Package: symSEM (via r-universe)

October 14, 2024

Type Package

Title Symbolic Computation for Structural Equation Models

Version 0.4

Date 2024-05-15

Depends caracas

Imports OpenMx, metaSEM

Suggests testthat, roxygen2

Maintainer Mike Cheung <mikewlcheung@nus.edu.sg>

Description A collection of functions for symbolic computation using the 'caracas' package for structural equation models and other statistical analyses. Among its features is the ability to calculate the model-implied covariance (and correlation) matrix and the sampling covariance matrix of variable functions using the delta method.

License GPL (>=2)

Encoding UTF-8

SystemRequirements Python (>= 3.6.0)

LazyLoad yes

LazyData yes

ByteCompile yes

URL https://github.com/mikewlcheung/symsem

RoxygenNote 7.3.1

Repository https://mikewlcheung.r-universe.dev

RemoteUrl https://github.com/mikewlcheung/symsem

RemoteRef HEAD

RemoteSha 1dbfa712fd4f53daa95c0dac7c1ce72c93acd048

Contents

`symSEM-package` *A collection of functions for symbolic computation using 'caracas' package for structural equation models and other statistical analyses. Among its features is the ability to calculate the model-implied covariance (and correlation) matrix and the sampling covariance matrix of variable functions using the delta method.*

Description

A collection of functions for symbolic computation using 'caracas' package for structural equation models and other statistical analyses. Among its features is the ability to calculate the model-implied covariance (and correlation) matrix and the sampling covariance matrix of variable functions using the delta method.

Note

As 'caracas' uses 'SymPy" in the backend. Reserved words in SymP, such as"lambda" and "I" are converted to some random strings first. These random strings are converted back to R.

`deltamethod` *Compute the Variance-Covariance Matrix of Functions using the first-order Delta Method*

Description

It computes the variance-covariance matrix of functions using the first-order delta method.

Usage

```
deltamethod(fn, Covvars, vars, Var.name = "V", Cov.name = "C", simplify = TRUE)
```

Arguments

fn	A function in character strings or a vector of functions.
Covvars	Variance-covariance matrix of the variables. Users must ensure the order of variables is the same as that in vars; Otherwise, the results are likely incorrect. If it is not specified, they are automatically generated.
vars	A vector of characters of the random variables. If the random variables are not listed in 'vars', they are treated as constants. If 'vars' is missing, all names in 'RAM' are treated as random variables.
Var.name	Name of the variances.
Cov.name	Name of the covariances.
simplify	Attempt to simplify the output.

Value

Variance-covariance matrix of the functions.

Author(s)

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

Examples

```
## Not run:

#### Fisher-z-transformation
fn  <- "0.5*log((1+r)/(1-r))"

## Sampling variance of r
Covvars <- "(1-r^2)^2/n"

deltamethod(fn=fn, Covvars=Covvars, vars="r")
## $fn
##     [,1]
## fn1 "0.5*log((r+1)/(1-r))"

## $Covfn
##     fn1
## fn1 "1/n"

## $vars
## [1] "r"

## $Covvars
##   r
## r "(1-r^2)^2/n"

## $Jmatrix
##     r
## fn1 "(0.5*(1-r+r+1)*(1-r))/((1-r)^2*(r+1))"
```

```
#### Raw mean difference: y_treatment - y_control
fn <- "yt - yc"

## Sampling covariance matrix
## S2p: pooled variance
## nt: n_treatment
## nc: n_control
Covvars <- matrix(c("S2p/nt", 0,
                    0, "S2p/nc"),
                  ncol=2, nrow=2)

deltamethod(fn=fn, Covvars=Covvars, vars=c("yt", "yc"))
## $fn
##     [,1]
## fn1 "yt-yc"

## $Covfn
##     fn1
## fn1 "(S2p*nt+S2p*nc)/(nt*nc)"

## $vars
## [1] "yt" "yc"

## $Covvars
##    yt       yc
## yt "S2p/nt" "0"
## yc "0"      "S2p/nc"

## $Jmatrix
##     yt  yc
## fn1 "1" "-1"

#### log(odds)
fn <- "log(p/(1-p))"

## Sampling variance of p
Covvars <- "p*(1-p)/n"

## Though it is correct, the simplification does not work well.
deltamethod(fn=fn, Covvars=Covvars, vars="p")
## $fn
##     [,1]
## fn1 "log(p/(1-p))"

## $Covfn
##     fn1
## fn1 "(3*p^2-p^3-3*p+1)/((p^4-4*p^3+6*p^2-4*p+1)*p*n)"

## $vars
## [1] "p"

## $Covvars
```

```
##   p
## p "(p*(1-p))/n"

## $Jmatrix
##     p
## fn1 "((1-p+p)*(1-p))/((1-p)^2*p)"

## End(Not run)
```

`impliedS` *Compute a Symbolic Model-Implied Covariance/Correlation Matrix*

Description

It computes a symbolic model-implied covariance (or correlation) matrix in SEM using the RAM specification inputs.

Usage

```
impliedS(RAM, corr = FALSE, replace.constraints = FALSE, convert = TRUE)
```

Arguments

`RAM`	A RAM object including a list of matrices of the model returned from `lavaan2RAM`
`corr`	Whether the model implied matrix is covariance (default) or correlation structure.
`replace.constraints`	Whether to replace the parameters with the constraints in the `mxalgebras` slot. Suppose the formula is `para1==para2+para3`, `para1` will be replaced by `para2+para3` if this argument is `TRUE`.
`convert`	Whether to convert random strings back to parameters. For internal use only. Users unlikely need to use this argument.

Value

A list of object with class `implieS`. It stores the A, S, and F matrices and the model implied covariance (or correlation) matrix and the vector of the means.

Author(s)

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

Examples

```
## Not run:

#### A mediation model
model1 <- "y ~ c*x + b*m
           m ~ a*x
           ## Means
           y ~ b0*1
           m ~ m0*1
           x ~ x0*1"

RAM1 <- metaSEM::lavaan2RAM(model1)

## Model-implied covariance matrix and mean structure
impliedS(RAM1, corr=FALSE)

## Model-implied correlation matrix
impliedS(RAM1, corr=TRUE)

#### A CFA model
model2 <- "f =~ x1 + x2 + x3 + x4
           ## Mean
           f ~ fmean*1"

RAM2 <- metaSEM::lavaan2RAM(model2)

## Model-implied covariance matrix
impliedS(RAM2, corr=FALSE)

## Model-implied correlation matrix
impliedS(RAM2, corr=TRUE)

## End(Not run)
```

`JacobianRAM` *Compute a Jacobian Matrix of the Implied Covariance/Correlation Matrix based on a RAM model.*

Description

It computes a symbolic Jacobian matrix of the model-implied covariance (or correlation) matrix in SEM using the RAM specification.

Usage

```
JacobianRAM(RAM, vars, corr = FALSE)
```

Arguments

RAM	A RAM object including a list of matrices of the model returned from lavaan2RAM
vars	A vector of characters of the random variables. If the random variables are not listed in 'vars', they are treated as constants. If 'vars' is missing, all names in 'RAM' are treated as random variables.
corr	Whether the model implied matrix is covariance (default) or correlation structure.

Value

A Jacobian matrix.

Author(s)

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

Examples

```
## Not run:

#### A mediation model
model1 <- "y ~ c*x + b*m
           m ~ a*x
           ## Means
           y ~ b0*1
           m ~ m0*1
           x ~ x0*1"

RAM1 <- metaSEM::lavaan2RAM(model1)

## Model-implied covariance matrix and mean structure
JacobianRAM(RAM1, corr=FALSE)

## Model-implied correlation matrix
JacobianRAM(RAM1, corr=TRUE)

#### A CFA model
model2 <- "f =~ x1 + x2 + x3 + x4#'
           ## Mean
           f ~ fmean*1"

RAM2 <- metaSEM::lavaan2RAM(model2)

## Model-implied covariance matrix
JacobianRAM(RAM2, corr=FALSE)

## Model-implied correlation matrix
JacobianRAM(RAM2, corr=TRUE)

## End(Not run)
```

Index