from either class `tssem1FEM`, `tssem1FEM.cluster`, `tssem1REM`, `class wls`, `class wls.cluster`, `class meta`, `class reml` or `class MxRAMModel`
- `select` Select all for both fixed- and random-effects parameters, `fixed` for the fixed-effects parameters or `random` for the random-effects parameters. For `meta3LFIML` objects, `allX` is used to extract all parameters including the predictors and auxiliary variables.
- `...` Further arguments; currently none is used

Value

Parameter estimates for both fixed-effects (if any) and random-effects (if any)

Note

`coef.MxRAMModel` is simply a wrapper of `omxGetParameters`. Extra arguments will be passed to it

Author(s)

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

See Also

tsseml, wls, meta, reml, omxGetParameters, osmasem
Examples

```r
## Random-effects meta-analysis
model1 <- meta(y=yi, v=vi, data=Hox02)
coef(model1)

## Fixed-effects only
coef(model1, select="fixed")
```

Cooke16  
*Correlation Matrices from Cooke et al. (2016)*

Description

The data set includes correlation matrices on using the theory of planned behavior to predict alcohol consumption reported by Cooke et al. (2016).

Usage

```r
data(Cooke16)
```

Details

A list of data with the following structure:

- `data` A list of correlation matrices. The variables are `SN` (subjective norm), `ATT` (attitude), `PBC` (perceived behavior control), `BI` (behavioral intention), and `BEH` (behavior).
- `n` A vector of sample sizes.
- `MeanAge` Mean age of the participants except for Ajzen and Sheikh (2013), which is the median age, and Glassman, et al. (2010a) to Glassman, et al. (2010d), which are based on the range of 18 to 24.
- `Female` Percentage of female participants.

Source


References

## Examples

```
## Not run:
## Check whether the correlation matrices are valid (positive definite)
Cooke16$data[is.pd(Cooke16$data)==FALSE]

## Since the correlation matrix in Study 3 is not positive definite,
## we exclude it in the following analyses
my.data <- Cooke16$data[-3]
my.n <- Cooke16$n[-3]

## Show the no. of studies per correlation
pattern.na(my.data, show.na = FALSE)

## Show the total sample sizes per correlation
pattern.n(my.data, my.n)

## Stage 1 analysis
## Random-effects model
random1 <- tssem1(my.data, my.n, method="REM", RE.type="Diag", acov="weighted")
summary(random1)

A1 <- create.mxMatrix(c(0,0,0,0,0,
0,0,0,0,0,
0,0,0,0,0,
"0.2*SN2BI","0.2*ATT2BI","0.2*PBC2BI",0,0,
0,0,"0.2*PBC2BEH","0.2*BI2BEH",0),
type="Full", ncol=5, nrow=5,
byrow=TRUE, as.mxMatrix=FALSE)

## This step is not necessary but it is useful for inspecting the model.
dimnames(A1)[[1]] <- dimnames(A1)[[2]] <- colnames(Cooke16$data[[1]])

## Display A1
A1

S1 <- create.mxMatrix(c(1,
"0.1*ATT_SN", 1,
"0.1*PBC_SN", "0.1*PBC_ATT", 1,
0, 0, 0, "0.5*VarBI",
0, 0, 0, 0, "0.5*VarBEH"),
type = "Symm", ncol=5, nrow=5,
byrow=TRUE, as.mxMatrix=FALSE)

dimnames(S1)[[1]] <- dimnames(S1)[[2]] <- colnames(Cooke16$data[[1]])
S1

## Stage 2 analysis
random2 <- tssem2(random1, Amatrix=A1, Smatrix=S1, diag.constraints=FALSE,
intervals.type="LB")
summary(random2)

## Display the model
```
```r
plot(random2, what="path")

## Display the model with the parameter estimates
plot(random2, color="yellow")

## End(Not run)
```

---

**Cooper03** *Selected effect sizes from Cooper et al. (2003)*

---

**Description**

Fifty-six effect sizes from 11 districts from Cooper et al. (2003) were reported by Konstantopoulos (2011).

**Usage**

data(Cooper03)

**Details**

The variables are:

- **District** District ID
- **Study** Study ID
- **y** Effect size
- **v** Sampling variance
- **Year** Year of publication

**Source**


**References**

Examples

```r
## Not run:
data(Cooper03)

#### ML estimation method
## No predictor
summary( model1 <- meta3L(y=y, v=v, cluster=District, data=Cooper03) )

#### REML estimation method
## No predictor
summary( reml3L(y=y, v=v, cluster=District, data=Cooper03) )

#### Handling missing covariates with FIML
## Create 20/56 MCAR data in Year
set.seed(10000)
Year_MCAR <- Cooper03$Year
Year_MCAR[sample(56, 20)] <- NA
summary( reml3LFIML(y=y, v=v, cluster=District, x2=scale(Year_MCAR, scale=FALSE),
data=Cooper03, model.name="NA in Year_MCAR") )
```

Cor2DataFrame

Convert correlation or covariance matrices into a dataframe of correlations or covariances with their sampling covariance matrices
Description

It converts the correlation or covariance matrices into a dataframe of correlations or covariances with their asymptotic sampling covariance matrices. It uses the asyCov at the backend.

Usage

Cor2DataFrame(x, n, v.na.replace = TRUE, row.names.unique = FALSE, cor.analysis = TRUE, acov="weighted", append.vars=TRUE, asyCovOld=FALSE, ...)

Arguments

x A list of data with correlation/covariance matrix in x$data and sample sizes x$n. Additional variables in x can be attached.

n If x is a list of correlation matrices without x$data and x$n, a vector of sample sizes n must be provided.

v.na.replace Logical. Missing value is not allowed in definition variables. If it is TRUE (the default), missing value is replaced by a large value (1e10). These values are not used in the analysis.

row.names.unique Logical, If it is FALSE (the default), unique row names are not created.

cor.analysis Logical. The output is either a correlation or covariance matrix.

acov If it is weighted, the average correlation/covariance matrix is calculated based on the weighted mean with the sample sizes. The average correlation/covariance matrix is used to calculate the sampling variance-covariance matrices.

append.vars Whether to append the additional variables to the output dataframe.

asyCovOld Whether to use the old version of asyCov. See asyCov.

... Further arguments to be passed to asyCov.

Value

A list of components: (1) a data frame of correlations or covariances with their sampling covariance matrices; (2) a vector of sample sizes; (3) labels of the correlations; and (3) labels of their sampling covariance matrices.

Author(s)

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

See Also

asyCov, osmasem, create.vechsR, create.Tau2, create.V
create.Fmatrix

Create an F matrix to select observed variables

Description

It creates an F matrix to select observed variables for wls function.

Usage

create.Fmatrix(x, name, as.mxMatrix=TRUE, ...)

Arguments

x A vector of logical type
name Name of the matrix. If it is missing, "Fmatrix" will be used.
as.mxMatrix Logical. If it is TRUE, the output is a matrix of MxMatrix-class. If it is FALSE, it is a numeric matrix.
... Not used.

Author(s)

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>
create.modMatrix

Create a moderator matrix used in OSMASEM

Description

It creates a moderator matrix used in OSMASEM.

Usage

create.modMatrix(RAM, output=c("A", "S"), mod)

Arguments

RAM
output
mod

A RAM object including a list of matrices of the model returned from lavaan2RAM.
Whether the output is an "A" or "S" matrix.
A string of moderator in the dataset.

Value

A character matrix.
create.mxMatrix

Author(s)

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

Examples

```r
## A multiple regression model
model <- "y ~ x1 + x2
           x1 ~~ 1*x1
           x2 ~~ 1*x2
           x1 ~~ x2"

## RAM specification
RAM <- lavaan2RAM(model, obs.variables=c("y", "x1", "x2"))

## Create a moderator matrix on A with "meanAge as the moderator.
A1 <- create.modMatrix(RAM=RAM, output="A", mod="meanAge")
A1

## Create a moderator matrix on S with "meanAge as the moderator.
S1 <- create.modMatrix(RAM=RAM, output="S", mod="meanAge")
S1
```

create.mxMatrix  

Create a Vector into MxMatrix-class

Description

It converts a vector into MxMatrix-class via mxMatrix.

Usage

```r
create.mxMatrix(x, type=c("Full","Symm","Diag","Stand"), ncol=NA,
                 nrow=NA, as.mxMatrix=TRUE, byrow=FALSE, ...)
```

Arguments

- **x**: A character or numeric vector
- **type**: Matrix type similar to those listed in mxMatrix
- **ncol**: The number of columns. It is necessary when type="Full". It is ignored and determined by the length of x for the other types of matrices.
- **nrow**: The number of rows. It is necessary when type="Full". It is ignored and determined by the length of x for the other types of matrices.
- **as.mxMatrix**: Logical. If it is TRUE, the output is a matrix of MxMatrix-class. If it is FALSE, it is a numeric matrix.
- **byrow**: Logical. If FALSE (the default) the matrix is filled by columns, otherwise the matrix is filled by rows.
- **...**: Further arguments to be passed to mxMatrix. Please note that type, nrow, ncol, values, free and labels will be created automatically. Thus, these arguments except labels should be avoided in ...
create.mxModel

Create an mxModel

Description

It creates a mxModel from a RAM object.

Details

If there are non-numeric values in x, they are treated as the labels of the free parameters. If an "*" is present, the numeric value on the left-hand side will be treated as the starting value for a free parameter or a fixed value for a fixed parameter. If it is a matrix of numeric values, there are no free parameters in the output matrix. nrow and ncol will be calculated from the length of x unless type="Full" is specified.

Value

A MxMatrix-class object with the same dimensions as x

Author(s)

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

See Also

mxMatrix, create.mxMatrix, create.Fmatrix

Examples

## a and b are free parameters with starting values and labels
(a1 <- c(1:4, "5*a", 6, "7*b", 8, 9))
(mat1 <- create.mxMatrix(a1, ncol=3, nrow=3, name="mat1"))

## Arrange the elements by row
(mat2 <- create.mxMatrix(a1, ncol=3, nrow=3, as.mxMatrix=FALSE, byrow=TRUE))
(a3 <- c(1:3, "4*f4", "5*f5", "6*f6"))
(mat3 <- create.mxMatrix(a3, type="Symm", name="mat3"))

## Create character matrix
(mat4 <- create.mxMatrix(a3, type="Symm", as.mxMatrix=FALSE))

## Arrange the elements by row
(mat5 <- create.mxMatrix(a3, type="Symm", as.mxMatrix=FALSE, byrow=TRUE))
(mat6 <- create.mxMatrix(a3, type="Diag", lbound=6:1, name="mat6"))
create.mxModel

Usage

create.mxModel(model.name="mxModel", RAM=NULL, data=NULL,
    Cov=NULL, means=NULL, numObs,
    intervals.type = c("z", "LB"), startvalues=NULL,
    mxModel.Args=NULL, run=TRUE, mxTryHard=FALSE,
    silent=TRUE, ...)

Arguments

model.name A string for the model name in mxModel.
RAM A RAM object including a list of matrices of the model returned from lavaan2RAM.
data A data frame or matrix of data.
Cov A covariance matrix may also be used if data==NULL.
means A named vector of means (options) if Cov is used.
numObs If Cov is used, a sample size must be provided.
intervals.type Either z (default if missing) or LB. If it is z, it calculates the 95% confidence intervals (CIs) based on the estimated standard error. If it is LB, it calculates the 95% likelihood-based CIs on the parameter estimates.
startvalues A list of starting values for the free parameters.
mxModel.Args A list of arguments passed to mxModel.
run Logical. If FALSE, only return the mx model without running the analysis.
mxTryHard If TRUE, call mxTryHard to run the analysis. If FALSE, call mxRun to run the analysis.
silent Logical. An argument is passed to either mxRun or mxTryHard
...
Further arguments will be passed to either mxRun or mxTryHard

Value

An object of class mxModel

Author(s)

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

Examples

## Not run:
## Generate data
set.seed(100)
n <- 100
x <- rnorm(n)
y <- 0.5*x + rnorm(n, mean=0, sd=sqrt(1-0.5^2))
my.df <- data.frame(y=y, x=x)

## A regression model
model <- "y ~ x # Regress y on x"
create.Tau2

Create a variance component of the heterogeneity of the random effects

Description

It creates variance component of the heterogeneity of the random effects by decomposing the variance component into matrices of correlation and standard deviations.

Usage

create.Tau2(RAM, no.var, Tau1.labels=seq(no.var),
              RE.type = c("Diag", "Symm", "Zero", "User"),
              level=c("single", "between", "within"),
              RE.User=NULL, Transform = c("expLog", "sqSD"),
              RE.startvalues=0.05)

Arguments

- **RAM**: The RAM model for testing. no.var is calculated from it.
- **no.var**: If RAM is missing, the user has to specify the no.var argument. It represents the no.var by no.var of the random effects.
- **Tau1.labels**: Parameter labels in Tau1. The default is Tau1_1, Tau1_2, etc.
create.V

RE.type
Either "Diag", "Symm", "Zero" or "User". If it is "Diag" (the default if missing), a diagonal matrix is used for the random effects meaning that the random effects are independent. If it is "Symm", a symmetric matrix is used for the random effects on the covariances among the correlation (or covariance) vectors. If it is "Zero", a zero matrix is assumed on the variance component of the random effects. If it is "User", users have to specify the RE.User argument.

level
whether it is for single-level, between-, or within-level analyses. The only difference are the names of the matrices.

RE.User
It represents the no.var by no.var symmetric matrix of TRUE or FALSE for the variance component. If the elements are FALSE, they are fixed at 0.

Transform
Either "expLog" or "sqSD". If it is "expLog", the variances are estimated by applying a log and exp transformation. If it is "sqSD", the variances are estimated by applying a square on the SD. The transformation may improve the estimation when the heterogeneity is small or close to zero.

RE.startvalues
Starting values for the variances.

Value
A list of MxMatrix-class. The variance component is computed in Tau2.

Author(s)
Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

See Also
osmasem, create.V, create.vechsR

Examples
## Not run:
T0 <- create.Tau2(no.var=4, RE.type="Diag", Transform="expLog", RE.startvalues=0.05)
T0

T1 <- create.Tau2(no.var=4, Tau1.labels=c("a", "b", "c", "d"))
T1

## End(Not run)
Usage

create.V(x, type = c("Symm", "Diag", "Full"), as.mxMatrix = TRUE)

Arguments

x
A character vector of variable names of the sampling covariance matrix.

type
Either "Symm", "Diag" or "Full". Suppose the number of variables is p, the
numbers of variable names for "Symm", "Diag", and "Full" are \( p(p - 1)/2 \), \( p \),
and \( p \times p \), respectively. The elements are arranged in a column major.

as.mxMatrix
Logical. Whether to convert the output into MxMatrix-class.

Value

A list of MxMatrix-class. The V-known sampling covariance matrix is computed in V.

Author(s)

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

See Also

osmasem, create.Tau2, create.vechsR

Examples

```r
## Not run:
my.df <- Cor2DataFrame(Nohe15A1)
## Create known sampling variance covariance matrix
V0 <- create.V(my.df$vlabels)
V0
```

---

create.vechsR  Create a model implied correlation matrix with implicit diagonal constraints

Description

It creates implicit diagonal constraints on the model implied correlation matrix by treating the error
variances as functions of other parameters.

Usage

create.vechsR(A0, S0, F0 = NULL, Ax = NULL, Sx = NULL, A.lbound=NULL, A.ubound=NULL)
Arguments

A0      A Amatrix, which will be converted into MxMatrix-class via as.mxMatrix.
S0      A Smatrix, which will be converted into MxMatrix-class via as.mxMatrix.
F0      A Fmatrix, which will be converted into MxMatrix-class via as.mxMatrix.
Ax      A Amatrix of a list of Amatrix with definition variables as the moderators of the Amatrix.
Sx      A Smatrix of a list of Smatrix with definition variables as the moderators of the Smatrix.
A.1bound A matrix of lower bound of the Amatrix. If a scalar is given, the lbound matrix will be filled with this scalar.
A.ubound A matrix of upper bound of the Amatrix. If a scalar is given, the ubound matrix will be filled with this scalar.

Value

A list of MxMatrix-class. The model implied correlation matrix is computed in impliedR and vechsR.

Note

Since A0 are the intercepts and Ax are the regression coefficients. The parameters in Ax must be a subset of those in A0.

Author(s)

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

See Also

osmasem, create.Tau2, create.V

Examples

## Not run:
## Proposed model
model1 <- 'W2 ~ w2w*W1 + s2w*S1
S2 ~ w2s*W1 + s2s*S1
W1 ~~ w1WITHs1*S1
W2 ~~ w2WITHs2*S2
W1 ~~ 1*W1
S1 ~~ 1*S1
W2 ~~ Errw2*W2
S2 ~~ Errs2*S2'

## Convert into RAM
RAM1 <- lavaan2RAM(model1, obs.variables=c("W1", "S1", "W2", "S2"))

## No moderator
M0 <- create.vechsR(A0=RAM1$A, S0=RAM1$S, F0=NULL, Ax=NULL, Sx=NULL)

## Lag (definition variable) as a moderator on the paths in the Amatrix
Ax <- matrix(c(0,0,0,0,
              0,0,0,0,
              "0*data.Lag","0*data.Lag",0,0,
              "0*data.Lag","0*data.Lag",0,0),
             nrow=4, ncol=4, byrow=TRUE)

M1 <- create.vechsR(A0=RAM1$A, S0=RAM1$S, F0=NULL, Ax=Ax, Sx=NULL)

## Lag (definition variable) as a moderator on the correlation in the Smatrix
Sx <- matrix(c(0,"0*data.Lag",0,0,
              "0*data.Lag",0,0,0,
              0,0,0,"0*data.Lag",
              0,"0*data.Lag",0),
             nrow=4, ncol=4, byrow=TRUE)

M2 <- create.vechsR(A0=RAM1$A, S0=RAM1$S, F0=NULL, Ax=NULL, Sx=Sx)

## End(Not run)

---

**Diag**

### Matrix Diagonals

**Description**

Extract or replace the diagonal of a matrix, or construct a diagonal matrix with the same behaviors as `diag` prior to R-3.0.0.

**Usage**

```r
Diag(x, ...)  
Diag(x) <- value
```

**Arguments**

- `x`: A matrix, vector or 1D array, or missing.
- `...`: Optional dimensions (nrow and ncol) for the result when `x` is not a matrix.
- `value`: Either a single value or a vector of length equal to that of the current diagonal. Should be of a mode which can be coerced to that of `x`.

**Details**

Started from R-3.0.0, `diag(x)` returns a numeric matrix with NA in the diagonals when `x` is a character vector. Although this follows what the manual says, this breaks the metaSEM. The `Diag` has the same functions as `diag` except that `Diag(x)` works for a character vector of `x` by returning a square matrix of character "0" with `x` as the diagonals.
Note

Author(s)
Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

See Also
diag

Examples
v <- c("a", "b")
Diag(v)

---

Digman97  Factor Correlation Matrices of Big Five Model from Digman (1997)

Description
The data set includes fourteen studies of the factor correlation matrices of the Five-Factor Model of personality reported by Digman (1997).

Usage
data(Digman97)

Details
A list of data with the following structure:

- **data**: A list of 14 studies of correlation matrices. The variables are *Agreeableness* (A), *Conscientiousness* (C), *Emotional Stability* (ES), *Extraversion* (E) and *Intellect* (I)
- **n**: A vector of sample sizes
- **cluster**: Types of participants of the studies

Source

References
Examples

```r
## Not run:
Digman97

##### Fixed-effects TSSEM
fixed1 <- tssem1(Digman97$data, Digman97$n, method="FEM")
summary(fixed1)

## Factor covariance among latent factors
Phi <- matrix(c(1,"0.3*cor","0.3*cor",1), ncol=2, nrow=2)

## Error covariance matrix
Psi <- Diag(c("0.2*e1","0.2*e2","0.2*e3","0.2*e4","0.2*e5"))

## S matrix
S1 <- bdiagMat(list(Psi, Phi))

## This step is not necessary but it is useful for inspecting the model.
dimnames(S1)[[1]] <- dimnames(S1)[[2]] <- c("A","C","ES","E","I","Alpha","Beta")

## Display S1
S1

## A matrix
Lambda <- matrix(c(".3*Alpha_A",".3*Alpha_C",".3*Alpha_ES",rep(0,5),".3*Beta_E",".3*Beta_I"), ncol=2, nrow=5)
A1 <- rbind(cbind(matrix(0,ncol=5,nrow=5), Lambda), matrix(0, ncol=7, nrow=2) )

## This step is not necessary but it is useful for inspecting the model.
dimnames(A1)[[1]] <- dimnames(A1)[[2]] <- c("A","C","ES","E","I","Alpha","Beta")

## Display A1
A1

## F matrix to select the observed variables
F1 <- create.Fmatrix(c(1,1,1,1,1,0,0), as.mxMatrix=FALSE)

## Display F1
F1
```

# Alternative model specification in lavaan model syntax
model <- "## Factor loadings
Alpha=~A+C+ES
Beta=~E+I
## Factor correlation
Alpha~~Beta"

## Display the model
plot(model)
RAM <- lavaan2RAM(model, obs.variables=c("A","C","ES","E","I"),
                   A.notation="on", S.notation="with")

RAM

A1 <- RAM$A
S1 <- RAM$S
F1 <- RAM$F

fixed2 <- tssem2(fixed1, Amatrix=A1, Smatrix=S1, Fmatrix=F1,
                 model.name="TSSEM2 Digman97")

summary(fixed2)

## Display the model with the parameter estimates
plot(fixed2)

#### Fixed-effects TSSEM with several clusters
#### Create a variable for different samples
#### Younger participants: Children and Adolescents
#### Older participants: others
cluster <- ifelse(Digman97$cluster %in% c("Children","Adolescents"),
                   yes="Younger participants", no="Older participants")

#### Show the cluster
cluster

#### Example of Fixed-effects TSSEM with several clusters
fixed1.cluster <- tssem1(Digman97$data, Digman97$n, method="FEM",
                          cluster=cluster)

summary(fixed1.cluster)

fixed2.cluster <- tssem2(fixed1.cluster, Amatrix=A1, Smatrix=S1, Fmatrix=F1)

#### Please note that the estimates for the younger participants are problematic.
summary(fixed2.cluster)

#### Setup two plots
layout(t(1:2))

#### Plot the first group
plot(fixed2.cluster[[1]])
title("Younger participants")

#### Plot the second group
plot(fixed2.cluster[[2]])
title("Older participants")

#### Random-effects TSSEM with random effects on the diagonals
random1 <- tssem1(Digman97$data, Digman97$n, method="REM",
                   RE.type="Diag")

summary(random1)

random2 <- tssem2(random1, Amatrix=A1, Smatrix=S1, Fmatrix=F1)
## Display the model with the parameter estimates

plot(random2, color="green")

## End(Not run)

---

**Gleser94**  
**Two Datasets from Gleser and Olkin (1994)**

**Description**

It includes two datasets in multiple-treatment studies and multiple-endpoint studies reported by Gleser and Olkin (1994).

**Usage**

```r
data("Gleser94")
```

**Format**

A list of two data frames.

- **MTS** A data frame of multiple-treatment studies.
- **MES** A data frame of multiple-endpoint studies.

**Source**


**See Also**

- smdMTS
- smdMES

**Examples**

```r
## Not run:
data(Gleser94)

#### Multiple-treatment studies
Gleser94$MTS

## Assuming homogeneity of variances
my.MTS <- t(apply(Gleser94$MTS, MARGIN=1, 
function(x) 
v=x[c("SD.C", "SD.E1", "SD.E2", "SD.E3", "SD.E4", "SD.E5")]*2, 
n=x[c("N.C", "N.E1", "N.E2", "N.E3", "N.E4", "N.E5")],
```
homogeneity="variance", list.output=FALSE)))

## Fixed-effects multivariate meta-analysis
fit.MTS <- meta(y=my.MTS[, 1:5],
   v=my.MTS[, 6:20],
   RE.constraints = diag(0, ncol=5, nrow=5),
   model.name="MTS")
summary(fit.MTS)

#### Multiple-endpoint studies
Gleser94$MES

## Calculate the sampling variances and covariance and amend into the data set
Gleser94$MES$Uncoached.V11 <- with(Gleser94$MES, SD.Uncoached.Math^2)
Gleser94$MES$Uncoached.V22 <- with(Gleser94$MES, SD.Uncoached.Verbal^2)
Gleser94$MES$Coached.V11 <- with(Gleser94$MES, SD.Coached.Math^2)
Gleser94$MES$Coached.V22 <- with(Gleser94$MES, SD.Coached.Verbal^2)

## Assuming homogeneity of covariance matrices
my.MES <- t(apply(Gleser94$MES, MARGIN=1,
   function(x)
     smdMES(m1=x["Mean.Uncoached.Math", "Mean.Uncoached.Verbal"],
     m2=x["Mean.Coached.Math", "Mean.Coached.Verbal"],
     V1=vec2symMat(x["Uncoached.V11", "Uncoached.V21", "Uncoached.V22"]),
     n1=x["N.Uncoached"],
     n2=x["N.Coached"],
     homogeneity="covariance", list.output=FALSE)))

## Fixed-effects multivariate meta-analysis
fit.MES <- meta(y=my.MES[, 1:2],
   v=my.MES[, 3:5],
   RE.constraints = diag(0, ncol=2, nrow=2),
   model.name="MES")
summary(fit.MES)

## End(Not run)

---

Gnambs18  

**Correlation Matrices from Gnambs, Scharl, and Schroeders (2018)**

**Description**

The data set includes 113 correlation matrices on the Rosenberg Self-Esteem Scale reported by Gnambs, Scharl, and Schroeders (2018). Thirty-six studies were based on the reported correlation
matrices (CorMat=1) whereas the correlation matrices of the other 77 studies were calculated from the reported factor loadings.

Usage

data(Gnambs18)

Details

A list of data with the following structure:

- **data** A list of 113 correlation matrices. The variable names are from I1 to I10.
- **n** A vector of sample sizes.
- **Year** The year of publications.
- **Country** The country of studies conducted.
- **Language** The language used in the studies.
- **Publication** Whether the studies were published (1) or unpublished (0).
- **MeanAge** Mean age of the participants.
- **FemaleProp** Proportion of the female participants.
- **Individualism** Individualism score of the country.
- **CorMat** Whether the correlation matrices are obtained from the original studies (1) or reproduced from the factor loadings (0).

Source


---

**HedgesOlkin85**  
*Effects of Open Education Reported by Hedges and Olkin (1985)*

Description

Effects of open education on attitude toward school and on reading achievement reported by Hedges and Olkin (1985).

Usage

data(HedgesOlkin85)
Details

The variables are:

study Study number

d_att Standardized mean difference on attitude

d_ach Standardized mean difference on achievement

var_att Sampling variance of the effect size of attitude

cov_att_ach Sampling covariance between the effect sizes

var_ach Sampling variance of the effect size of achievement

Source


References


Examples

data(HedgesOlkin85)

## Fixed-effects meta-analysis
summary( meta(y=cbind(d_att, d_ach),
          v=cbind(var_att, cov_att_ach, var_ach),
          data=HedgesOlkin85,
          RE.constraints=matrix(0, nrow=2, ncol=2)) )

homoStat Test the Homogeneity of Effect Sizes

Description

It tests the homogeneity of univariate and multivariate effect sizes.

Usage

homoStat(y, v)
Arguments

**y**  
A vector of effect size for univariate meta-analysis or a \( k \times p \) matrix of effect sizes for multivariate meta-analysis where \( k \) is the number of studies and \( p \) is the number of effect sizes.

**v**  
A vector of the sampling variance of the effect size for univariate meta-analysis or a \( k \times p^* \) matrix of the sampling covariance matrix of the effect sizes for multivariate meta-analysis where \( p^* = p(p + 1)/2 \). It is arranged by column major as used by `vech`. It is assumed that there is no missing value in \( v \) if \( y \) is complete. If there are missing values in \( v \) due to the missingness on \( y \), the missing values in \( v \) will be removed automatically.

Value

A list of

**Q**  
Q statistic on the null hypothesis of homogeneity of effect sizes. It has an approximate chi-square distribution under the null hypothesis.

**Q.df**  
Degrees of freedom of the Q statistic

**pval**  
p-value on the test of homogeneity of effect sizes

Author(s)

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

References


See Also

`meta`

Examples

```r
with( Hox02, homoStat(yi, vi) )

with( HedgesOlkin85, homoStat(y=cbind(d_att, d_ach),
v=cbind(var_att, cov_att_ach, var_ach)) )
```
Simulated Effect Sizes Reported by Hox (2002)

Description

Twenty stimulated studies on standardized mean difference and one continuous study characteristic reported by Hox (2002).

Usage

data(Hox02)

Details

The variables are:

- study  Study number
- yi  Effect size (standardized mean difference)
- vi  Sampling variance of the effect size
- weeks Duration of the experimental intervention in terms of weeks

Source


References


Examples

```r
## Not run:
data(Hox02)

#### ML estimation method
## Random-effects meta-analysis
summary( meta(y=yi, v=vi, data=Hox02, I2=c("I2q", "I2hm"), intervals.type="LB") )

## Fixed-effects meta-analysis
summary( meta(y=yi, v=vi, data=Hox02, RE.constraints=0, model.name="Fixed effects model") )

## Mixed-effects meta-analysis with "weeks" as a predictor
## Request likelihood-based CI
summary( meta(y=yi, v=vi, x=weeks, data=Hox02, intervals.type="LB", model.name="Mixed effects meta analysis with LB CI") )
```
### REML estimation method

#### Random-effects meta-analysis with REML

```r
summary( VarComp <- reml(y=yi, v=vi, data=Hox02) )
```

#### Extract the variance component

```r
VarComp_REML <- matrix( coef(VarComp), ncol=1, nrow=1 )
```

#### Meta-analysis by treating the variance component as fixed

```r
summary( meta(y=yi, v=vi, data=Hox02, RE.constraints=VarComp_REML) )
```

#### Mixed-effects meta-analysis with "weeks" as a predictor

#### Request Wald CI

```r
summary( reml(y=yi, v=vi, x=weeks, intervals.type="z", data=Hox02, model.name="REML with LB CI") )
```

#### End(Not run)

---

**Hunter83**  
*Fourteen Studies of Correlation Matrices reported by Hunter (1983)*

**Description**

This dataset includes fourteen studies of Correlation Matrices reported by Hunter (1983)

**Usage**

```r
data(Hunter83)
```

**Details**

A list of data with the following structure:

- **data** A list of 14 studies of correlation matrices. The variables are *Ability, Job knowledge, Work sample* and *Supervisor rating*
- **n** A vector of sample sizes

**Source**

## Examples

```r
## Not run:
data(Hunter83)

#### Fixed-effects model
#### First stage analysis
fixed1 <- tssem1(Hunter83$data, Hunter83$n, method="FEM",
model.name="TSSEM1 fixed effects model")
summary(fixed1)

#### Second stage analysis
#### Model without direct effect from Ability to Supervisor
## A1 <- create.mxMatrix(c(0,"0.1*A2J","0.1*A2W",0,0,"0.1*J2W","0.1*J2S",
## 0,0,0,"0.1*W2S",0,0,0,0),
## type="Full", ncol=4, nrow=4, as.mxMatrix=FALSE)
## dimnames(A1)[[1]] <- dimnames(A1)[[2]] <- c("Ability","Job","Work","Supervisor")
## A1

## S1 <- create.mxMatrix(c(1,"0.1*Var_e_J", "0.1*Var_e_W", "0.1*Var_e_S"),
## type="Diag", as.mxMatrix=FALSE)
## dimnames(S1)[[1]] <- dimnames(S1)[[2]] <- c("Ability","Job","Work","Supervisor")
## S1

## Regression paths
Job_knowledge ~ A2J*Ability
Work_sample ~ A2W*Ability + J2W*Job_knowledge
Supervisor ~ J2S*Job_knowledge + W2S*Work_sample

## Fix the variance of Ability at 1
Ability ~~ 1*Ability

## Label the error variances of the dependent variables
Job_knowledge ~~ VarE_J*Job_knowledge
Work_sample ~~ VarE_W*Work_sample
Supervisor ~~ VarE_S*Supervisor

## Define an indirect effect
ind := A2J*J2S+A2J*J2W*W2S+A2W*W2S

## Display the model
plot(model1, layout="spring", sizeMan=10)
RAM1 <- lavaan2RAM(model1, obs.variables=c("Ability","Job_knowledge",
"Work_sample","Supervisor"))
RAM1
```
fixed2 <- tssem2(fixed1, RAM=RAM1, intervals.type="z", 
               diag.constraints=FALSE, 
               model.name="TSSEM2 fixed effects model")
summary(fixed2)

## Display the model with the parameter estimates
plot(fixed2, layout="spring")

## Coefficients
coeff(fixed2)

## VCOV based on parametric bootstrap
vcov(fixed2)

#### Random-effects model with diagonal elements only
## First stage analysis
random1 <- tssem1(Hunter83$data, Hunter83$n, method="REM", RE.type="Diag", 
                   acov="weighted", model.name="TSSEM1 random effects model")
summary(random1)

model2 <- "## Regression paths
Job_knowledge ~ A2J*Ability
Work_sample ~ A2W*Ability + J2W*Job_knowledge
Supervisor ~ J2S*Job_knowledge + W2S*Work_sample

## Fix the variance of Ability at 1
Ability ~~ 1*Ability

## Label the error variances of the dependent variables
Job_knowledge ~~ VarE_J*Job_knowledge
Work_sample ~~ VarE_W*Work_sample
Supervisor ~~ VarE_S*Supervisor"

RAM2 <- lavaan2RAM(model2, obs.variables=c("Ability","Job_knowledge", 
                                            "Work_sample","Supervisor"))

## Second stage analysis
## Model without direct effect from Ability to Supervisor
## The "ind" effect is defined in tssem2()
random2 <- tssem2(random1, RAM=RAM2, intervals.type="LB", 
                   diag.constraints=FALSE, 
                   mx.algebras= 
                   list(ind=mxAlgebra(A2J*J2S+A2J*J2W*W2S+A2W*W2S, name="ind")), 
                   model.name="TSSEM2 random effects model")
summary(random2)

## Display the model with the parameter estimates
plot(random2, layout="spring")

## End(Not run)
**impliedR**

*Create or Generate the Model Implied Correlation or Covariance Matrices*

**Description**

It creates or generates the model implied correlation or covariance matrices based on the RAM model specification.

**Usage**

```r
impliedR(RAM, Amatrix, Smatrix, Fmatrix, corr=TRUE, labels, ...)
rimpliedR(RAM, Amatrix, Smatrix, Fmatrix, AmatrixSD, SmatrixSD, k=1, corr=TRUE, nonPD.pop=c("replace", "nearPD", "accept"))
```

**Arguments**

- **RAM**
  A RAM object including a list of matrices of the model returned from `lavaan2RAM`.

- **Amatrix**
  If `RAM` is not specified, an `Amatrix` is required. An asymmetric matrix in the RAM specification with `MxMatrix-class`. If it is a matrix, it will be converted into `MxMatrix-class` by the `as.mxMatrix` function.

- **Smatrix**
  If `RAM` is not specified, an `Smatrix` is required. A symmetric matrix in the RAM specification with `MxMatrix-class`. If it is a matrix, it will be converted into `MxMatrix-class` by the `as.mxMatrix` function.

- **Fmatrix**
  A filter matrix in the RAM specification with `MxMatrix-class`. If it is missing, an identity matrix with the same dimensions of `Smatrix` will be created, which means that all variables are observed. If it is a matrix, it will be converted into `MxMatrix-class` by the `as.mxMatrix` function. It is not required when there is no latent variable.

- **AmatrixSD**
  Standard deviations (SD) of the elements in the `Amatrix`. If it is missing, a matrix of zero is created.

- **SmatrixSD**
  Standard deviations (SD) of the elements in the `Smatrix`. If it is missing, a matrix of zero is created.

- **k**
  Number of studies.

- **corr**
  Logical. The output is either the model implied correlation matrix or the covariance matrix.

- **labels**
  A character vector of the observed and latent variables with the same dimensions as that in the `Amatrix` and `Smatrix`.

- **nonPD.pop**
  If it is `replace`, generated non-positive definite matrices are replaced by generated new ones which are positive definite. If it is `nearPD`, they are replaced by nearly positive definite matrices by calling `Matrix::nearPD()`. If it is `accept`, they are accepted.

- **...**
  Not used.
Details

This function can be used to generate the model implied correlation matrix for the standardized parameters with the `corr=TRUE` argument. Suppose we want to calculate the population correlation matrix for a mediation model with x, m, and y. We only need to specify the population path coefficients among x, m, and y in the `Amatrix`. We do not need to specify the population error variances of m and y. We treat the error variances as unknown parameters by giving them starting values in the `Smatrix` matrix. When the covariance matrix is requested by specifying `corr=FALSE`, it simply calculates the population model covariance matrix by treating the values in `Smatrix` as the population values.

Value

A list of RAM matrices, the model implied correlation or covariance matrix of the observed variables (`SigmaObs`), of both observed and latent variables (`SigmaAll`), the minimum fit (minFit) which should be zero, and the status code of the optimization (status) which should also be zero when the optimization is fine. The last object is `mx.fit` which is the output after running the model. It can be used in the diagnosis.

Note

It is important to ensure that all the population values in `Amatrix` must be set as fixed parameters; otherwise, these values may be altered with the `corr=TRUE` argument. When there is an error or warning message about the status code, there is a high chance that some of the values in `Amatrix` are incorrectly set as free parameters.

Author(s)

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

Examples

```r
set.seed(100)

## A simple mediation model
## All are population parameters in the A matrix
A1 <- matrix(c(0, 0, 0,
               0.3, 0, 0,
               0.2, 0.4, 0), nrow=3, ncol=3, byrow=TRUE,
               dimnames=list(c("x", "m", "y"), c("x", "m", "y")))
A1

## Variance of x is fixed at 1 while the other variances are free.
S1 <- matrix(c(1, 0, 0,
               0, "0.1*ErrVarM",0,
               0, 0, "0.1*ErrVarY"), nrow=3, ncol=3,
               dimnames=list(c("x", "m", "y"), c("x", "m", "y")))
S1

impliedR(Amatrix=A1, Smatrix=S1)
```
## SD of A1

A1SD <- matrix(c(0, 0, 0, 0.1, 0, 0, 0.1, 0.1, 0), nrow=3, ncol=3, byrow=TRUE, 
dimnames=list(c("x", "m", "y"), c("x", "m", "y")))

A1SD

rimpliedR(Amatrix=A1, Smatrix=S1, AmatrixSD=A1SD, k=2)

## A CFA model

A2 <- matrix(c(0, 0, 0, 0.3, 0, 0, 0, 0.4, 0, 0, 0, 0.5, 0, 0, 0, 0), nrow=4, ncol=4, byrow=TRUE, 
dimnames=list(c("x1", "x2", "x3", "f"), c("x1", "x2", "x3", "f")))

A2

## Variance of f is fixed at 1 while the other variances are free.

S2 <- matrix(c("0.7*Err1", 0, 0, 0, 0, "0.7*Err2", 0, 0, 0, 0, "0.7*Err3", 0, 0, 0, 1), nrow=4, ncol=4, 
dimnames=list(c("x1", "x2", "x3", "f"), c("x1", "x2", "x3", "f")))

S2

F2 <- create.Fmatrix(c(1,1,1,0), as.mxMatrix=FALSE)

F2

## Model implied correlation matrix

rimpliedR(Amatrix=A2, Smatrix=S2, Fmatrix=F2, corr=TRUE)

## Model implied covariance matrix

rimpliedR(Amatrix=A2, Smatrix=S2, Fmatrix=F2, corr=FALSE)

## SD of A2

A2SD <- matrix(c(0, 0, 0, 0.1, 0, 0, 0, 0.1, 0, 0, 0, 0.1, 0, 0, 0, 0), nrow=4, ncol=4, byrow=TRUE, 
dimnames=list(c("x1", "x2", "x3", "f"), c("x1", "x2", "x3", "f")))

A2SD

## SD of S2: correlated between x1 and x2

S2SD <- matrix(c(0, 0.1, 0, 0, 0.1, 0, 0, 0, 0, 0, 0, 0.1, 0, 0, 0, 0), nrow=4, ncol=4, byrow=TRUE, 
dimnames=list(c("x1", "x2", "x3", "f"), c("x1", "x2", "x3", "f")))

S2SD
indirectEffect

Estimate the asymptotic covariance matrix of standardized or unstandardized indirect and direct effects

Description

It estimates the standardized or unstandardized indirect and direct effects and their asymptotic sampling covariance matrix.

Usage

indirectEffect(x, n, standardized = TRUE, direct.effect = TRUE, run = TRUE)

Arguments

x A 3x3 correlation/covariance matrix or a list of correlation/covariance matrices. Variables are arranged as the dependent variable (y), mediator (m) and independent variable (x)
n Sample size or a vector of sample sizes
standardized Logical. Whether the indirect effect is standardized.
direct.effect Logical. Whether the direct effect is estimated. If it is FALSE, the direct effect is fixed at zero.
run Logical. If FALSE, only return the mx model without running the analysis.

Details

Cheung (2009) estimated the standardized indirect effect and its standard error with non-linear constraints. Since OpenMx does not generate standard errors when there are non-linear constraints, Kwan and Chan’s (2011) approach is used in this function. Delta method is used to calculate the asymptotic covariance matrix.

Value

A vector (or a matrix if the input is a list of matrices) of (standardized) indirect effect, standardized direct effect, and their asymptotic sampling covariance matrices

Author(s)

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>
is.pd 73

References


Examples

```r
## A correlation matrix as input
x <- matrix(c(1, 0.4, 0.2, 0.4, 1, 0.3, 0.2, 0.3, 1), ncol=3)
dimnames(x) <- list( c("y", "m", "x"), c("y", "m", "x") )
indirectEffect(x, n=300)

## A list of correlation matrices
indirectEffect( list(x, x), n=c(300,500), standardized=FALSE )
```

---

**is.pd**

*Test Positive Definiteness of a List of Square Matrices*

**Description**

It tests the positive definiteness of a square matrix or a list of square matrices. It returns `TRUE` if the matrix is positive definite. It returns `FALSE` if the matrix is either non-positive definite or not symmetric. Variables with NA in the diagonals will be removed before testing. It returns NA when there are missing correlations even after deleting the missing variables.

**Usage**

```r
is.pd(x, check.aCov=FALSE, cor.analysis=TRUE, tol=1e-06)
```

**Arguments**

- `x` A square matrix or a list of square matrices
- `check.aCov` If it is `TRUE`, it mirrors the checking in `asyCov`.
- `cor.analysis` Whether the input matrix is a correlation or a covariance matrix. It is ignored when `check.aCov=FALSE`.
- `tol` Tolerance (relative to largest variance) for numerical lack of positive-definiteness in `x`. It is adopted from `mvnorm`.

**Value**

If the input is a matrix, it returns `TRUE`, `FALSE` or `NA`. If the input is a list of matrices, it returns a list of `TRUE`, `FALSE` or `NA`.

**Author(s)**

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>
Examples

A <- diag(1,3)
is.pd(A)
# TRUE

B <- matrix(c(1,2,2,1), ncol=2)
is.pd(B)
# FALSE

is.pd(list(A, B))
# TRUE FALSE

C <- A
C[2,1] <- C[1,2] <- NA
is.pd(C)
# NA

issp05 A Dataset from ISSP (2005)

Description

Thirty-two covariance matrices on work-related attitudes were extracted from the International Social Survey Programme 2005: Work Orientation III (ISSP, 2005). Seven variables were selected for demonstration purposes. They were grouped into three constructs: Importance of Job Prospects measured by job security (JP1), high income (JP2), and opportunity for advancement (JP3); Importance of Job Autonomy measured by work independently (JA1) and decide time of work (JA2); and Importance of Contributions to Society measured by help other people (CS1) and a job useful to society (CS2).

Usage

data(issp05)

Details

A list of data with the following structure:

data  A list of 32 covariance matrices
n  A vector of sample sizes

Author(s)

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

Source

### Not run:
```r
data(issp05)

#### Fixed-effects TSSEM
fixed1 <- tssem1(issp05$data, issp05$n, method="FEM")
summary(fixed1)
```

#### Prepare a model implied matrix
#### Factor correlation matrix
```r
Phi <- create.mxMatrix( c("0.3*corf2f1","0.3*corf3f1","0.3*corf3f2"),
                         type="Stand", as.mxMatrix=FALSE )
```
#### Error variances
```r
Psi <- create.mxMatrix( paste("0.2*e", 1:7, sep=""), type="Diag",
                        as.mxMatrix=FALSE )
```
#### Create Smatrix
```r
S1 <- bdiagMat(list(Psi, Phi))
```
#### Create Amatrix
```r
A1 <- rbind( cbind(Zero1, Lambda),
              Zero2 )
```
#### Create Fmatrix
```r
F1 <- create.Fmatrix(c(rep(1,7), rep(0,3)))
```

### Fixed-effects model: Stage 2 analysis
```r
fixed2 <- tssem2(fixed1, Amatrix=A1, Smatrix=S1, Fmatrix=F1,
                 intervals.type="LB")
summary(fixed2)
```
A Dataset from Cheung and Chan (2005; 2009)

Description

Eleven covariance matrices on work-related attitudes were extracted from the Inter-University Consortium for Political and Social Research (1989). Nine variables were selected by Cheung and Chan (2005; 2009) for demonstration purposes. They were grouped into three constructs: *Job Prospects* measured by job security (JP1), income (JP2), and advancement opportunity (JP3); *Job Nature* measured by interesting job (JN1), independent work (JN2), help other people (JN3), and useful to society (JN4); and *Time Demand* measured by flexible working hours (TD1) and lots of leisure time (TD2).

Usage

data(issp89)

Details

A list of data with the following structure:

- **data**: A list of 11 studies of covariance matrices
- **n**: A vector of sample sizes

Author(s)

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

Source


References


See Also

issp05
Examples

```r
## Not run:
data(issp89)

#### Analysis of correlation structure in Cheung and Chan (2005)
#### Fixed-effects model: Stage 1 analysis
cor1 <- tssem1(issp89$data, issp89$n, method="FEM", cor.analysis=TRUE)
summary(cor1)

#### Prepare a model implied matrix
#### Factor correlation matrix
Phi <- create.mxMatrix( c("0.3*corf2f1","0.3*corf3f1","0.3*corf3f2"),
  type="Stand", as.mxMatrix=FALSE )
#### Error variances
Psi <- create.mxMatrix( paste("0.2*e", 1:9, sep=""), type="Diag",
  as.mxMatrix=FALSE )

#### Create Smatrix
S1 <- bdiagMat(list(Psi, Phi))
## dimnames(S1)[[1]] <- dimnames(S1)[[2]] <- c(paste("x",1:9,sep=""),
##   paste("f",1:3,sep=""))
S1 <- as.mxMatrix(S1)

#### Factor loadings
Lambda <- create.mxMatrix( c(".3*f1x1",".3*f1x2",".3*f1x3",rep(0,9),
  ".3*f2x4",".3*f2x5",".3*f2x6",".3*f2x7",rep(0,9),".3*f3x8",".3*f3x9"),
  type="Full", ncol=3, nrow=9, as.mxMatrix=FALSE )
Zero1 <- matrix(0, nrow=9, ncol=9)
Zero2 <- matrix(0, nrow=3, ncol=12)

#### Create Amatrix
A1 <- rbind( cbind(Zero1, Lambda),
  Zero2 )
## dimnames(A1)[[1]] <- dimnames(A1)[[2]] <- c(paste("x",1:9,sep=""),
##   paste("f",1:3,sep=""))
A1 <- as.mxMatrix(A1)

#### Create Fmatrix
F1 <- create.Fmatrix(c(rep(1,9), rep(0,3)))

#### Fixed-effects model: Stage 2 analysis
cor2 <- tssem2(cor1, Amatrix=A1, Smatrix=S1, Fmatrix=F1, intervals.type="LB")
summary(cor2)

#### Display the model with the parameter estimates
plot(cor2, nDigits=1)

#### Analysis of covariance structure in Cheung and Chan (2009)
#### Fixed-effects model: Stage 1 analysis
```
cov1 <- tssem1(issp89$data, issp89$n, method="FEM", cor.analysis=FALSE)
summary(cov1)

### Fixed-effects model: Stage 2 analysis
cov2 <- tssem2(cov1, Amatrix=A1, Smatrix=S1, Fmatrix=F1)
summary(cov2)

## Display the model with the parameter estimates
plot(cov2, nDigits=1)

## End(Not run)

Jaramillo05  

---

**Dataset from Jaramillo, Mulki & Marshall (2005)**

### Description

A dataset of the relationship between organizational commitment (OC) and salesperson job performance (JP) from Jaramillo, Mulki & Marshall (2005).

### Usage

data(Jaramillo05)

### Format

A data frame with 61 observations on the following 10 variables.

- **Author**: a character vector of study
- **Sample_size**: sample size of the study
- **Sales**: sample type; either "mixed", "nonsales" or "sales"
- **Country**: a character vector of country of study
- **IDV**: Hofstede’s (1997) individualism index
- **OC_scale**: scale of OC; either "Porter or Mowday", "Meyer" or "other"
- **OC_alpha**: Coefficient alpha of organizational commitment
- **JP_alpha**: Coefficient alpha of job performance
- **r**: correlation between organizational commitment and job performance
- **r_v**: sampling variance of r

### Source

## Examples

```r
## Not run:
## Research question 4.4.1
summary(meta(r, r_v, data=Jaramillo05))

## Research question 4.4.2
## Select cases with either "sales" or "nonsales"
Sales.df <- subset(Jaramillo05, Sales %in% c("sales", "nonsales"))

## Create a predictor with 1 and 0 when they are "sales" or "nonsales", respectively
predictor <- ifelse(Jaramillo05$Sales=="sales", yes=1, no=0)

## Mixed-effects meta-analysis
summary( meta(y = r, v = r_v, x = predictor, data = Jaramillo05) )

## Research question 4.4.3
summary(meta(r, r_v, x=IDV, data=Jaramillo05))

## End(Not run)
```

---

**Kalaian96**

*Multivariate effect sizes reported by Kalaian and Raudenbush (1996)*

**Description**

This data set includes 47 multivariate effect sizes reported by Kalaian and Raudenbush (1996, Table 1).

**Usage**

```r
data(Kalaian96)
```

**Details**

A list of data with the following structure:

- **Study** Study name
- **Year** Year of publication
- **n_e** Sample size of the experimental group
- **n_c** Sample size of the control group
- **dSAT_V** Standardized mean difference of the Scholastic Aptitude Test (SAT) on verbal
- **dSAT_M** Standardized mean difference of SAT on math
- **var_V** Sampling variance of dSAT_V
- **cov_VM** Sampling covariance of dSAT_V and dSAT_M with a common correlation of 0.66
- **var_M** Sampling variance of dSAT_M
- **Hr** Hours of training
ETS Educational Testing Service

Study_type Either Randomized, Matched or Nonequivalent comparison

Home_work Home work

Source


Examples

## Not run:
data(Kalaian96)
## End(Not run)

lavaan2RAM Convert lavaan models to RAM models

Description

It converts models specified in lavaan model syntax to RAM models.

Usage

lavaan2RAM(model, obs.variables = NULL, A.notation = "ON", 
        S.notation = "WITH", M.notation = "mean", 
        A.start=0.1, S.start=0.5, M.start=0, 
        auto.var = TRUE, std.lv = TRUE, ngroups = 1, ...)

Arguments

- **model**: A character string of model using the lavaan model syntax.
- **obs.variables**: A character vector of the observed variables. The observed variables in the RAM specification will follow the order specified in obs.variables. It is important to check whether the order of the observed variables matches the order in the dataset.
- **A.notation**: A character string to be used in the A matrix if the labels are not included in the lavaan model. For example, the label will be "yONx" for regressing "y" on "x".
- **S.notation**: A character string to be used in the S matrix if the labels are not included in the lavaan model. For example, the label will be "yWITHx" for the covariance between "y" with "x" and "yWITHy" for the (error) variance of "y".
- **M.notation**: A character string to be used in the M matrix if the labels are not included in the lavaan model. For example, the label will be "ymean" for the mean of "y" if M.notation="mean".
A.start  A numeric value of starting value for the A matrix when the starting values are not provided.
S.start  A numeric value of starting value for the S matrix when the starting values are not provided.
M.start  A numeric value of starting value for the M matrix when the starting values are not provided.
auto.var Logical. If TRUE, the residual variances and the variances of exogenous latent variables are included in the model and set free. See `model.syntax`.
std.lv Logical. If TRUE, the metric of each latent variable is determined by fixing their variances to 1.0. If FALSE, the metric of each latent variable is determined by fixing the factor loading of the first indicator to 1.0. See `model.syntax`.
ngroups Number. The number of groups in the model. See `model.syntax`.
... Further arguments to be passed to `model.syntax`.

Please note that `fixed.x` is set at FALSE. Thus it cannot be passed to `...`.

Details

It uses the `model.syntax` to do the conversion. Experimental: functions of parameters (`:`=` in lavaan) and constraints (`==`, `>`, and `<` in lavaan) will be converted to mxAlgebra and mxConstraint in OpenMx. As there are differences between lavaan and OpenMx, they may not work properly.

Value

A list of RAM specification with A, S, F, and M matrices.

Author(s)

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

See Also

`ramModel`, `Becker92`, `Becker09`, `Digman97`, `Hunter83`, `as.mxMatrix`, `checkRAM`

Examples

```r
## Regression model on correlation matrix
modell <- "## y is modelled by x1, x2, and x3
y ~ b1*x1 + b2*x2 + b3*x3
## Fix the independent variables at 1
x1 ~~ 1*x1
x2 ~~ 1*x2
x3 ~~ 1*x3
## Declare the correlations among the independent variables
x1 ~~ x2
x1 ~~ x3
x2 ~~ x3
## Constraint
```
### list2matrix

Convert a List of Symmetric Matrices into a Stacked Matrix

#### Description

It converts a list of symmetric matrices into a stacked matrix. Dimensions of the symmetric matrices have to be the same. It tries to preserve the dimension names if possible. Dimension names will be created if there are no dimension names in the first symmetric matrix.

#### Usage

```r
list2matrix(x, diag = FALSE)
```

#### Arguments

- **x**: A list of $k \times p \times p$ symmetric matrices.
- **diag**: Logical. If it is `TRUE`, `vech` is used to vectorize the (covariance) matrices. If it is `FALSE`, `vechs` is used to vectorize the (correlation) matrices.
Value

A \( k \times p^* \) stacked matrix where \( p^* = \frac{p(p-1)}{2} \) for \( \text{diag} = \text{FALSE} \) or \( p^* = \frac{p(p+1)}{2} \) for \( \text{diag} = \text{TRUE} \).

Author(s)

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

Examples

```r
C1 <- matrix(c(1,0.5,0.4,0.5,1,0.2,0.4,0.2,1), ncol=3)
C2 <- matrix(c(1,0.4,NA,0.4,1,NA,NA,NA,NA), ncol=3)

## A list without dimension names
list2matrix(list(C1, C2))

## A list with dimension names
list2matrix(list(C1, C2))
```

Description

Eight studies from Mak et al. (2009) were reported by Cheung et al. (2012).

Usage

`data(Mak09)`

Format

A data frame with 8 observations on the following 10 variables.

- **Study**: a character vector of study
- **type**: a character vector
- **AF.BP**: a numeric vector
- **Tot.BP**: a numeric vector
- **AF.non.BP**: a numeric vector
Mathieu15

TOT.non.BP a numeric vector
yi a numeric vector
vi a numeric vector
age.mean a numeric vector
study.duration a numeric vector

Source


References


Examples

```r
## Random-effects meta-analysis
( meta1 <- summary(meta(y=yi, v=vi, data=Mak09, I2=c("I2q", "I2hm"))) )

## Convert the estimates back into odds ratio
OR <- with(coef(meta1), exp(c(Estimate[1], lbound[1], ubound[1])))
names(OR) <- c("Estimate in OR", "lbound in OR", "ubound in OR")

## Mixed-effects meta-analysis with mean age as a predictor
summary( meta(y=yi, v=vi, x=age.mean, data=Mak09) )
```

Mathieu15

*Correlation Matrices from Mathieu et al. (2015)*

Description

The data set includes a list of correlation matrices of panel studies between cohesion (C) and performance (P) in Mathieu et al. (2015, Table 1).

Usage

data(Mathieu15)
Details

A list of data with the following structure:

- **data** A list of studies of correlation matrices. The variables are $C_1$, $P_1$, $C_2$, and $P_2$.
- **n** A vector of sample sizes.
- **Year** Year of publication.
- **Sample** Sample characteristics.
- **Student** Whether the samples are student or non-student based on Sample.

Source


Examples

```r
## Not run:
# TSSEM
## Model 1: no constraint
## Stage 1 analysis
tsem1.fit <- tsem1(Mathieu15$data, Mathieu15$n)
summary(tsem1.fit)

## Proposed model in lavaan syntax
model1 <- 'C2 ~ c2c*C1 + p2c*P1
P2 ~ c2p*C1 + p2p*P1
C1 ~~ c1withp1*P1
C1 ~~ 1*C1
P1 ~~ 1*P1
C2 ~~ c2withp2*P2'

## Convert the lavaan model to RAM specification
RAM1 <- lavaan2RAM(model1, obs.variables=c("C1", "P1", "C2", "P2"))
RAM1

## Stage 2 analysis
tsem1b.fit <- tsem2(tsem1.fit, RAM=RAM1)
summary(tsem1b.fit)

plot(tsem1b.fit, col="yellow", edge.label.position=0.58)

## Model 2: Equality constraints on the path coefficient
## Proposed model with equal effects time 1 to time 2
model2 <- 'C2 ~ same*C1 + diff*P1
P2 ~ diff*C1 + same*P1
C1 ~~ c1withp1*P1
C1 ~~ 1*C1
P1 ~~ 1*P1
C2 ~~ c2withp2*P2'
```
## Convert the lavaan model to RAM specification

```r
RAM2 <- lavaan2RAM(model2, obs.variables=c("C1", "P1", "C2", "P2"))
```

## Stage 2 analysis

```r
tssem2b.fit <- tssem2(tssem1.fit, RAM=RAM2)
summary(tssem2b.fit)
```

## Compare the models with and without the constraints.

```r
anova(tssem1b.fit, tssem2b.fit)
```

## Plot the model

```r
plot(tssem2b.fit, col="yellow", edge.label.position=0.60)
```

## OSMASEM

```r
my.df <- Cor2DataFrame(Mathieu15)
head(my.df$data)
```

## Model without any moderator

```r
osmasem.fit1 <- osmasem(model.name="No moderator", RAM=RAM1, data=my.df)
summary(osmasem.fit1)
```

## Extract the heterogeneity variance-covariance matrix

```r
diag(VarCorr(osmasem.fit1))
```

## Model with student sample as a moderator on the regression coefficients

```r
A1 <- create.modMatrix(RAM1, output="A", "Student")
A1
```

## Model with a moderator

```r
osmasem.fit2 <- osmasem(model.name="Student sample as a moderator", RAM=RAM1,
                         Ax=A1, data=my.df)
summary(osmasem.fit2)
```

## Compare the models with and without the moderator

```r
anova(osmasem.fit2, osmasem.fit1)
```

## Get the R2 of the moderator

```r
osmasemR2(osmasem.fit2, osmasem.fit1)
```

## End(Not run)
**matrix2bdiag**

Description

It converts a matrix into a block diagonal matrix.

Usage

\[
\text{matrix2bdiag}(x, \ldots)
\]

Arguments

- **x**: A \( k \times p \) matrix of numerics or characters.
- **\ldots**: Further arguments to be passed to **vec2symMat**

Details

Each row of \( x \) is converted into a symmetric matrix via **vec2symMat**. Then the list of the symmetric matrices is converted into a block diagonal matrix via a function written by Scott Chasalow posted at http://www.math.yorku.ca/Who/Faculty/Monette/pub/stmp/0827.html.

Author(s)

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

See Also

- **vec2symMat**

Examples

```r
(m1 <- matrix(1:12, ncol=6, byrow=TRUE))
# [1,]  1  2  3  4  5  6
# [2,]  7  8  9 10 11 12

matrix2bdiag(m1)
# [1,]  1  2  3  0  0  0
# [2,]  2  4  5  0  0  0
# [3,]  3  5  6  0  0  0
# [4,]  0  0  0  7  8  9
# [5,]  0  0  0  8 10 11
# [6,]  0  0  0  9 11 12
```
Univariate and Multivariate Meta-Analysis with Maximum Likelihood Estimation

Description

It conducts univariate and multivariate meta-analysis with maximum likelihood estimation method. Mixed-effects meta-analysis can be conducted by including study characteristics as predictors. Equality constraints on intercepts, regression coefficients, and variance components can be easily imposed by setting the same labels on the parameter estimates.

Usage

```r
meta(y, v, x, data, intercept.constraints = NULL, coef.constraints = NULL, RE.constraints = NULL, RE.startvalues=0.1, RE.lbound = 1e-10, intervals.type = c("z", "LB"), I2="I2q", R2=TRUE, model.name="Meta analysis with ML", suppressWarnings = TRUE, silent = TRUE, run = TRUE, ...)
metaFIML(y, v, x, av, data, intercept.constraints=NULL, coef.constraints=NULL, RE.constraints=NULL, RE.startvalues=0.1, RE.lbound=1e-10, intervals.type=c("z", "LB"), R2=TRUE, model.name="Meta analysis with FIML", suppressWarnings=TRUE, silent=TRUE, run=TRUE, ...)
```

Arguments

- **y**
  A vector of effect size for univariate meta-analysis or a $k \times p$ matrix of effect sizes for multivariate meta-analysis where $k$ is the number of studies and $p$ is the number of effect sizes.

- **v**
  A vector of the sampling variance of the effect size for univariate meta-analysis or a $k \times p^*$ matrix of the sampling covariance matrix of the effect sizes for multivariate meta-analysis where $p^* = p(p + 1)/2$. It is arranged by column major as used by `vech`.

- **x**
  A predictor or a $k \times m$ matrix of predictors where $m$ is the number of predictors.

- **av**
  An auxiliary variable or a $k \times m$ matrix of auxiliary variables where $m$ is the number of auxiliary variables.

- **data**
  An optional data frame containing the variables in the model.

- **intercept.constraints**
  A $1 \times p$ matrix specifying whether the intercepts of the effect sizes are fixed or free. If the input is not a matrix, the input is converted into a $1 \times p$ matrix with `t(as.matrix(intercept.constraints))`. The default is that the intercepts are free. When there is no predictor, these intercepts are the same as the pooled effect sizes. The format of this matrix follows `as.mxFMatrix`. The intercepts can be constrained equally by using the same labels.
coef.constraints
A $p \times m$ matrix specifying how the predictors predict the effect sizes. If the input is not a matrix, it is converted into a matrix by `as.matrix()`. The default is that all $m$ predictors predict all $p$ effect sizes. The format of this matrix follows `as.mxMatrix`. The regression coefficients can be constrained equally by using the same labels.

RE.constraints
A $p \times p$ matrix specifying the variance components of the random effects. If the input is not a matrix, it is converted into a matrix by `as.matrix()`. The default is that all covariance/variance components are free. The format of this matrix follows `as.mxMatrix`. Elements of the variance components can be constrained equally by using the same labels. If a zero matrix is specified, it becomes a fixed-effects meta-analysis.

RE.startvalues
A vector of $p$ starting values on the diagonals of the variance component of the random effects. If only one scalar is given, it will be duplicated across the diagonals. Starting values for the off-diagonals of the variance component are all 0. A $p \times p$ symmetric matrix of starting values is also accepted.

RE.lbound
A vector of $p$ lower bounds on the diagonals of the variance component of the random effects. If only one scalar is given, it will be duplicated across the diagonals. Lower bounds for the off-diagonals of the variance component are set at `NA`. A $p \times p$ symmetric matrix of the lower bounds is also accepted.

intervals.type
Either `z` (default if missing) or `LB`. If it is `z`, it calculates the 95% Wald confidence intervals (CIs) based on the z statistic. If it is `LB`, it calculates the 95% likelihood-based CIs on the parameter estimates. Note that the z values and their associated p values are based on the z statistic. They are not related to the likelihood-based CIs.

I2
Possible options are "I2q", "I2hm" and "I2am". They represent the I2 calculated by using a typical within-study sampling variance from the Q statistic, the harmonic mean and the arithmetic mean of the within-study sampling variances (Xiong, Miller, & Morris, 2010). More than one options are possible. If `intervals.type="LB"`, 95% confidence intervals on the heterogeneity indices will be constructed.

R2
Logical. If TRUE and there are predictors, R2 is calculated (Raudenbush, 2009).

model.name
A string for the model name in `mxModel`.

suppressWarnings
Logical. If TRUE, warnings are suppressed. The argument to be passed to `mxRun`.

silent
Logical. An argument to be passed to `mxRun`.

run
Logical. If FALSE, only return the mx model without running the analysis.

...
Further arguments to be passed to `mxRun`.

Value
An object of class `meta` with a list of

call
Object returned by `match.call`
data
A data matrix of y, v and x
no.y   No. of effect sizes
no.x   No. of predictors
miss.x A vector indicating whether the predictors are missing. Studies will be removed before the analysis if they are TRUE
I2     Types of I2 calculated
R2     Logical
mx.fit A fitted object returned from mxRun
mx@0.fit A fitted object without any predictor returned from mxRun

Note
Missing values (NA) in y and their related elements in v will be removed automatically. When there are missing values in v but not in y, missing values will be replaced by 1e5. Effectively, these effect sizes will have little impact on the analysis. metaFIML() uses FIML to handle missing covariates in X. It is experimental. It may not be stable.

Author(s)
Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

References


See Also
reml, Hox02, Berkey98, wvs94a
**Description**

It converts objects in class `wls` into objects of class `semPlotModel`.

**Usage**

```r
meta2semPlot(object, manNames = NULL, latNames = NULL, labels = c("labels", "RAM"), ...)
```

**Arguments**

- `object` An object of class `wls` returned from `wls()` or `tssem2()`.
- `manNames` A character vector of the manifest names. The program will try to get it from the object if it is not given.
- `latNames` A character vector of the latent names. The program will create it by using "L1", "L2", etc if it is not given.
- `labels` Either `labels` (default if missing) or `RAM`. If `labels`, the labels of the parameters are used in plotting. If `RAM`, the RAM notations are used in plotting.
- `...` Further arguments to be passed to `ramModel`.

**Details**

It uses the `ramModel()` to do the conversion.

**Value**

A "semPlotModel" object.

**Author(s)**

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

**See Also**

`ramModel`, `Becker92`, `Becker09`, `Digman97`, `Hunter83`
Three-Level Univariate Meta-Analysis with Maximum Likelihood Estimation

Description

It conducts three-level univariate meta-analysis with maximum likelihood estimation method. Mixed-effects meta-analysis can be conducted by including study characteristics as predictors. Equality constraints on the intercepts, regression coefficients and variance components on the level-2 and on the level-3 can be easily imposed by setting the same labels on the parameter estimates.

Usage

```r
## Depreciated in the future
meta3L(y, v, cluster, x, data, intercept.constraints = NULL,
       coef.constraints = NULL, RE2.constraints = NULL,
       RE2.lbound = 1e-10, RE3.constraints = NULL, RE3.lbound = 1e-10,
       intervals.type = c("z", "LB"), I2="I2q",
       R2=TRUE, model.name = "Meta analysis with ML",
       suppressWarnings = TRUE, silent = TRUE, run = TRUE, ...)
```

Arguments

- `y` A vector of `k` studies of effect size.
- `v` A vector of `k` studies of sampling variance.
- `cluster` A vector of `k` string or number indicating the clusters.
- `x` A predictor or a `k x m` matrix of level-2 and level-3 predictors where `m` is the number of predictors.
x2  A predictor or a $k \times m$ matrix of level-2 predictors where $m$ is the number of predictors.

x3  A predictor or a $k \times m$ matrix of level-3 predictors where $m$ is the number of predictors.

av2  A predictor or a $k \times m$ matrix of level-2 auxiliary variables where $m$ is the number of variables.

av3  A predictor or a $k \times m$ matrix of level-3 auxiliary variables where $m$ is the number of variables.

data  An optional data frame containing the variables in the model.

intercept.constraints  A $1 \times 1$ matrix specifying whether the intercept of the effect size is fixed or constrained. The format of this matrix follows as.mxMatrix. The intercept can be constrained with other parameters by using the same label.

coef.constraints  A $1 \times m$ matrix specifying how the level-2 and level-3 predictors predict the effect sizes. If the input is not a matrix, it is converted into a matrix by as.matrix(). The default is that all $m$ predictors predict the effect size. The format of this matrix follows as.mxMatrix. The regression coefficients can be constrained equally by using the same labels.

RE2.constraints  A scalar or a $1 \times 1$ matrix specifying the variance components of the random effects. The default is that the variance components are free. The format of this matrix follows as.mxMatrix. Elements of the variance components can be constrained equally by using the same label.

RE2.lb  A scalar or a $1 \times 1$ matrix of lower bound on the level-2 variance component of the random effects.

RE3.constraints  A scalar or a $1 \times 1$ matrix specifying the variance components of the random effects at level-3. The default is that the variance components are free. The format of this matrix follows as.mxMatrix. Elements of the variance components can be constrained equally by using the same label.

RE3.lb  A scalar or a $1 \times 1$ matrix of lower bound on the level-3 variance component of the random effects.

intervals.type  Either z (default if missing) or LB. If it is z, it calculates the 95% Wald confidence intervals (CIs) based on the z statistic. If it is LB, it calculates the 95% likelihood-based CIs on the parameter estimates. Note that the z values and their associated p values are based on the z statistic. They are not related to the likelihood-based CIs.

I2  Possible options are "I2q", "I2hm", "I2am" and "ICC". They represent the I2 calculated by using a typical within-study sampling variance from the Q statistic, the harmonic mean, the arithmetic mean of the within-study sampling variances, and the intra-class correlation. More than one options are possible. If intervals.type= "LB", 95% confidence intervals on the heterogeneity indices will be constructed.

R2  Logical. If TRUE and there are predictors, R2 is calculated.
model.name A string for the model name in `mxModel`.
suppressWarnings Logical. If TRUE, warnings are suppressed. It is passed to `mxRun`.
silent Logical. An argument to be passed to `mxRun`.
run Logical. If FALSE, only return the mx model without running the analysis.

... Further arguments to be passed to `mxRun`.

Details

\[ y_{ij} = \beta_0 + \beta' \cdot x_{ij} + u_{(2)ij} + u_{(3)j} + e_{ij} \]

where \( y_{ij} \) is the effect size for the \( i \)th study in the \( j \)th cluster, \( \beta_0 \) is the intercept, \( \beta \) is the regression coefficients, \( x_{ij} \) is a vector of predictors, \( u_{(2)ij} \sim N(0, \tau^2_2) \) and \( u_{(3)j} \sim N(0, \tau^2_3) \) are the level-2 and level-3 heterogeneity variances, respectively, and \( e_{ij} \sim N(0, v_{ij}) \) is the conditional known sampling variance.

`meta3L()` does not differentiate between level-2 or level-3 variables in \( x \) since both variables are treated as a design matrix. When there are missing values in \( x \), the data will be deleted. `meta3LFIML()` treats the predictors \( x_2 \) and \( x_3 \) as level-2 and level-3 variables. Thus, their means and covariance matrix will be estimated. Missing values in \( x_2 \) and \( x_3 \) will be handled by (full information) maximum likelihood (FIML) in `meta3LFIML()`. Moreover, auxiliary variables \( av_2 \) at level-2 and \( av_3 \) at level-3 may be included to improve the estimation. Although `meta3LFIML()` is more flexible in handling missing covariates, it is more likely to encounter estimation problems.

Author(s)

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

References


See Also

`reml3L`, `Cooper03`, `Bornmann07`
Description

This dataset includes 59 studies reported by Nam, Mengersen, and Garthwaite (2003) on the potential health effects among children exposed to environmental tobacco smoke (ETS), or passive smoking. The effect sizes are the log odds ratios of asthma and lower respiratory disease (LRD).

Usage

data(Nam03)

Details

A list of data with the following structure:

- **ID**: Study identification number.
- **Size**: Total number of valid subjects in the study.
- **Age**: Mean age of participants.
- **Year**: Year of publication.
- **Country**: Country code.
- **Smoke**: Source of ETS.
- **Adj**: Whether the reported odds ratio is adjusted for covariates.
- **Asthma_logOR**: Log odds ratio of asthma.
- **LRD_logOR**: Log odds ratio of lower respiratory disease.
- **Asthma_v**: Sampling variance of Asthma_logOR.
- **LRD_v**: Sampling variance of LRD_logOR.
- **AsthmaLRD_cov_05**: Sampling covariance between Asthma_logOR and LRD_logOR by assuming a correlation of 0.5

Source


Examples

data(Nam03)
Correlation Matrices from Nohe et al. (2015)

Description
The data sets include two lists of correlation matrices of panel studies between work-family conflict and strain reported in Table A1 (Nohe15A1) and Table A2 (Nohe15A2) by Nohe et al. (2015).

Usage
```r
data(Nohe15A1)
data(Nohe15A2)
```

Details
A list of data with the following structure:

- **data** A list of studies of correlation matrices. The variables are W1, S1, W2, and S2 in Nohe15A1 and F1, S1, F2, and S2 in Nohe15A2
- **n** A vector of sample sizes
- **RelXX** The reliabilities of W1, S1, W2 and S2 in Nohe15A1 and the reliabilities of F1, S1, F2, and S2 in Nohe15A2
- **FemalePer** Percentage of female participants
- **Publication** Whether the studies were published (P) or unpublished (U)
- **Lag** Time lag between the coded measurement waves in months

Source

Examples
```r
## Not run:
#### TSSEM
## Set seed for replicability
set.seed(23891)

## Table A1
randA1a <- tssem1(Nohe15A1$data, Nohe15A1$n, method="REM", RE.type="Diag")
summary(randA1a)

model1 <- 'W2 ~ w2*w1 + s2*w1
S2 ~ w2*s1 + s2*s1
W1 ~~ w1WITH*s1'
```
## Display the model
plot(model1, layout="spring")

RAM1 <- lavaan2RAM(model1, obs.variables=c("W1", "S1", "W2", "S2"))
RAM1

randA1b <- tssem2(randA1a, Amatrix=RAM1$A, Smatrix=RAM1$S)
summary(randA1b)

## Display the model with the parameter estimates
plot(randA1b, layout="spring")

## Table A2
randA2a <- tssem1(Nohe15A2$data, Nohe15A2$n, method="REM", RE.type="Diag")
## Rerun to remove error code
randA2a <- rerun(randA2a)
summary(randA2a)

model2 <- 'F2 ~ f2f*F1 + s2F*S1
S2 ~ f2s*F1 + s2s*S1
F1 ~ 1*F1
S1 ~ 1*S1
F2 ~ Errf2*F2
S2 ~ Errs2*S2'

## Display the model
plot(model2, layout="spring")

RAM2 <- lavaan2RAM(model2, obs.variables=c("F1", "S1", "F2", "S2"))
RAM2

randA2b <- tssem2(randA2a, Amatrix=RAM2$A, Smatrix=RAM2$S)
summary(randA2b)

## Display the model with the parameter estimates
plot(randA2b, layout="spring")

## Estimate the heterogeneity of the parameter estimates
tsssemParaVar(randA1a, randA2b)

## Parametric bootstrap based on Yu et al. (2016)
## I assume that you know what you are doing!

## Set seed for reproducibility
set.seed(39128482)
## Average the correlation coefficients with the univariate-r approach
```r
uni1 <- uniR1(Nohe15A1$data, Nohe15A1$n)
```

## Generate random correlation matrices
```r
boot.cor <- bootuniR1(uni1, Rep=50)
```

## Display the quality of the generated correlation matrices
```r
summary(boot.cor)
```

## Proposed saturated model
```r
model1 <- 'W2 + S2 ~ W1 + S1'
```

## Use the harmonic mean of the sample sizes as n in SEM
```r
n <- uni1$n.harmonic
boot.fit1 <- bootuniR2(model=model1, data=boot.cor, n=n)
summary(boot.fit1)
```

## Proposed model with equal regression coefficients
```r
model2 <- 'W2 ~ Same*W1 + Cross*S1
S2 ~ Cross*W1 + Same*S1'
```

## Convert it into RAM specification
```r
RAM1 <- lavaan2RAM(model1, obs.variables=c("W1", "S1", "W2", "S2"))
```

#### OSMASEM

## Calculate the sampling variance-covariance matrix of the correlation matrices.
```r
my.df <- Cor2DataFrame(Nohe15A1)
```

## Standardize the moderator “Lag”
```r
my.df$data$Lag <- scale(my.df$data$Lag)
```

## Proposed model
```r
model1 <- 'W2 ~ w2w*W1 + s2w*S1
S2 ~ w2s*W1 + s2s*S1
W1 ~ w1WITHs1*S1
W2 ~ w2WITHs2*S2
W1 ~ 1*W1
S1 ~ 1*S1
W2 ~ Errw2*W2
S2 ~ Errs2*S2'
```

## Convert it into RAM specification
```r
RAM1 <- lavaan2RAM(model1, obs.variables=c("W1", "S1", "W2", "S2"))
```
## with implicit diagonal constraints
## Create heterogeneity variances
## RE.type= either "Diag" or "Symm"
## Transform= either "expLog" or "sqSD" for better estimation on variances
## T0 <- create.Tau2(RAM=RAM1, RE.type="Diag")
## Fit the model
## fit0 <- osmasem(model.name="No moderator", Mmatrix=M0, Tmatrix=T0, data=my.df)

## Fit the model
fit0 <- osmasem(model.name="No moderator", RAM=RAM1, data=my.df)
summary(fit0)

## Get the SRMR
osmasemSRMR(fit0)

## Get the transformed variance component of the random effects
VarCorr(fit0)

## "lag" as a moderator on A matrix
A1 <- matrix(c(0,0,0,0,
               0,0,0,0,
               "0*data.Lag","0*data.Lag",0,0,
               "0*data.Lag","0*data.Lag",0,0),
nrow=4, ncol=4, byrow=TRUE)

## M1 <- create.vechsR(A0=RAM1$A, S0=RAM1$S, Ax=A1)
## Fit the model
## fit1 <- osmasem(model.name="Lag as a moderator for Amatrix", Mmatrix=M1,
## Tmatrix=T0, data= my.df)

fit1 <- osmasem(model.name="Lag as a moderator for Amatrix",
                RAM=RAM1, Ax=A1, data= my.df)
summary(fit1)
VarCorr(fit1)

## Compare the models with and without the moderator "lag"
anova(fit1, fit0)

## Calculate the R2
osmasemR2(fit0, fit1)

## End(Not run)
Description

The data set includes 28 studies on 14 items measuring the Hospital Anxiety and Depression Scale (HADS) Reported by Norton et al. (2013).

Usage

data(Norton13)

Details

The variables are:

- `data` A list of 28 studies of correlation matrices. The variables are 14 items (x1 to x14) measuring HADS.
- `n` A vector of sample sizes
- `population` A vector of the population of the data
- `group` A vector of classification into patients vs. non-patients based on population

Source


References


Examples

data(Norton13)

---

osmasem | *One-stage meta-analytic structural equation modeling*
---

Description

It fits MASEM with the one-stage MASEM (OSMASEM) approach.
Usage

```r
osmasem(model.name="osmasem", RAM=NULL, Mmatrix=NULL,
    Tmatrix=NULL, Jmatrix=NULL, Ax=NULL, Sx=NULL,
    A.lbound=NULL, A.ubound=NULL,
    RE.type=c("Diag", "Symm"), data,
    subset.variables=NULL, subset.rows=NULL,
    intervals.type = c("z", "LB"),
    mxModel.Args=NULL, mxRun.Args=NULL,
    suppressWarnings=TRUE, silent=TRUE, run=TRUE, ...)"
```

Arguments

- `model.name` A string for the model name in `mxModel`.
- `RAM` A RAM object including a list of matrices of the model returned from `lavaan2RAM`. If it is given, `Mmatrix` and `Tmatrix` arguments will be ignored.
- `Mmatrix` A list of matrices of the model implied correlation matrix created by the `create.vechsR`. It is only required when `RAM` is null.
- `Tmatrix` A list of matrices of the heterogeneity variance-covariance matrix created by the `create.Tau2`. It is only required when `RAM` is null.
- `Jmatrix` The Jacobian matrix of the mean structure in `mxMatrix`. The covariance structure is `Jmatrix %&% Tau2 + Vi`. If it is not given, an identity matrix will be used.
- `Ax` A `Amatrix` of a list of `Amatrix` with definition variables as the moderators of the `Amatrix`. It is used to create the `Mmatrix`.
- `Sx` A `Smatrix` of a list of `Smatrix` with definition variables as the moderators of the `Smatrix`. It is used to create the `Mmatrix`.
- `A.lbound` A matrix of lower bound of the `Amatrix`. If a scalar is given, the `lbound` matrix will be filled with this scalar.
- `A.ubound` A matrix of upper bound of the `Amatrix`. If a scalar is given, the `ubound` matrix will be filled with this scalar.
- `RE.type` Type of the random effects.
- `data` A list of data created by the `Cor2DataFrame`.
- `subset.variables` A character vector of the observed variables selected for the analysis.
- `subset.rows` A logical vector of the same length as the number of rows in the data to select which rows are used in the analysis.
- `intervals.type` Either `z` (default if missing) or `LB`. If it is `z`, it calculates the 95% confidence intervals (CIs) based on the estimated standard error. If it is `LB`, it calculates the 95% likelihood-based CIs on the parameter estimates.
- `mxModel.Args` A list of arguments passed to `mxModel`.
- `mxRun.Args` A list of arguments passed to `mxRun`.
- `suppressWarnings` Logical. If it is `TRUE`, warnings are suppressed. This argument is passed to `mxRun`.
silent Logical. An argument is passed to \texttt{mxRun}
run Logical. If \texttt{FALSE}, only return the \texttt{mx} model without running the analysis.
\texttt{...} Not used yet.

\textbf{Value}

An object of class \texttt{osmasem}

\textbf{Author(s)}

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

\textbf{References}


\textbf{See Also}

\texttt{Cor2DataFrame, create.vechsR, create.Tau2, create.V, osmasem, Nohe15}

---

\textbf{osmasemR2}  \hspace{1cm} \textit{Calculate the R2 in OSMASEM and OSMASEM3L}

\textbf{Description}

It calculates the R2 of the moderators in explaining the variances in the heterogeneity variances.

\textbf{Usage}

\texttt{osmasemR2(model1, model0, R2.truncate=TRUE)}

\textbf{Arguments}

\begin{itemize}
  \item \texttt{model1} An object in class \texttt{osmasem}.
  \item \texttt{model0} An object in class \texttt{osmasem}.
  \item \texttt{R2.truncate} Whether to truncate the negative R2 to zero.
\end{itemize}

\textbf{Value}

\texttt{model1} and \texttt{model0} are the models with and without the moderators, respectively. The function does not check whether the models are nested. It is the users’ responsibility to make sure that the models with and without the moderators are nested. It returns a list of the diagonals of the heterogeneity variances of the models without and with the moderators, and the R2.

\textbf{Author(s)}

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>
osmasemSRMR

### See Also

osmasem

---

#### osmasemSRMR

*Calculate the SRMR in OSMASEM and OSMASEM3L*

#### Description

It calculates the standardized root mean squared residuals (SRMR) in OSMASEM and OSMASEM3L.

#### Usage

```r
osmasemSRMR(x)
```

#### Arguments

- **x**: An OSMASEM object without any moderators.

#### Value

It calculates the model implied correlation matrix and its saturated counterpart to calculate the SRMR. It should be noted that the heterogeneity variances are ignored in the calculations.

#### Author(s)

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

---

#### pattern.n

*Display the Accumulative Sample Sizes for the Covariance Matrix*

#### Description

It displays the accumulative sample sizes for the covariance matrix.

#### Usage

```r
pattern.n(x, n)
```

#### Arguments

- **x**: A list of square matrices
- **n**: A vector of sample sizes.
### Description

It displays the pattern of missing data (or pattern of data that are present) of a list of square matrices with the same dimensions.

### Usage

```r
pattern.na(x, show.na = TRUE, type = c("tssem", "osmasem"))
```

### Arguments

- **x**: A list of square matrices
- **show.na**: If it is TRUE, it shows the pattern of missing data. If it is FALSE, it shows the pattern of data that are present.
- **type**: If it is tssem, it reports the pattern of missing correlations for the tssem approach. If it is osmasem, it reports the pattern of missing correlations for the data created by `Cor2DataFrame`.

### Value

A square matrix of numerical values with the same dimensions of the input matrices.

### Author(s)

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>
Examples

## Show the pattern of missing data
pattern.na(Hunter83$data, show.na=TRUE)

<table>
<thead>
<tr>
<th></th>
<th>Ability</th>
<th>Knowledge</th>
<th>Work sample</th>
<th>Supervisor</th>
</tr>
</thead>
<tbody>
<tr>
<td># Ability</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td># Knowledge</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td># Work sample</td>
<td>3</td>
<td>4</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td># Supervisor</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>

## Show the pattern of data that are present
pattern.na(Hunter83$data, show.na=FALSE)

<table>
<thead>
<tr>
<th></th>
<th>Ability</th>
<th>Knowledge</th>
<th>Work sample</th>
<th>Supervisor</th>
</tr>
</thead>
<tbody>
<tr>
<td># Ability</td>
<td>13</td>
<td>11</td>
<td>11</td>
<td>12</td>
</tr>
<tr>
<td># Knowledge</td>
<td>11</td>
<td>12</td>
<td>10</td>
<td>11</td>
</tr>
<tr>
<td># Work sample</td>
<td>11</td>
<td>10</td>
<td>12</td>
<td>11</td>
</tr>
<tr>
<td># Supervisor</td>
<td>12</td>
<td>11</td>
<td>11</td>
<td>13</td>
</tr>
</tbody>
</table>

plot

Plot methods for various objects

Description

It plots the models from either the lavaan model or meta, wls, and osmasem objects.

Usage

## S3 method for class 'meta'
plot(x, effect.sizes, add.margin = 0.1, interval = 0.95,
     main = "Effect Sizes and their Confidence Ellipses",
     axis.labels = paste("Effect size ", effect.sizes, sep = " "),
     study.col = "black", study.pch = 19, study.min.cex = 0.8,
     study.weight.plot = FALSE, study.ellipse.plot = TRUE,
     study.ellipse.col = "black", study.ellipse.lty = 2,
     study.ellipse.lwd = 0.5, estimate.col = "blue",
     estimate.pch = 18, estimate.cex = 2,
     estimate.ellipse.plot = TRUE, estimate.ellipse.col = "red",
     estimate.ellipse.lty = 1, estimate.ellipse.lwd = 2,
     randeff.ellipse.plot = TRUE, randeff.ellipse.col = "green",
     randeff.ellipse.lty = 1, randeff.ellipse.lwd = 2,
     univariate.plot = TRUE, univariate.lines.col = "gray",
     univariate.lines.lty = 3, univariate.lines.lwd = 1,
     univariate.polygon.width = 0.02,
     univariate.polygon.col = "red",
     univariate.arrows.col = "green", univariate.arrows.lwd = 2,
     diag.panel = FALSE, xlim=NULL, ylim=NULL, ...)

## S3 method for class 'character'
plot(x, fixed.x=FALSE, nCharNodes=0, nCharEdges=0,
    layout=c("tree", "circle", "spring", "tree2", "circle2"),
    sizeMan=8, sizeLat=8, edge.label.cex=1.3, color="white", ...)
## S3 method for class 'wls'
plot(x, manNames=NULL, latNames=NULL, labels=c("labels", "RAM"),
    what="est", nCharNodes=0, nCharEdges=0,
    layout=c("tree", "circle", "spring", "tree2", "circle2"),
    sizeMan=8, sizeLat=8, edge.label.cex=1.3, color="white",
    weighted=FALSE, ...)
## S3 method for class 'osmasem'
plot(x, manNames=NULL, latNames=NULL, labels=c("labels", "RAM"),
    what="est", nCharNodes=0, nCharEdges=0,
    layout=c("tree", "circle", "spring", "tree2", "circle2"),
    sizeMan=8, sizeLat=8, edge.label.cex=1.3, color="white",
    weighted=FALSE, ...)

Arguments

x An object returned from either a lavean model class character, osmasem, osmasem3L, wls or meta
effect.sizes Numeric values indicating which effect sizes to be plotted. At least two ef-
fect sizes are required. To plot the effect sizes of $y_1$ and $y_2$, one may use
effect.sizes=c(1,2). If it is missing, all effect sizes will be plotted in a
pairwise way.
add.margin Value for additional margins on the left and bottom margins.
interval Interval for the confidence ellipses.
main Main title of each plot. If there are multiple plots, a vector of character titles
may be used.
axis.labels Labels for the effect sizes.
study.col The color for individual studies. See col in par.
study.pch Plotting character of individual studies. See pch in points.
study.min.cex The minimum value of cex for individual studies. See cex in par.
study.weight.plot Logical. If TRUE, the plotting size of individual studies (cex) will be proportional
to one over the square root of the determinant of the sampling covariance matrix
of the study.
study.ellipse.plot Logical. If TRUE, the confidence ellipses of individual studies are plotted.
study.ellipse.col The color of the confidence ellipses of individual studies. See col in par.
study.ellipse.lty The line type of the confidence ellipse of individual studies. See lty in par.
study.ellipse.lwd The line width of the confidence ellipse of individual studies. See lwd in par.
estimate.col The color of the estimated effect size. See col in par.
estimate.pch  Plotting character of the estimated effect sizes. See pch in points.
estimate.cex  The amount of plotting of the estimated effect sizes. See cex in par.
estimate.ellipse.plot  Logical. If TRUE, the confidence ellipse of the estimated effect sizes will be plotted.
estimate.ellipse.col  The color of the confidence ellipse of the estimated effect sizes. See col in par.
estimate.ellipse.lty  The line type of the confidence ellipse of the estimated effect sizes. See lty in par.
estimate.ellipse.lwd  The line width of the confidence ellipse of the estimated effect sizes. See lwd in par.
randeff.ellipse.plot  Logical. If TRUE, the confidence ellipses of the random effects will be plotted.
randeff.ellipse.col  Color of the confidence ellipses of the random effects. See col in par.
randeff.ellipse.lty  The line type of the confidence ellipses of the random effects. See lty in par.
randeff.ellipse.lwd  The line width of the confidence ellipses of the random effects. See lwd in par.
univariate.plot  Logical. If TRUE, the estimated univariate effect sizes will be plotted.
univariate.lines.col  The color of the estimated univariate effect sizes. See col in par.
univariate.lines.lty  The line type of the estimated univariate effect sizes. See lty in par.
univariate.lines.lwd  The line width of the estimated univariate effect sizes. See lwd in par.
univariate.polygon.width  The width of the polygon of the estimated univariate effect sizes.
univariate.polygon.col  The color of the polygon of the estimated univariate effect sizes.
univariate.arrows.col  The color of the arrows of the estimated univariate effect sizes.
univariate.arrows.lwd  The line width of the arrows of the estimated univariate effect sizes.
diag.panel  Logical. If TRUE, diagonal panels will be created. They can then be used for forrest plots for univariate meta-analysis.
xlim  NULL or a numeric vector of length 2; if it is NULL, it provides defaults estimated from the data.
ylim  NULL or a numeric vector of length 2; if it is NULL, it provides defaults estimated from the data.
fixed.x  Argument passed to semPlotModel.
manNames  Argument passed to semPaths
latNames  Argument passed to semPaths
labels  Argument passed to semPaths
what  Argument passed to semPaths
nCharNodes  Argument passed to semPaths
nCharEdges  Argument passed to semPaths
layout  Argument passed to semPaths
color  Argument passed to semPaths
sizeMan  Argument passed to semPaths
sizeLat  Argument passed to semPaths
edge.label.cex  Argument passed to semPaths
weighted  Argument passed to semPaths
... Further arguments passed to the methods.

Note
The estimated effect sizes and random effects are based on the labels Intercept1, Intercept2, ... and Tau2_1_1, Tau2_2_1, Tau2_2_2, etc. At least two effect sizes are required for this function.

Author(s)
Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

References

See Also
Berkey98, wvs94a meta2semPlot semPaths

Examples

```r
## Not run:
## lavaan model
model <- "y ~ m + x
              m ~ x"
plot(model)

## End(Not run)
```
Print Methods for various Objects

Description

Print methods for the tssem1FEM, tssem1FEM.cluster, tssem1REM, wls, meta, meta3LFIML, reml, uniR1 and impliedR objects.

Usage

```
## S3 method for class 'tssem1FEM'
print(x, ...)
## S3 method for class 'tssem1FEM.cluster'
print(x, ...)
## S3 method for class 'tssem1REM'
print(x, ...)
## S3 method for class 'wls'
print(x, ...)
## S3 method for class 'meta'
print(x, ...)
## S3 method for class 'meta3LFIML'
print(x, ...)
## S3 method for class 'reml'
print(x, ...)
## S3 method for class 'uniR1'
print(x, ...)
## S3 method for class 'impliedR'
print(x, ...)
```

Arguments

- `x` An object returned from either class tssem1FEM, class tssem1FEM.cluster, class tssem1REM, class wls, class meta, class meta3LFIML, class reml, class uniR1 or class impliedR
- `...` Further arguments to be passed to summary.default or unused.

Author(s)

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

See Also

tsseml, wls, meta, reml
rCor

Generate (Nested) Sample/Population Correlation/Covariance Matrices

Description

It generates (nested) random sample or population correlation or covariance matrices. `rCor()` is a wrapper to call `rCorPop()` and then `rCorSam()`.

Usage

```r
rCor(Sigma, V, n, corr=TRUE, raw.data=FALSE, nonPD.pop=c("replace", "nearPD", "accept"), nonPD.sam=c("stop", "nearPD"))
rCorPop(Sigma, V, k, corr=TRUE, nonPD.pop=c("replace", "nearPD", "accept"))
rCorSam(Sigma, n, corr=TRUE, raw.data=FALSE, nonPD.sam=c("stop", "nearPD"))
rCor3L(Sigma, V.B, V.W, n, cluster, corr=TRUE, raw.data=FALSE, nonPD.pop=c("replace", "nearPD", "accept"), nonPD.sam=c("stop", "nearPD"))
```

Arguments

- **Sigma**: A list of population correlation/covariance matrices or a single matrix.
- **V**: A variance-covariance matrix of Sigma.
- **V.B**: A variance-covariance matrix of between-study Sigma.
- **V.W**: A variance-covariance matrix of within-study Sigma.
- **n**: A vector or a single sample sizes.
- **cluster**: A vector of number of studies in clusters.
- **corr**: Logical. Whether to generate correlation or covariance matrices.
- **raw.data**: Logical. Whether correlation/covariance matrices are generated via raw.data or directly from a Wishart distribution.
- **nonPD.pop**: If it is replace, generated non-positive definite matrices are replaced by generated new ones which are positive definite. If it is nearPD, they are replaced by nearly positive definite matrices by calling Matrix::nearPD(). If it is accept, they are accepted.
- **nonPD.sam**: If it is stop, the program stops when the inputs in the rCorSam are non-positive definite. If it is nearPD, they are replaced by nearly positive definite matrices by calling Matrix::nearPD().
- **k**: A vector or a single number of studies.

Value

An object of the generated population/sample correlation/covariance matrices.
Author(s)

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

Examples

```r
Sigma <- matrix(c(1, .2, .3, .2, 1, .4, .3, .4, 1), ncol=3, nrow=3)
V <- diag(c(.1, .1, .1))

## Generate two population correlation matrices
Pop.corr <- rCorPop(Sigma, V, k=2)
Pop.corr

summary(Pop.corr)

## Generate two sample correlation matrices
rCorSam(Sigma=Pop.corr, n=c(10, 10))

## The above code is the same as the following one
rCor(Sigma, V, n=c(10, 10))
```

readData

Read External Correlation/Covariance Matrices

Description

It reads full/lower triangle/stacked vectors of correlation/covariance data into a list of correlation/covariance matrices.

Usage

```r
readFullMat(file, ...)
readStackVec(file, ...)
readLowTriMat(file, no.var, ...)
```

Arguments

- `file`  
  File name of the data.
- `no.var`  
  The number of variables in the data.
- `...`  
  Further arguments to be passed to `scan` for `readLowTriMat` and to `read.table` for `readFullMat` and `readStackVec`.

Value

A list of correlation/covariance matrices.
Examples

## Not run:
## Write two full correlation matrices into a file named "fullmat.dat".
## x2 is missing in the second matrix.
## The content of "fullmat.dat" is
# 1.0 0.3 0.4
# 0.3 1.0 0.5
# 0.4 0.5 1.0
# 1.0 NA 0.4
# NA NA NA
# 0.4 NA 1.0
## cat("1.0 0.3 0.4\n0.3 1.0 0.5\n0.4 0.5 1.0
1.0 NA 0.4
NA NA NA
0.4 NA 1.0",
## file="fullmat.dat", sep=""
## my.full <- readFullMat("fullmat.dat")
## my.full

## Write two lower triangle correlation matrices into a file named "lowertriangle.dat".
## x2 is missing in the second matrix.
## The content of "lowertriangle.dat" is
# 1.0
# 0.3 1.0
# 0.4 0.5 1.0
# 1.0
# NA NA
# 0.4 NA 1.0
## cat("1.0\n0.3 1.0\n0.4 0.5 1.0\n1.0\nNA NA\n0.4 NA 1.0",
## file="lowertriangle.dat", sep=""
## my.lowertri <- readLowTriMat(file = "lowertriangle.dat", no.var = 3)
## my.lowertri
## my.vec <- readStackVec("stackvec.dat")

## my.vec
# $`1`
#  x1  x2  x3
#  x1  1.0  0.3  0.4
#  x2  0.3  1.0  0.5
#  x3  0.4  0.5  1.0
#
# $`2`
#  x1  x2  x3
#  x1  1.0  NA  0.4
#  x2  NA  NA  NA
#  x3  0.4  NA  1.0

## End(Not run)

---

### Description

It estimates the variance components of random-effects in univariate and multivariate meta-analysis with restricted (residual) maximum likelihood (REML) estimation method.

### Usage

```r
reml(y, v, x, data, RE.constraints = NULL, RE.startvalues = 0.1,
```
RE.lbound = 1e-10, intervals.type = c("z", "LB"),
model.name="Variance component with REML",
suppressWarnings = TRUE, silent = TRUE, run = TRUE, ...)

**Arguments**

- **y** A vector of effect size for univariate meta-analysis or a $k \times p$ matrix of effect sizes for multivariate meta-analysis where $k$ is the number of studies and $p$ is the number of effect sizes.
- **v** A vector of the sampling variance of the effect size for univariate meta-analysis or a $k \times p^*$ matrix of the sampling covariance matrix of the effect sizes for multivariate meta-analysis where $p^* = p(p + 1)/2$. It is arranged by column major as used by `vech`.
- **x** A predictor or a $k \times m$ matrix of predictors where $m$ is the number of predictors.
- **data** An optional data frame containing the variables in the model.
- **RE.constraints** A $p \times p$ matrix specifying the variance components of the random effects. If the input is not a matrix, it is converted into a matrix by `as.matrix()`. The default is that all covariance/variance components are free. The format of this matrix follows `as.mxMatrix`. Elements of the variance components can be constrained equally by using the same labels. If a zero matrix is specified, it becomes a fixed-effects meta-analysis.
- **RE.startvalues** A vector of $p$ starting values on the diagonals of the variance component of the random effects. If only one scalar is given, it will be repeated across the diagonals. Starting values for the off-diagonals of the variance component are all 0. A $p \times p$ symmetric matrix of starting values is also accepted.
- **RE.lbound** A vector of $p$ lower bounds on the diagonals of the variance component of the random effects. If only one scalar is given, it will be repeated across the diagonals. Lower bounds for the off-diagonals of the variance component are set at `NA`. A $p \times p$ symmetric matrix of the lower bounds is also accepted.
- **intervals.type** Either `z` (default if missing) or `LB`. If it is `z`, it calculates the 95% Wald confidence intervals (CIs) based on the z statistic. If it is `LB`, it calculates the 95% likelihood-based CIs on the parameter estimates. Note that the z values and their associated p values are based on the z statistic. They are not related to the likelihood-based CIs.
- **model.name** A string for the model name in `mxModel`.
- **suppressWarnings** Logical. If `TRUE`, warnings are suppressed. It is passed to `mxRun`.
- **silent** Logical. An argument to be passed to `mxRun`.
- **run** Logical. If `FALSE`, only return the mx model without running the analysis.
- **...** Further arguments to be passed to `mxRun`.

**Details**

Restricted (residual) maximum likelihood obtains the parameter estimates on the transformed data that do not include the fixed-effects parameters. A transformation matrix $M = I - X(X'X)^{-1}X$
is created based on the design matrix $X$ which is just a column vector when there is no predictor in $x$. The last $N$ redundant rows of $M$ is removed where $N$ is the rank of $X$. After pre-multiplying by $M$ on $y$, the parameters of fixed-effects are removed from the model. Thus, only the parameters of random-effects are estimated.

An alternative but equivalent approach is to minimize the $-2\times\log$-likelihood function:

$$
\log(\det|V + T^2|) + \log(\det|X'(V + T^2)^{-1}X|) + (y - X\hat{\alpha})'(V + T^2)^{-1}(y - X\hat{\alpha})
$$

where $V$ is the known conditional sampling covariance matrix of $y$, $T^2$ is the variance component of the random effects, and $\hat{\alpha} = (X'(V + T^2)^{-1}X)^{-1}X'(V + T^2)^{-1}y$. `reml()` minimizes the above likelihood function to obtain the parameter estimates.

Value

An object of class `reml` with a list of

- `call`: Object returned by `match.call`
- `data`: A data matrix of $y$, $v$ and $x$
- `no.y`: No. of effect sizes
- `no.x`: No. of predictors
- `miss.vec`: A vector indicating missing data. Studies will be removed before the analysis if they are `TRUE`
- `mx.fit`: A fitted object returned from `mxRun`

Note

`reml` is more computationally intensive than `meta`. Moreover, `reml` is more likely to encounter errors during optimization. Since a likelihood function is directly employed to obtain the parameter estimates, there is no number of studies and number of observed statistics returned by `mxRun`. Ad-hoc steps are used to modify `mx.fit@runstate$objectives[[1]]@numObs` and `mx.fit@runstate$objectives[[1]]@numStats`.

Author(s)

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

References


See Also

`meta`, `reml3`, `Hox02`, `Berkey98`
Estimate Variance Components in Three-Level Univariate Meta-Analysis with Restricted (Residual) Maximum Likelihood Estimation

Description
It estimates the variance components of random-effects in three-level univariate meta-analysis with restricted (residual) maximum likelihood (REML) estimation method.

Usage
```r
## Depreciated in the future
reml3(y, v, cluster, x, data, RE2.startvalue=0.1, RE2.lbound=1e-10,
     RE3.startvalue=RE2.startvalue, RE3.lbound=RE2.lbound, RE.equal=FALSE,
     intervals.type=c("z", "LB"), model.name="Variance component with REML",
     suppressWarnings=TRUE, silent=TRUE, run=TRUE, ...)
reml3L(y, v, cluster, x, data, RE2.startvalue=0.1, RE2.lbound=1e-10,
      RE3.startvalue=RE2.startvalue, RE3.lbound=RE2.lbound, RE.equal=FALSE,
      intervals.type=c("z", "LB"), model.name="Variance component with REML",
      suppressWarnings=TRUE, silent=TRUE, run=TRUE, ...)
```

Arguments
- `y`: A vector of \(k\) studies of effect size.
- `v`: A vector of \(k\) studies of sampling variance.
- `cluster`: A vector of \(k\) characters or numbers indicating the clusters.
- `x`: A predictor or a \(k \times m\) matrix of level-2 and level-3 predictors where \(m\) is the number of predictors.
- `data`: An optional data frame containing the variables in the model.
- `RE2.startvalue`: Starting value for the level-2 variance.
- `RE2.lbound`: Lower bound for the level-2 variance.
- `RE3.startvalue`: Starting value for the level-3 variance.
- `RE3.lbound`: Lower bound for the level-3 variance.
- `RE.equal`: Logical. Whether the variance components at level-2 and level-3 are constrained equally.
- `intervals.type`: Either \(z\) (default if missing) or \(LB\). If it is \(z\), it calculates the 95% Wald confidence intervals (CIs) based on the \(z\) statistic. If it is \(LB\), it calculates the 95% likelihood-based CIs on the parameter estimates. Note that the \(z\) values and their associated \(p\) values are based on the \(z\) statistic. They are not related to the likelihood-based CIs.
- `model.name`: A string for the model name in `mxModel`.
- `suppressWarnings`: Logical. If \(TRUE\), warnings are suppressed. It is passed to `mxRun`.
silent Logical. Argument to be passed to mxRun.
run Logical. If FALSE, only return the mx model without running the analysis.

Details
Restricted (residual) maximum likelihood obtains the parameter estimates on the transformed data that do not include the fixed-effects parameters. A transformation matrix $M = I - X(X'X)^{-1}X$ is created based on the design matrix $X$ which is just a column vector when there is no predictor in $x$. The last $N$ redundant rows of $M$ is removed where $N$ is the rank of $X$. After pre-multiplying by $M$ on $y$, the parameters of fixed-effects are removed from the model. Thus, only the parameters of random-effects are estimated.

An alternative but the equivalent approach is to minimize the $-2\log$-likelihood function:

$$
\log(\text{det}|V + T^2|) + \log(\text{det}|X'(V + T^2)^{-1}X|) + (y - X\hat{\alpha})'(V + T^2)^{-1}(y - X\hat{\alpha})
$$

where $V$ is the known conditional sampling covariance matrix of $y$, $T^2$ is the variance component combining level-2 and level-3 random effects, and $\hat{\alpha} = (X'(V + T^2)^{-1}X)^{-1}X'(V + T^2)^{-1}y$. reml() minimizes the above likelihood function to obtain the parameter estimates.

Value
An object of class reml with a list of

- call Object returned by match.call
- data A data matrix of y, v, and x
- mx.fit A fitted object returned from mxRun

Note
reml is more computationally intensive than meta. Moreover, reml is more likely to encounter errors during optimization. Since a likelihood function is directly employed to obtain the parameter estimates, there is no number of studies and number of observed statistics returned by mxRun. Ad-hoc steps are used to modify mx.fit@runstate$objectives[[1]]@numObs and mx.fit@runstate$objectives[[1]]@numStats.

Author(s)
Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

References


rerun

Rerun models via mxTryHard()

Description

It reruns models via mxTryHard().

Usage

rerun(object, autofixtau2=FALSE, extraTries=10, ...)

Arguments

- **object**: An object of either class `tssem1FEM`, class `tssem1REM`, class `wls`, class `meta`, class `reml`, class `osmasem`, class `osmasem3L`, and class `MxModel`.
- **autofixtau2**: Logical. Whether automatically fixes elements of tau2 with NA of standard errors. It only works for objects of class `tssem1REM`, class `meta`, and class `osmasem`.
- **extraTries**: The number of attempts to run the model in addition to the first.
- **...**: Further arguments to be passed to `mxTryHard`.

Author(s)

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

Examples

```r
## Not run:
random1 <- tssem(Digman97$data, Digman97$n, method="REM", RE.type="Diag")
random1_rerun <- rerun(random1)
summary(random1_rerun)

## End(Not run)
```
Description

The data set includes 45 studies on the influence of affective teacher-student relationships on students’ school engagement and achievement reported by Roorda et al. (2011).

Usage

data(Roorda11)

Details

The variables are:

data A list of 45 studies of correlation matrices. The variables are pos (positive teacher-student relations), neg (negative teacher-student relations), enga (student engagement), and achiev (student achievement).

n A vector of sample sizes

SES A vector of average socio-economic status (SES) of the samples

Source


References


Examples

## Not run:

## Random-effects model: First stage analysis
random1 <- tssem1(Cov = Roorda11$data, n = Roorda11$n, method = "REM",
                          RE.type = "Diag")
summary(random1)

varnames <- c("pos", "neg", "enga", "achiev")

## Prepare a regression model using create.mxMatrix()
A <- create.mxMatrix(c(0,0,0,0,
                          0,0,0,0,
                          0,0,0,0,
                          0,0,0,0),
                          type = "Diag",
                          dimnames = list(varnames, varnames))
Scalco17

Correlation Matrices from Scalco et al. (2017)

Description
The data set includes correlation matrices using the theory of planned behavior to predict organic food consumption reported by Scalco17 et al. (2017).

Usage
data(Scalco17)

Details
A list of data with the following structure:

- **data**: A list of correlation matrices. The variables are *ATT* (attitude), *SN* (subjective norm), *PBC* (perceived behavior control), *BI* (behavioral intention), and *BEH* (behavior)
- **n**: A vector of sample sizes
- **Age**: A vector of the mean age of the samples
- **Female**: A vector of the percentage of the female samples
Source

Examples
data(Scalco17)

---

**smdMES**  
*Compute Effect Sizes for Multiple End-point Studies*

**Description**
It computes the standardized mean differences and their asymptotic sampling covariance matrix for two multiple end-point studies with \( p \) effect sizes.

**Usage**
```
smdMES(m1, m2, V1, V2, n1, n2,  
    homogeneity=c("covariance", "correlation", "none"),  
    bias.adjust=TRUE, list.output=TRUE, lavaan.output=FALSE)
```

**Arguments**
- \( m1 \): A vector of \( p \) sample means of the first group.
- \( m2 \): A vector of \( p \) sample means of the second group.
- \( V1 \): A \( p \) by \( p \) sample covariance matrix of the first group.
- \( V2 \): A \( p \) by \( p \) sample covariance matrix of the second group.
- \( n1 \): The sample size of the first group.
- \( n2 \): The sample size of the second group.
- \( \text{homogeneity} \): If it is covariance (the default), homogeneity of covariance matrices is assumed. The common standard deviations are used as the standardizers in calculating the effect sizes. If it is correlation, homogeneity of correlation is not assumed. The standard deviations of the first group are used as the standardizer in calculating the effect sizes. If it is none, no homogeneity assumption is made. The standard deviations of the first group are used as the standardizer in calculating the effect sizes.
- \( \text{bias.adjust} \): If it is TRUE (the default), the effect sizes are adjusted for small bias by multiplying \( 1 - 3/(4 \times (n1 + n2) - 9) \).
- \( \text{list.output} \): If it is TRUE (the default), the effect sizes and their sampling covariance matrix are outputed as a list. If it is FALSE, they will be stacked into a vector.
- \( \text{lavaan.output} \): If it is FALSE (the default), the effect sizes and its sampling covariance matrix are reported. If it is TRUE, it outputs the fitted *lavaan-class* object.
Details

Gleser and Olkin (2009) introduce formulas to calculate the standardized mean differences and their sampling covariance matrix for multiple end-point studies under the assumption of homogeneity of the covariance matrix. This function uses a structural equation modeling (SEM) approach introduced in Chapter 3 of Cheung (2015) to calculate the same estimates. The SEM approach is more flexible in two ways: (1) it allows homogeneity of covariance or correlation matrices or not; and (2) it allows users to test this assumption by checking the fitted lavaan-class object.

Author(s)

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

References


See Also

Gleser94, smdMTS, calEffSizes

Examples

```r
## Not run:
## Sample means for the two constructs in Group 1
m1 <- c(2.5, 4.5)

## Sample means for the two constructs in Group 2
m2 <- c(3, 5)

## Sample covariance matrix in Group 1
V1 <- matrix(c(3,2,2,3), ncol=2)

## Sample covariance matrix in Group 2
V2 <- matrix(c(3.5,2.1,2.1,3.5), ncol=2)

## Sample size in Group 1
n1 <- 20

## Sample size in Group 2
n2 <- 25

## SMD with the assumption of homogeneity of covariance matrix
smdMES(m1, m2, V1, V2, n1, n2, homogeneity="cov", bias.adjust=TRUE,
```
## SMD with the assumption of homogeneity of correlation matrix
smdMES(m1, m2, V1, V2, n1, n2, homogeneity="cor", bias.adjust=TRUE, lavaan.output=FALSE)

## SMD without any assumption of homogeneity
smdMES(m1, m2, V1, V2, n1, n2, homogeneity="none", bias.adjust=TRUE, lavaan.output=FALSE)

## Output the fitted lavaan model
## It provides a likelihood ratio test to test the null hypothesis of
## homogeneity of variances.
fit <- smdMES(m1, m2, V1, V2, n1, n2, homogeneity="cov", bias.adjust=TRUE, lavaan.output=TRUE)

lavaan::summary(fit)
lavaan::parameterestimates(fit)

## End(Not run)

---

### smdMTS

**Compute Effect Sizes for Multiple Treatment Studies**

#### Description

It computes the standardized mean differences and their asymptotic sampling covariance matrix for
$k$ multiple treatment studies. The first group is assumed as the control group.

#### Usage

```r
smdMTS(m, v, n, homogeneity=c("variance", "none"), bias.adjust=TRUE, all.comparisons=FALSE, list.output=TRUE, lavaan.output=FALSE)
```

#### Arguments

- `m` A vector of $k$ sample means.
- `v` A vector of $k$ sample variances.
- `n` A vector of $k$ sample sizes.
- `homogeneity` If it is variance (the default), homogeneity of variances is assumed. The common standard deviation is used as the standardizer in calculating the effect sizes. If it is none, homogeneity of variances is not assumed. The standard deviation of the first group is used as the standardizer in calculating the effect sizes.
- `bias.adjust` If it is TRUE (the default), the effect sizes are adjusted for small bias by multiplying $1 - 3/(4 \times (n1 + n2) - 9)$.  

all.comparisons
If it is FALSE (the default), all groups (except the first group) are compared against the first group. If it is TRUE, all pairwise comparisons are calculated. This may be useful in network meta-analysis.

list.output
If it is TRUE (the default), the effect sizes and their sampling covariance matrix are outputed as a list. If it is FALSE, they will be stacked into a vector.

lavaan.output
If it is FALSE (the default), the effect sizes and its sampling covariance matrix are reported. If it is TRUE, it outputs the fitted lavaan-class object.

Details
Gleser and Olkin (2009) introduce formulas to calculate the standardized mean differences and their sampling covariance matrix for multiple treatment studies under the assumption of homogeneity of the covariance matrix. This function uses a structural equation modeling (SEM) approach introduced in Chapter 3 of Cheung (2015) to calculate the same estimates. The SEM approach is more flexible in three ways: (1) it allows homogeneity of variances or not; (2) it allows users to test the assumption of homogeneity of variances by checking the fitted lavaan-class object; and (3) it may calculate all pairwise comparisons.

Author(s)
Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

References


See Also
Gleser94, smdMES, calEffSizes

Examples
## Not run:
## Sample means for groups 1 to 3
m <- c(5,7,9)

## Sample variances
v <- c(10,11,12)

## Sample sizes
n <- c(50,52,53)
## Assuming homogeneity of variances
smdMTS(m, v, n, homogeneity = "var", bias.adjust=TRUE, all.comparisons=FALSE, lavaan.output=FALSE)

## Not assuming homogeneity of variances and comparing all pairwise groups
## Please note that the SD of the first group is used as the standardizer
smdMTS(m, v, n, homogeneity = "none", bias.adjust=TRUE, all.comparisons=TRUE, lavaan.output=FALSE)

## Output the fitted lavaan model
## It provides a likelihood ratio test to test the null hypothesis of
## homogeneity of variances.
fit <- smdMTS(m, v, n, homogeneity = "var", bias.adjust=FALSE, all.comparisons=FALSE, lavaan.output=TRUE)
lavaan::summary(fit)
lavaan::parameterestimates(fit)

## End(Not run)

---

### Stadler15

#### Correlations from Stadler et al. (2015)

#### Description

The data set includes correlations between complex problem solving and intelligence reported by Stadler et al. (2015).

#### Usage

data(Stadler15)

#### Details

A list of data with the following structure:

- **ID**  ID of the effect sizes
- **Authors**  Authors of the studies
- **Year**  Year of the studies
- **N**  Sample size
- **CPSMeasure**  Complex problem solving (CPS) measure
- **IntelligenceMeasure**  Intelligence measure
- **r**  Correlation between CPS and intelligence
- **v**  Sampling variance of r
Source

---

**summary**

Summary Method for `tssem1`, `wls`, `meta`, and `meta3LFIML` Objects

**Description**

It summaries results for various class.

**Usage**

```r
## S3 method for class 'tssem1FEM'
summary(object, ...) 
## S3 method for class 'tssem1FEM.cluster'
summary(object, ...) 
## S3 method for class 'tssem1REM'
summary(object, robust=FALSE, ...) 
## S3 method for class 'wls'
summary(object, df.adjustment=0, ...) 
## S3 method for class 'wls.cluster'
summary(object, df.adjustment=0, ...) 
## S3 method for class 'meta'
summary(object, homoStat=TRUE, robust=FALSE, ...) 
## S3 method for class 'meta3LFIML'
summary(object, allX=FALSE, robust=FALSE, ...) 
## S3 method for class 'reml'
summary(object, ...) 
## S3 method for class 'CorPop'
summary(object, ...) 
## S3 method for class 'Cor3L'
summary(object, ...) 
## S3 method for class 'bootuniR2'
summary(object, probs=c(0, 0.1, 0.5, 0.9, 1),
          cutoff.chisq.pvalue=0.05, cutoff.CFI=0.9, cutoff.SRMR=0.1,
          cutoff.RMSEA=0.05, ...) 
## S3 method for class 'osmasem'
summary(object, fitIndices=FALSE, numObs, robust=FALSE, ...) 
## S3 method for class 'tssem1FEM'
print.summary(x, ...) 
## S3 method for class 'wls'
print.summary(x, ...) 
## S3 method for class 'meta'
print.summary(x, ...) 
## S3 method for class 'meta3LFIML'
```
print.summary(x, ...)  
## S3 method for class 'reml'  
print.summary(x, ...)  
## S3 method for class 'CorPop'  
print.summary(x, ...)  
## S3 method for class 'Cor3L'  
print.summary(x, ...)  
## S3 method for class 'bootuniR2'  
print.summary(x, ...)  

Arguments

object  An object returned from either class `tssem1FEM`, class `tssem1FEM.cluster`, class `tssem1REM`, class `wls`, class `wls.cluster`, class `meta`, class `meta3LFIML`, class `reml` or class `CorPop`.  
x  An object returned from either class `summary.tssem1FEM`, class `tssem1FEM.cluster`, class `summary.wls`, class `summary.meta`, class `summary.meta3LFIML`, class `summary.reml` or class `summary.CorPop`.  
homoStat  Logical. Whether to conduct a homogeneity test on the effect sizes.  
allX  Logical. Whether to report the predictors and the auxiliary variables.  
robust  Logical. Whether to use robust standard error from `imxRobustSE`.  
df.adjustment  Numeric. Adjust the degrees of freedom manually. It may be necessary if the df calculated is incorrect when `diag.constraints=TRUE`.  
probs  Quantiles for the parameter estimates.  
cutoff.chisq.pvalue  Cutoff of the p-value for the chi-square statistic.  
cutoff.CFI  The cutoff of the CFI.  
cutoff.SRMR  The cutoff of the SRMR.  
cutoff.RMSEA  The cutoff of the RMSEA.  
fitIndices  Whether to calculate the chi-square statistic and various goodness-of-fit indices in osmasem. Note. It may take a while since statistics of the saturated and independence models are required.  
numObs  The number of observations in calculating the fit statistics in osmasem. If it is missing, the total number of observations is used.  
...  Further arguments to be passed to `printCoefmat`  

Note

If the OpenMx status1 is either 0 or 1, the estimation is considered fine. If the OpenMx status1 is other values, it indicates estimation problems. Users should refer to https://openmx.ssri.psu.edu/wiki/errors for more details.

Author(s)

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>
## Tenenbaum02

**Correlation coefficients reported by Tenenbaum and Leaper (2002)**

### Description

Forty-eight studies reported by Tenenbaum and Leaper (2002, Table 1).

### Usage

```r
data(Tenenbaum02)
```

### Details

The variables are:

- **Authors**: Authors of the study
- **Year**: Year of publication
- **N**: Sample size
- **r**: Correlation between parents’ gender schemas and their offspring’s gender-related cognitions.
- **v**: Sampling variance of r
- **Author_gender**: Gender of the first author: "W"="woman", "M"="man"
- **Parent_type**: Parent type: "M"="mother", "F"="father", "MF"="mother and father"
- **Parent_predictor**: Parent predictor: "S"="self gender schema", "A"="gender attitudes about others"
- **Offspring_age**: Offspring age (months)
- **Offspring_type**: Offspring type: "D"="daughter", "S"="son", "DS"="daughter and son"
- **Offspring_outcome**: Offspring outcome: "S"="gender schema for self", "A"="gender attitudes toward others", "I"="gender-related interests and preferences", "W"="work-related attitudes"

### Source


### Examples

```r
## Not run:
data(Tenenbaum02)
## End(Not run)
```
First Stage of the Two-Stage Structural Equation Modeling (TSSEM)

Description

It conducts the first stage analysis of TSSEM by pooling correlation/covariance matrices. tssem1FEM() and tssem1REM() use fixed- and random-effects models, respectively. tssem1() is a wrapper of these functions.

Usage

tssem1(Cov, n, method=c("REM","FEM"), cor.analysis = TRUE, cluster=NULL, RE.type=c("Diag", "Symm", "Zero", "User"), RE.startvalues=0.1, RE.lbound=1e-10, RE.constraints=NULL, I2="I2q", acov=c("weighted", "individual", "unweighted"), asyCovOld=FALSE, model.name=NULL, suppressWarnings=TRUE, silent=TRUE, run=TRUE, ...)
tssem1FEM(Cov, n, cor.analysis=TRUE, model.name=NULL, cluster=NULL, suppressWarnings=TRUE, silent=TRUE, run=TRUE, ...)
tssem1REM(Cov, n, cor.analysis=TRUE, RE.type=c("Diag", "Symm", "Zero", "User"), RE.startvalues=0.1, RE.lbound=1e-10, RE.constraints=NULL, I2="I2q", acov=c("weighted", "individual", "unweighted"), asyCovOld=FALSE, model.name=NULL, suppressWarnings=TRUE, silent=TRUE, run=TRUE, ...)

Arguments

Cov A list of correlation/covariance matrices
n A vector of sample sizes
method Either "REM" (default if missing) or "FEM". If it is "REM", a random-effects meta-analysis will be applied. If it is "FEM", a fixed-effects meta-analysis will be applied.
cor.analysis Logical. The output is either a pooled correlation or a covariance matrix.
cluster A character vector in tssem3L1 and tssemRobust1 or a vector of characters or numbers indicating the clusters in tssem1. Analyses will be conducted for each cluster. It will be ignored when method="REM".
RE.type Either "Diag", "Symm", "Zero" or "User". If it is "Diag", a diagonal matrix is used for the random effects meaning that the random effects are independent. If it is "Symm" (default if missing), a symmetric matrix is used for the random effects on the covariances among the correlation (or covariance) vectors. If it is "Zero", there is no random effects which is similar to the conventional Generalized Least Squares (GLS) approach to fixed-effects analysis. "User", the user has to specify the variance component via the RE.constraints argument. This argument will be ignored when method="FEM".
RE.startvalues Starting values on the diagonals of the variance component of the random effects. It will be ignored when method="FEM".
RE.lbound  Lower bounds on the diagonals of the variance component of the random effects. It will be ignored when method="FEM".

RE.constraints  A $p^* \times p^*$ matrix specifying the variance components of the random effects, where $p^*$ is the number of effect sizes. If the input is not a matrix, it is converted into a matrix by as.matrix(). The default is that all covariance/variance components are free. The format of this matrix follows as.mxMatrix. Elements of the variance components can be constrained equally by using the same labels. If a zero matrix is specified, it becomes a fixed-effects meta-analysis.

I2  Possible options are "I2q", "I2hm" and "I2am". They represent the $I^2$ calculated by using a typical within-study sampling variance from the Q statistic, the harmonic mean and the arithmetic mean of the within-study sampling variances (Xiong, Miller, & Morris, 2010). More than one options are possible. If intervals.type="LB", 95% confidence intervals on the heterogeneity indices will be constructed.

acov  If it is individual, the sampling variance-covariance matrices are calculated based on individual correlation/covariance matrix. If it is either unweighted or weighted (the default), the average correlation/covariance matrix is calculated based on the unweighted or weighted mean with the sample sizes. The average correlation/covariance matrix is used to calculate the sampling variance-covariance matrices. This argument is ignored with the method="FEM" argument.

asyCovOld  Whether the old asyCov is used. See asyCov.

model.name  A string for the model name in mxModel.

suppressWarnings  Logical. If TRUE, warnings are suppressed. It is passed to mxRun.

silent  Logical. An argument to be passed to mxRun.

run  Logical. If FALSE, only return the mx model without running the analysis.

...  Further arguments to be passed to mxRun

Value

Either an object of class tssem1FEM for fixed-effects TSSEM, an object of class tssem1FEM.cluster for fixed-effects TSSEM with cluster argument, or an object of class tssem1REM for random-effects TSSEM.

Author(s)

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

References


**See Also**

`wls`, `Cheung09`, `Becker92`, `Digman97`, `issp89`, `issp05`

---

### tssemParaVar

Estimate the heterogeneity (SD) of the parameter estimates of the TSSEM object

#### Description

It estimates the heterogeneity of the parameter estimates of the TSSEM objects using either the bootstrap or the delta methods.

#### Usage

```r
tssemParaVar(tssem1.obj, tssem2.obj, method=c("bootstrap", "delta"),
             interval=0.8, Rep=50, output=c("data.frame", "matrices"),
             nonPD.pop=c("replace", "nearPD", "accept"))
```

#### Arguments

- `tssem1.obj`: An object of class `tssem1REM` returned from `tssem1()`.
- `tssem2.obj`: An object of class `wls` returned from `tssem2()` or `wls()`.
- `method`: If it is `bootstrap`, random correlation matrices are sampled from the `tssem1.obj` by the parametric bootstrap. If it is `delta`, the delta method is used to estimate the heterogeneity of the parameter estimates.
- `interval`: The desired interval, e.g., .8 or .95.
- `Rep`: The number interval. It is ignored when the method is `delta`.
- `output`: Either a `data.frame` or matrices of the output.
- `nonPD.pop`: If it is `replace`, generated non-positive definite matrices are replaced by generated new ones which are positive definite. If it is `nearPD`, they are replaced by nearly positive definite matrices by calling `Matrix::nearPD()`. If it is `accept`, they are accepted.

#### Details

The bootstrap method is based on the discussion in Cheung (2018) and Yu et al. (2016). The delta method is an alternative method to obtain the heterogeneity.
Value

Either a data.frame or matrices of the output.

Author(s)

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

References


See Also

bootuniR1, bootuniR2, Nohe15

---

**uniR1**

*First Stage analysis of the univariate R (uniR) approach*

Description

It conducts the first stage analysis of the uniR analysis by pooling elements of the correlation coefficients individually.

Usage

`uniR1(Cor, n, ...)`

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Cor</code></td>
<td>A list of correlation matrices</td>
</tr>
<tr>
<td><code>n</code></td>
<td>A vector of sample sizes</td>
</tr>
<tr>
<td><code>...</code></td>
<td>Further arguments which are currently ignored</td>
</tr>
</tbody>
</table>

Details

This function implements the univariate r approach proposed by Viswesvaran and Ones (1995) to conduct meta-analytic structural equation modeling (MASEM). It uses Schmidt and Hunter’s approach to combine correlation coefficients. It is included in this package for research interests. The two-stage structural equation modeling (TSSEM) approach is preferred (e.g., Cheung, 2015; Cheung & Chan, 2005).
uniR2

Value
An object of class uniR1 of the original data, the sample sizes, the harmonic mean of sample sizes, the average correlation matrix, the standard errors of the correlation matrix, and the standard deviations (heterogeneity) of the correlation matrix.

Author(s)
Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

References

See Also
uniR2, Becker09

uniR2

Second Stage analysis of the univariate R (uniR) approach

Description
It conducts the second stage analysis of the uniR analysis by fitting structural equation models on the average correlation matrix.

Usage
uniR2mx(x, RAM = NULL, Amatrix = NULL, Smatrix = NULL, Fmatrix = NULL, model.name=NULL, suppressWarnings=TRUE, silent=TRUE, run=TRUE, ...)
uniR2lavaan(x, model, ...)

Arguments
x An object of class uniR1 from uniR1.
RAM A RAM object including a list of matrices of the model returned from lavaan2RAM.
Amatrix If RAM is not specified, an Amatrix is required. An asymmetric matrix in the RAM specification with MxMatrix-class. If it is a matrix, it will be converted into MxMatrix-class by the as.mxMatrix function.
uniR2

S matriz
If RAM is not specified, an Smatrix is required. A symmetric matrix in the RAM specification with MxMatrix-class. If it is a matrix, it will be converted into MxMatrix-class by the as.mxMatrix function.

F matrix
If RAM is not specified, an Fmatrix is required. A filter matrix in the RAM specification with MxMatrix-class. If it is NULL (the default), an identity matrix with the same dimensions of Cov will be created. If it is a matrix, it will be converted into MxMatrix-class by the as.mxMatrix function. It is not required when there is no latent variable.

model.name
A string for the model name in mxModel. If it is missing, the default is "UniR2".

suppressWarnings
Logical. If TRUE, warnings are suppressed. It is passed to mxRun.

silent
Logical. An argument to be passed to mxRun

run
Logical. If FALSE, only return the mx model without running the analysis.

model
A model specified using lavaan syntax see model.syntax

...
Further arguments to be passed to either mxRun or sem. For sem, fixed.x=FALSE is passed automatically.

Details
This function implements the univariate r approach proposed by Viswesvaran and Ones (1995) to conduct meta-analytic structural equation modeling (MASEM). It treats the average correlation matrix as if it was a covariance matrix in fitting structural equation models. The harmonic mean of the sample sizes in combining correlation coefficients is used as the sample size in fitting structural equation models. It is included in this package for research interests. The two-stage structural equation modeling (TSSEM) approach is preferred (e.g., Cheung, 2015; Cheung & Chan, 2005).

Value
A fitted object returned from mxRun or sem.

Author(s)
Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

References


See Also
uniR1, lavaan2RAM, Becker09
Dataset on the effectiveness of multidimensional family therapy in treating adolescents with multiple behavior problems

Description

This dataset includes 61 effect sizes from 19 manuscripts nested from 8 studies reported by van der Pol et al. (2017). It studies the effectiveness of multidimensional family therapy in treating adolescents with multiple behavior problems.

Usage

data(vanderPol17)

Details

A list of data with the following structure:

- Number: Number of the effect size.
- Study: Authors of the studies.
- N: Total sample size.
- N_target: Sample size in the target group.
- N_control: Sample size in the control group.
- Comparison_condition: Either cognitive behavioral therapy (CBT), combined treatment (CT) or group therapy (Group).
- Study_ID: Level-3 cluster.
- Age_mean: Mean age of the participants.
- Follow_up: Follow-up duration (in months).
- Per_Males: Percentage of males.
- Per_Minorities: Percentage of minorities.
- Per_Conduct_disorder: Percentage of participants with conduct disorder.
- Per_Severe_cannabis_users: Percentage of participants of severe cannabis use.
- Outcome_measure: Either substance abuse, delinquency, externalizing and internalizing psychopathology, and family functioning.
- d: Effect size in Cohen's d.
- v: Sampling variance of d.

Source

Examples

data(vanderPol17)

---

VarCorr  
_Extract Variance-Covariance Matrix of the Random Effects_

Description

It extracts the variance-covariance matrix of the random effects (variance component) from either the meta or osmasem objects.

Usage

VarCorr(x, ...)

Arguments

  x  
  An object returned from either class meta or osmasem

  ...  
  Further arguments; currently none is used

Value

A variance-covariance matrix of the random effects.

Note

It is similar to coef(object, select="random") in tssem. The main difference is that coef() returns a vector while VarCorr() returns its correspondent matrix.

Author(s)

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

See Also

coeff, vcov

Examples

## Multivariate meta-analysis on the log of the odds
## The conditional sampling covariance is 0
bcg <- meta(y=cbind(ln_Odd_V, ln_Odd_NV), data=BCG,  
v=cbind(v_ln_Odd_V, cov_V_NV, v_ln_Odd_NV))  
VarCorr(bcg)
### vcov

#### Extract Covariance Matrix Parameter Estimates from Objects of Various Classes

**Description**

It extracts the variance-covariance matrix of the parameter estimates from objects of various classes.

**Usage**

```r
## S3 method for class 'tssem1FEM'
vcov(object, ...)
# S3 method for class 'tssem1FEM.cluster'
vcov(object, ...)
## S3 method for class 'tssem1REM'
vcov(object, select = c("all", "fixed", "random"), robust=FALSE, ...)
## S3 method for class 'wls'
vcov(object, ...)
## S3 method for class 'wls.cluster'
vcov(object, ...)
## S3 method for class 'meta'
vcov(object, select = c("all", "fixed", "random"), robust=FALSE, ...)
## S3 method for class 'meta3LFIML'
vcov(object, select = c("all", "fixed", "random","allX"), robust=FALSE, ...)
## S3 method for class 'reml'
vcov(object, ...)
## S3 method for class 'MxRAMModel'
vcov(object, ...)
## S3 method for class 'osmasem'
vcov(object, select=c("fixed", "all", "random"), robust=FALSE, ...)
```

**Arguments**

- `object` An object returned from objects of various classes
- `select` Select all for both fixed- and random-effects parameters, fixed for the fixed-effects parameters or random for the random-effects parameters. For meta3LFIML objects, allX is used to extract all parameters including the predictors and auxiliary variables.
- `robust` Logical. Whether to use robust standard error from `imxRobustSE`.
- `...` Further arguments; currently not in use except for `tssemRobust1`, which to be passed to `robust`.

**Value**

A variance-covariance matrix of the parameter estimates.
vec2symMat

Convert a Vector into a Symmetric Matrix

Description

It converts a vector into a symmetric matrix by filling up the elements into the lower triangle of the matrix.

Usage

vec2symMat(x, diag = TRUE, byrow = FALSE)

Arguments

x A vector of numerics or characters
diag Logical. If it is TRUE (the default), the diagonals of the created matrix are replaced by elements of x; otherwise, the diagonals of the created matrix are replaced by "1".
byrow Logical. If it is FALSE (the default), the created matrix is filled by columns; otherwise, the matrix is filled by rows.

Value

A symmetric square matrix based on column major
Author(s)
Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>

See Also
matrix2bdiag

Examples

```r
vec2symMat(1:6)
#  [,1] [,2] [,3]
# [1,] 1  2  3
# [2,] 2  4  5
# [3,] 3  5  6

vec2symMat(1:6, diag=FALSE)
# [1,] 1  1  2  3
# [2,] 1  1  4  5
# [3,] 2  4  1  6
# [4,] 3  5  6  1

vec2symMat(letters[1:6])
#  [,1] [,2] [,3]
# [1,] "a" "b" "c"
# [2,] "b" "d" "e"
# [3,] "c" "e" "f"
```

wls
Conduct a Correlation/Covariance Structure Analysis with WLS

Description
It fits a correlation or covariance structure with weighted least squares (WLS) estimation method where the inverse of the asymptotic covariance matrix is used as the weight matrix. tssem2 conducts the second stage analysis of the two-stage structural equation modeling (TSSEM). tssem2 is a wrapper of wls.

Usage

```r
wls(Cov, aCov, n, RAM=NULL, Amatrix=NULL, Smatrix=NULL, Fmatrix=NULL,
    diag.constraints=FALSE, cor.analysis=TRUE, intervals.type=c("z","LB"),
    mx.algebras=NULL, mxModel.Args=NULL, subset.variables=NULL, model.name=NULL,
    suppressWarnings=TRUE, silent=TRUE, run=TRUE, ...)

tssem2(tssem1.obj, RAM=NULL, Amatrix=NULL, Smatrix=NULL, Fmatrix=NULL,
    diag.constraints=FALSE, intervals.type=c("z", "LB"), mx.algebras=NULL,
    mxModel.Args=NULL, subset.variables=NULL, model.name=NULL, suppressWarnings=TRUE,
    silent=TRUE, run=TRUE, ...)
```
Arguments

**tssem1.obj**
An object of either class `tssem1FEM`, class `tssem1FEM.cluster` or class `tssem1REM` returned from `tssem1()`.  
**Cov**
A $p \times p$ sample correlation/covariance matrix where $p$ is the number of variables.  
**aCov**
A $p^* \times p^*$ asymptotic sampling covariance matrix of either `vechs (Cov)` or `vech (Cov)` where $p^* = p(p-1)/2$ for correlation matrix and $p^* = p(p+1)/2$ for covariance matrix.  
**n**
Sample size.  
**RAM**
A RAM object including a list of matrices of the model returned from `lavaan2RAM`.  
**Amatrix**
If RAM is not specified, an `Amatrix` is required. An asymmetric matrix in the RAM specification with `MxMatrix-class`. If it is NULL, a matrix of zero will be created. If it is a matrix, it will be converted into `MxMatrix-class` by the `as.mxMatrix` function.  
**Smatrix**
If RAM is not specified, an `Smatrix` is required. A symmetric matrix in the RAM specification with `MxMatrix-class`. If it is a matrix, it will be converted into `MxMatrix-class` by the `as.mxMatrix` function.  
**Fmatrix**
A filter matrix in the RAM specification with `MxMatrix-class`. If it is NULL (the default), an identity matrix with the same dimensions of Cov will be created. If it is a matrix, it will be converted into `MxMatrix-class` by the `as.mxMatrix` function. It is not required when there is no latent variable.  
**diag.constraints**
Logical. This argument is ignored when `cor.analysis=FALSE`. If `diag.constraints=TRUE`, the diagonals of the model implied matrix would be constrained at 1 by nonlinear constraints. The drawback is that standard error will not be generated. Parametric bootstrap is used to estimate the standard error by drawing samples from $N(vech(Cov), asyCov)$ for correlation analysis and $N(vechs(Cov), asyCov)$ for correlation analysis while asyCov is treated as fixed. This process is computationally intensive. A better approach is to request likelihood-based confidence intervals (CIs) by specifying `intervals.type="LB"`. If `diag.constraints=FALSE` and `cor.analysis=TRUE`, the diagonals are automatically constrained as ones by treating the error variances as computed values rather than as parameters. Since the error variances are not parameters, they are not reported.  
**cor.analysis**
Logical. Analysis of correlation or covariance structure. If `cor.analysis=TRUE`, `vechs` is used to vectorize $S$; otherwise, `vech` is used to vectorize $S$.  
**intervals.type**
Either z (default if missing) or LB. If it is z, it calculates the 95% Wald CIs based on the z statistic. If it is LB, it calculates the 95% likelihood-based CIs on the parameter estimates. Please note that the z values and their associated p values are based on the z statistic. They are not related to the likelihood-based CIs.  
**mx.algebras**
A list of `mxMatrix` or `mxAlgebra` objects on the `Amatrix`, `Smatrix`, and `Fmatrix`. It can be used to define new functions of parameters and their LBCIs. For example, if the regression coefficients to calculate an indirect effect are stored in $A[1,2]$ and $A[1,3]$, we may define `list(ind=mxAlgebra(Amatrix[1,2]*Amatrix[1,3], name="ind"))` See the examples in `Becker92` and `Hunter83`. It should be noted that `Fmatrix`, `Amatrix`, `Smatrix`, `Iden` ($a p \times p$ identity matrix), `sampleS` (sample
correlation or covariance matrix), impliedS1, impliedS (model implied correlation or covariance matrix), vecS, invAcov, obj, One, select and constraint and Ematrix (computed error variances when diag.constraints=FALSE) have been defined internally. You should not create new matrices using these names.

**mxModel.Args** A list of arguments passed to `mxModel`. These include, for example, additional `mxMatrix` and `mxConstraint`.

**model.name** A string for the model name in `mxModel`. If it is missing, the default is "TSSEM2 (or WLS) Analysis of Correlation Structure" for cor.analysis=TRUE and "TSSEM2 (or WLS) Analysis of Covariance Structure" for cor.analysis=FALSE.

**subset.variables** An optional character vector of variable names to select variables in the analysis. For example, there are 10 variables in Cov, say, x1 to x10. We may use c("x1", "x2", "x3") to select three variables in the analysis. Please note that this argument does not reorder the data. That is, c("x3", "x2", "x1") is the same as c("x1", "x2", "x3").

**suppressWarnings** Logical. If TRUE, warnings are suppressed. The argument to be passed to `mxRun`.

**silent** Logical. An argument to be passed to `mxRun`

**run** Logical. If FALSE, only return the mx model without running the analysis.

**...** Further arguments to be passed to `mxRun`.

**Value**

An object of class `wls` with a list of

- `call` The matched call
- `Cov` Input data of either a covariance or correlation matrix
- `asyCov` The asymptotic covariance matrix of the input data
- `nObservedStat` Number of observed statistics
- `n` Sample size
- `cor.analysis` logical
- `nConstraints` Number of constraints imposed on S
- `indepModelChisq` Chi-square statistic of the independent model returned by .indepwlsChisq
- `indepModelDf` Degrees of freedom of the independent model returned by .indepwlsChisq
- `mx.fit` A fitted object returned from `mxRun`

**Note**

If the input is a list of `tssem1.obj`, it returns a list of results for each cluster.

**Author(s)**

Mike W.-L. Cheung <mikewlcheung@nus.edu.sg>
References


See Also
tsseml, Becker92, Digman97, Hunter83, issp89, issp05

Examples

```r
## Not run:
#### Analysis of correlation structure
R1.labels <- c("a1", "a2", "a3", "a4")

R1 <- matrix(c(1.00, 0.22, 0.24, 0.18,
               0.22, 1.00, 0.30, 0.22,
               0.24, 0.30, 1.00, 0.24,
               0.18, 0.22, 0.24, 1.00), ncol=4, nrow=4,
dimnames=list(R1.labels, R1.labels))

n <- 1000
acovR1 <- asyCov(R1, n)

#### One-factor CFA model using lavaan specification
model1 <- "f =~ a1 + a2 + a3 + a4"

RAM1 <- lavaan2RAM(model1, obs.variables=R1.labels)

wls.fit1a <- wls(Cov=R1, aCov=acovR1, n=n, RAM=RAM1,
cor.analysis=TRUE, intervals="LB")
summary(wls.fit1a)

#### One-factor CFA model using RAM specification
(A1 <- cbind(matrix(0, nrow=5, ncol=4),
matrix(c("0.2*a1","0.2*a2","0.2*a3","0.2*a4","0"),
ncol=1)))
```
(S1 <- Diag(c("0.2*e1","0.2*e2","0.2*e3","0.2*e4",1)))

## The first 4 variables are observed while the last one is latent.
(F1 <- create.Fmatrix(c(1,1,1,1,0), name="F1"))

wls.fit1b <- wls(Cov=R1, aCov=acovR1, n=n, Fmatrix=F1, Smatrix=S1, Amatrix=A1, cor.analysis=TRUE, intervals="LB")
summary(wls.fit1b)

## Select 3 variables to analyze
model2 <- "f =~ a1 + a2 + a3"

RAM2 <- lavaan2RAM(model2, obs.variables=R1.labels[-4])

wls.fit1c <- wls(Cov=R1, aCov=acovR1, n=n, RAM=RAM2, cor.analysis=TRUE, subset.variables=c("a1", "a2", "a3"))
summary(wls.fit1c)

#### Multiple regression analysis using lavaan specification

R2.labels <- c("y", "x1", "x2")
R2 <- matrix(c(1.00, 0.22, 0.24, 
               0.22, 1.00, 0.30, 
               0.24, 0.30, 1.00), ncol=3, nrow=3, 
              dimnames=list(R2.labels, R2.labels))
acovR2 <- asyCov(R2, n)

model3 <- "y ~ x1 + x2"
## Variances of x1 and x2 are 1
x1 ~~ 1*x1
x2 ~~ 1*x2
## x1 and x2 are correlated
x1 ~~ x2

RAM3 <- lavaan2RAM(model3, obs.variables=R2.labels)

wls.fit2a <- wls(Cov=R2, aCov=acovR2, n=n, RAM=RAM3, cor.analysis=TRUE, intervals="z")
summary(wls.fit2a)

#### Multiple regression analysis using RAM specification

## A2: Regression coefficients
# y x1 x2
# y T F F
# x1 F F F
# x2 F F F
(A2 <- mxMatrix("Full", ncol=3, nrow=3, byrow=TRUE, free=c(FALSE, rep(TRUE, 2), rep(FALSE, 6)), name="A2"))

## S2: Covariance matrix of free parameters
Forty-four Studies from Cheung (2013)
Description

Between 1990 and 1993, 57,561 adults aged 18 and above from 42 nations were interviewed by local academic institutes in Eastern European nations and by professional survey organizations in other nations. The standardized mean difference (SMD) between males and females on life satisfaction and life control in each country were calculated as the effect sizes. Positive values indicate that males have higher scores than females do.

Usage

data(wvs94a)

Details

The variables are:

- country Country
- lifesat SMD on life satisfaction
- lifecon SMD on life control
- lifesat_var Sampling variance of lifesat
- inter_cov Sampling covariance between lifesat and lifecon
- lifecon_var Sampling variance of lifecon
- gnp Gross National Product

Source


References


Examples

```r
## Not run:
data(wvs94a)

## Random-effects model
random.ma1 <- meta(y=cbind(lifesat, lifecon),
                   v=cbind(lifesat_var, inter_cov, lifecon_var), data=wvs94a,
                   model.name="Random effects model")
summary(random.ma1)

## Random-effects model with both population effect sizes fixed at 0
random.ma2 <- meta(y=cbind(lifesat, lifecon),
```
v=cbind(lifesat_var, inter_cov, lifecon_var), data=wvs94a,
intercept.constraints=matrix(0, nrow=1, ncol=2),
model.name="Effect sizes are fixed at 0")

summary(random.ma2)

## Compare the nested models
anova(random.ma1, random.ma2)

## Fixed-effects model by fixing the variance component at 0
fixed.ma <- meta(y=cbind(lifesat, lifecon),
 v=cbind(lifesat_var, inter_cov, lifecon_var), data=wvs94a,
 RE.constraints=matrix(0, ncol=2, nrow=2),
 model.name="Fixed effects model")

summary(fixed.ma)

## Mixed-effects model
## gnp is divided by 10000 and centered by using
## scale(gnp/10000, scale=FALSE)
mixed.ma1 <- meta(y=cbind(lifesat, lifecon),
 v=cbind(lifesat_var, inter_cov, lifecon_var),
 x=scale(gnp/10000, scale=FALSE), data=wvs94a,
 model.name="GNP as a predictor")

summary(mixed.ma1)

## Mixed-effects model with equal regression coefficients
mixed.ma2 <- meta(y=cbind(lifesat, lifecon),
 v=cbind(lifesat_var, inter_cov, lifecon_var),
 x=scale(gnp/10000, scale=FALSE), data=wvs94a,
 coef.constraints=matrix(c("0.0*Eq_slope",
                           "0.0*Eq_slope"), nrow=2),
 model.name="GNP as a predictor with equal slope")

summary(mixed.ma2)

## Compare the nested models
anova(mixed.ma1, mixed.ma2)

## Plot the multivariate effect sizes
plot(random.ma1, main="Estimated effect sizes and their 95% confidence ellipses",
     axis.label=c("Gender difference on life satisfaction",
                  "Gender difference on life control"))

## End(Not run)

Forty-four Covariance Matrices on Life Satisfaction, Job Satisfaction, and Job Autonomy

Description

Between 1990 and 1993, 57,561 adults aged 18 and above from 42 nations were interviewed by local academic institutes in Eastern European nations and by professional survey organizations in other
The covariance matrices among Life Satisfaction, Job Satisfaction, and Job Autonomy were calculated.

Usage

```r
data(wvs94b)
```

Details

The variables are:

- **data**: Covariance matrix among Life Satisfaction (LS), Job Satisfaction (JS), and Job Autonomy (JA)
- **n**: Sample size in the country

Source


References


Examples

```r
## Not run:
data(wvs94b)

## Get the indirect and the direct effects and
## their sampling covariance matrices for each study
indirect1 <- indirectEffect(wvs94b$data, wvs94b$n)
indirect1

## Multivariate meta-analysis on the indirect and direct effects
indirect2 <- meta(indirect1[, c("ind_eff", "dir_eff")],
                  indirect1[, c("ind_var", "ind_dir_cov", "dir_var")])

summary(indirect2)

## End(Not run)
```
Index

* **bootuniR**
  bootuniR1, 30
  bootuniR2, 31
* **compute effect sizes**
  indirectEffect, 72
* **datasets**
  Aloe14, 5
  BCG, 15
  Becker90, 18
  Becker83, 22
  Becker92, 23
  Becker94, 25
  Berkey98, 27
  Boer16, 29
  Bornmann07, 32
  Cheung00, 37
  Cheung09, 39
  Cooke16, 42
  Cooper03, 44
  Digman97, 57
  Gleser94, 60
  Gnambs18, 61
  HedgesOlkin85, 62
  Hox02, 65
  Hunter83, 66
  issp05, 74
  issp89, 76
  Jaramillo05, 78
  Kalaian96, 79
  Mak09, 83
  Mathieu15, 84
  Nam03, 95
  Nohe15, 96
  Norton13, 99
  Roorda11, 119
  Scalco17, 120
  Stadler15, 125
  Tenenbaum02, 128
  vanderPol117, 135
  wvs94a, 144
  wvs94b, 146
* **meta-analysis**
  asyCov, 13
  calEffSizes, 34
  homoStat, 63
  meta, 88
  meta3L, 92
  reml, 113
  reml3L, 116
  smdMES, 121
  smdMTS, 123
* **meta**
  rerun, 118
* **methods**
  anova, 7
  coef, 40
  lavaan2RAM, 80
  meta2semPlot, 91
  plot, 105
  print, 109
  summary, 126
  VarCorr, 136
  vcov, 137
* **osmasem3L**
  create.Tau2, 52
  osmasemR2, 102
  osmasemSRMR, 103
  rerun, 118
* **osmasem**
  Cor2DataFrame, 45
  create.Tau2, 52
  create.V, 53
  create.vechsR, 54
  osmasem, 100
  osmasemR2, 102
  osmasemSRMR, 103
  rerun, 118
* **package**
INDEX

metaSEM-package, 4

* **tssem**
  lavaan2RAM, 80
  meta2semPlot, 91
  rerun, 118
  tssem1, 129
  tssemParaVar, 131
  wls, 139

* **uniR**
  uniR1, 132
  uniR2, 133

* **utilities**
  as.mxAlgebra, 8
  as.mxMatrix, 10
  as.symMatrix, 12
  bdiagMat, 16
  bdiagRep, 17
  checkRAM, 36
  create.Fmatrix, 47
  create.modMatrix, 48
  create.mxMatix, 49
  create.mxModel, 50
  Diag, 56
  impliedR, 69
  is.pd, 73
  list2matrix, 82
  matrix2bdiag, 86
  pattern.n, 103
  pattern.na, 104
  rCor, 110
  readData, 111
  vec2symMat, 138

* **wls**
  rerun, 118

Aloe14, 5
anova, 7
as.mxAlgebra, 8
as.mxMatrix, 9, 10, 12, 36, 48, 81, 88, 89, 93, 114, 130
as.symMatrix, 11, 12
asyCov, 13, 46, 73, 130
asyCov0d (asyCov), 13

BCG, 15
bdiagMat, 16, 18
bdiagRep, 17, 17
Becker09, 18, 81, 91, 133, 134
Becker83, 22

Becker92, 23, 81, 91, 131, 140, 142
Becker94, 25
Berkey98, 27, 90, 108, 115
Boer16, 29
bootuniR1, 30, 132
bootuniR2, 31, 31, 32, 128, 132
Bornmann07, 32, 94, 118
calEffSizes, 34, 122, 124
checkRAM, 11, 36, 81
Cheung00, 37
Cheung09, 39, 131
coef, 40, 136
Cooke16, 42
Cooper03, 44, 94, 118
Cor2DataFrame, 45, 102, 104
create.Fmatrix, 11, 47, 50
create.modMatrix, 48
create.mxMatrix, 11, 48, 49, 50
create.mxModel, 50
create.Tau2, 46, 52, 54, 55, 102
create.V, 46, 53, 55, 102
create.vechsR, 46, 53, 54, 54, 102
Diag, 56
diag, 57
diag<- (Diag), 56
Digman97, 57, 81, 91, 131, 142
Gleser94, 60, 122, 124
Gnamsb18, 61
HedgesOlkin85, 62
homoStat, 63
Hox02, 65, 90, 115
Hunter83, 66, 81, 91, 140, 142

impliedR, 69
imxRobustSE, 127, 137
indirectEffect, 72
is.pd, 73
issp05, 74, 76, 131, 142
issp89, 75, 76, 131, 142
Jaramillo05, 78
Kalaian96, 79

lavaan2RAM, 11, 36, 48, 51, 69, 80, 101, 133, 134, 140
index
INDEX

tssemParaVar, 32, 131

uniR1, 132, 133, 134
uniR2, 133, 133
uniR2lavaan (uniR2), 133
uniR2mx (uniR2), 133

vanderPo117, 135
VarCorr, 136
vcov, 136, 137
vec2symMat, 87, 138
vech, 64, 82, 88, 114, 140
vechs, 82, 140

wls, 7, 41, 48, 109, 128, 131, 138, 139
wvs94a, 90, 108, 144
wvs94b, 146