Package ‘stremo’

February 20, 2015

Type Package
Title Functions to help the process of learning structural equation modelling
Version 0.2
Date 2011-09-05
Author Gustavo Carvalho, Marco Batalha, and Owen Petchey.
Maintainer Gustavo Carvalho <gustavo.bio@gmail.com>
Description Functions to assist the process of learning structural equation modeling
Depends lavaan (>= 0.4-9), numDeriv, MASS
License GPL
LazyLoad yes
Repository CRAN
Date/Publication 2011-09-29 11:28:21
NeedsCompilation no

R topics documented:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>stremo-package</td>
<td>2</td>
</tr>
<tr>
<td>albert</td>
<td>3</td>
</tr>
<tr>
<td>boot.lavaan</td>
<td>4</td>
</tr>
<tr>
<td>cor2cov</td>
<td>5</td>
</tr>
<tr>
<td>degrees of freedom</td>
<td>6</td>
</tr>
<tr>
<td>endogenous</td>
<td>7</td>
</tr>
<tr>
<td>exogenous</td>
<td>7</td>
</tr>
<tr>
<td>Factor analysis</td>
<td>8</td>
</tr>
<tr>
<td>fit.nlminb</td>
<td>8</td>
</tr>
<tr>
<td>GLS</td>
<td>10</td>
</tr>
<tr>
<td>grace</td>
<td>10</td>
</tr>
<tr>
<td>harnik</td>
<td>11</td>
</tr>
<tr>
<td>II</td>
<td>13</td>
</tr>
<tr>
<td>is.hermitian</td>
<td>13</td>
</tr>
</tbody>
</table>
stremo-package

Learning structural equation modeling.

Index

stremo-package  Learning structural equation modeling.

Description

Functions and toy data to play around whilst learning structural equation modeling. xx

Details

Package: stremo
Type: Package
Version: 1.0
Date: 2011-09-05
License: GPL
LazyLoad: yes
**Author(s)**
Gustavo Carvalho, Marco Batalha, and Owen Petchey.
Maintainer: Gustavo Carvalho <gustavo.bio@gmail.com>

**References**

**Examples**
```r
data(albert)
cat(albert.model)
albert.litho.cov
fit <- fit.nlminb(albert.model, albert.litho.cov, 107)
```

**Description**
Determinants of seed production in plants of *Erodium paularense* in rock and in lithosol.

**Usage**
```r
data(albert)
albert.litho.cov
albert.rock.cov
albert.model
```

**Format**
Two covariance matrices with the number of observations as an attribute and a character vector with the model specification.

**Note**
Models used are in page 1737. Correlation matrices are in Appendix A. 107 observations.

**Source**
Article:
[http://www.esajournals.org/doi/abs/10.1890/0012-9658%282001%29082%5B1734%3AFRSONE%5D2.0.CO%3B2](http://www.esajournals.org/doi/abs/10.1890/0012-9658%282001%29082%5B1734%3AFRSONE%5D2.0.CO%3B2)
Appendix:
References


Examples

```r
## Not run:
data(albert)
cat(albert.model)
albert.rock.fit <- sem(albert.model, sample.cov = albert.rock.cov,
                        sample.nobs = 107)
albert.litho.fit <- sem(albert.model, sample.cov = albert.litho.cov,
                        sample.nobs = 107)
summary(albert.rock.fit, standardized = TRUE)
summary(albert.litho.fit, standardized = TRUE)

## End(Not run)
```

---

**boot.lavaan**  
Nonparametrical bootstrapping of a SEM model fit by lavaan.

**Description**

This function does very crude nonparametrical bootstrapping of a fitted sem model using lavaan.

**Usage**

```r
boot.lavaan(fitted.model, n)
```

**Arguments**

- `fitted.model` A model fitted by sem or cfa.
- `n` Number of replications.

**Details**

Steps:  
1. Fit a model normally using the arguments `sample.cov` and `sample.nobs` instead of `data`;  
2. Get the sigma hat from the fitted model and build an empirical dataset with `sample.nobs` observations;  
3. Get `sample.nobs` observations from this new dataset with replacement;  
4. Fit a new model using the sample taken from the simulated dataset;  
5. Repeat 3 and 4 `n` times.
**Value**

- `x2` Bootstrapped maximum likelihood chi-squared
- `p.x2` P value of the bootstrapped maximum likelihood chi-squared
- `est` Estimated coefficients
- `se` Bootstrapped standard errors
- `z` Bootstrapped z values
- `p.est` P values of the bootstrapped coefficients
- `coefs` Matrix with the values of all coefficients in all runs

**Examples**

```r
## Not run:
data(albert)
fit <- sem(albert.model, sample.cov = albert.litho.cov, sample.nobs = 107)
booted.fit <- boot.lavaan(fit, 1000)
## End(Not run)
```

---

**cor2cov**  
* Covariance matrix from a correlation matrix.

## Description

Build a covariance matrix from a correlation matrix and variables standard deviations.

## Usage

```r
cor2cov(cormat, sds)
```

## Arguments

- `cormat` Correlation matrix to be transformed.
- `sds` Vector of standard deviations.

## Value

Covariance matrix.

## See Also

`cov2cor`, `mirror.tri`
Examples

covmat <- ability$ cov
sds <- sqrt(diag(covmat))
cormat <- cov2cor(covmat)

covmat.trans <- cor2cov(cormat, sds)
round(covmat - covmat.trans, 10)

degrees of freedom
Number of degrees of freedom.

Description
This function returns the number of degrees of freedom of a lavaanified model.

Usage

model.df(model)

Arguments

model A lavaanified model.

Value

free Number of free parameters
fixed Number of fixed variables
total.df Amount of information available
model.df Number of degrees of freedom

Examples

data(albert)
model <- lvnfy(albert.model, data = albert.litho$ cov, n = 107)
model.df(model)
endogenous

Display the endogenous variables of a lavaanified model

Description
This function returns a character vector containing the names of the endogenous variables in a given model.

Usage
endogenous(model)

Arguments
model A lavaanified model.

See Also
exogenous, manifests, observed, and latents.

Examples
data(albert)
model <- lvnfy(albert.model, data = albert.litho.cov, n = 107)
endogenous(model)

exogenous

Display the exogenous variables of a lavaanified model

Description
This function returns a character vector containing the names of the exogenous variables in a given model.

Usage
exogenous(model)

Arguments
model A lavaanified model.

See Also
exogenous, manifests, observed, and latents.
Examples

```r
data(albert)
model <- lvmfy(albert.model, data = albert.litho.cov, n = 107)
exogenous(model)
```

Factor analysis

Factor analysis using principal components as the extraction method.

Usage

```r
factanal.prcomp(covmat, nfactors = NULL)
```

Arguments

- `covmat`: A covariance matrix.
- `nfactors`: Number of factors.

See Also

`princomp` and `rmnorm`.

Examples

```r
data(albert)
factanal.prcomp(albert.litho.cov, NROW(albert.litho.cov))
```

fit.nlminb

Fit a structural equation model using `nlminb`.

Description

This function tries to find the best set of parameter estimates of a structural equation model. It is very slow compared to `sem` in `lavaan` and does not handle equality constraints, multiple groups and intercepts. We made this to aid those who are learning SEM to understand the very basics of the technique. Please refer to `lavaan` for more serious usage.

Usage

```r
fit.nlminb(model, sigma, n, fun = "fml")
```
**Arguments**

- **model**: A character vector containing a model specification.
- **sigma**: Population covariance matrix.
- **n**: Number of observations in the dataset used to build sigma.
- **fun**: Fitting function to be minimised by the optimiser. Should return a scalar.

**Details**

This function is coded to be as expressive as possible and speed has not been our aim here. Users should be able to understand at least some of what is going on internally by taking a peak at the code.

The first thing this function does is to call lavaan to get a good set of starting values for the parameters to be estimated and use lavaan’s internal representation of a SEM model as a starting point. After that, the chosen function calls an optimiser, `nlminb`, which will try to converge to a solution given a set of parameter estimates and a population covariance matrix. Then, it will try to calculate standard errors from the Hessian matrix, z-values for each estimate and its correspondent p-values. The next step is to calculate the fit of the model to, finally, build a response.

**Value**

- **model**: The internal representation of the model
- **est**: The best set of parameter estimates
- **se**: Standard errors
- **z**: Z-values
- **p**: P-values
- **chisquared**: The maximum likelihood chi-squared
- **df**: The number of degrees of freedom
- **n**: The number of observations
- **pval**: The significance of the chi-squared statistics
- **estimator**: The fitting function used
- **sigma.hat**: The model-implied covariance matrix
- **sigma**: The population covariance matrix

**See Also**

`sem` in `lavaan` is much faster and converges to a unique solution more frequently. It can also handle a much more diverse range of models, including those with multiple groups, equality constraints, growth models and so on.
Examples

```r
## Not run:
data(albert)
cat(albert.model)
fit <- fit.nlminb(albert.model, albert.litho.cov, 107)
summary(fit)

## End(Not run)
```

---

GLS  

*Generalised least squares.*

---

Description

GLS fitting function.

Usage

```r
fgls(sigma, sigma.hat)
```

Arguments

- `sigma`: Population covariance matrix.
- `sigma.hat`: Model-implied covariance matrix.

See Also

`fml`

---

**grace**  

*Data and tentative model syntax for the paper by Grace et al. 2010.*

---

Description

Grace et al. 2010 describe the use of structural equation meta-models to aid the translation of theory into structural equation models. They use an example where they relate plant diversity and community biomass to variations in environmental stress and disturbance.

Usage

```r
data(grace)
grace.cov
grace.model
```
harnik

Format
A covariance matrix and a character vector describing the model syntax.

Note
Our model syntax does not converge in lavaan or stremo, but we have included it anyway for learning purposes. The model we tried to fit is on page 81, figure 7. The covariance matrix is on page 76, table 2. There were 190 observations.

Source
http://www.esajournals.org/doi/full/10.1890/09-0464.1

References

Examples
```r
## Not run:
data(grace)
cat(grace.model)
grace.fit <- sem(grace.model, sample.cov =
    grace.cov,
    sample.nobs = 190)

## End(Not run)
```

harnik    Effects of biological factors on extinction risk in fossil bivalves.

Description
Raw data and model syntaxes for the paper by Harnik (2011) on the direct and indirect effects of biological factors on extinction risk in fossil bivalves.

Usage
data(harnik)
harnik.model.A
harnik.model.B
harnik.model.B

Format
A data frame with 108 observations on 8 variables and three character vectors containing model syntaxes.
Note

The models are in page 2 (figure 1). All models converge. The coefficients in model A come from three independent linear regressions. The dataset is included as it was made available in the supporting information. Data transformations should be done following the manuscript in order to get the same results. Model C involves groups (superfamilies).

Data transformation:
1. Abundance and range extent were transformed with the square root of the arcsine method.
2. Abundance and body size were log-transformed.
3. Abundance, range extent and body size were scaled and centered (we are replicating what was reported in the manuscript, but maximum likelihood estimation expects unstandardized data).

Source

Article:
http://www.pnas.org/cgi/doi/10.1073/pnas.1100572108

Supplementary information:
http://www.pnas.org/content/suppl/2011/07/25/1100572108.DCSupplemental

References


Examples

```r
## Not run:
data(harnik)
harnik.t <- harnik

harnik.t$abundance <- asin(sqrt(harnik.t$abundance))
harnik.t$range.extent <- asin(sqrt(harnik.t$range.extent))

harnik.t$abundance <- log(harnik.t$abundance)
harnik.t$body.size <- log(harnik.t$body.size)

harnik.t$abundance <- scale(harnik.t$abundance, center = TRUE, scale = TRUE)
harnik.t$range.extent <- scale(harnik.t$range.extent, center = TRUE, scale = TRUE)
harnik.t$body.size <- scale(harnik.t$body.size, center = TRUE, scale = TRUE)

harnik.model.A.1.fit <- sem(harnik.model.A[[1]], data = harnik.t)
harnik.model.A.2.fit <- sem(harnik.model.A[[2]], data = harnik.t)
harnik.model.A.3.fit <- sem(harnik.model.A[[3]], data = harnik.t)
harnik.model.B.fit <- sem(harnik.model.B, data = harnik.t)
harnik.model.C.fit <- sem(harnik.model.C, data = harnik.t, group = "superfamily",
group.equal = "regressions",
group.partial = "duration ~ body.size")
```
Identity matrix

Description
Identity matrix of a lavaanified model with same dimensions as BETA(model).

Usage
II(model)

Arguments
model A lavaanified model.

Value
An identity matrix.

Examples
data(albert)
tmp <- lvnfy(albert.model, data = albert.litho.cov, n = 107)
II(tmp)

is.hermitian    Check if a matrix is Hermitian.

Description
This function checks if a matrix is Hermitian, that is, if its below and above diagonal elements are equal.

Usage
is.hermitian(covmat)

Arguments
covmat A square matrix.
is.pd

Value
TRUE or FALSE.

See Also
is.pd to check if a matrix is positive-definite and is.within.bounds to check whether all covariances in a covariance matrix are within boundaries.

Examples

data(stiles)
st.cov <- cov(stiles[-1])
is.hermitian(st.cov) # TRUE
st.cov[5,3] <- 10
is.hermitian(st.cov) # FALSE

is.pd

Do some basic checking to see if a matrix is positive-definite.

Description
A covariance matrix used in structural equation modeling should be positive-definite, as it is going to be inverted during estimation and fit. This function checks whether a Hermitian matrix is positive-definite.

Usage
is.pd(covmat)

Arguments
covmat A Hermitian matrix.

Details
This function checks, in this order:
1. Matrix is Hermitian
2. All covariances are within bounds
3. See if matrix is invertible using solve
4. Check if all eigenvalues are positive
5. Check whether the determinant is positive

Value
TRUE or FALSE.
See Also

`is.within.bounds` and `is.hermitian`.

Examples

```r
data(latta)
is.pd(latta.greenhouse.cov)
```

is.within.bounds A covariance matrix to be tested. Should be Hermitian.

Value

TRUE or FALSE. If FALSE, some information about which covariances are out of bounds is printed.

See Also

`is.pd` and `is.hermitian`.

Examples

```r
data(latta)
is.within.bounds(latta.greenhouse.cov)
```
iterator

Minimization of a fitting function.

Description
Function passed to an optimizer to minimize a fitting function.

Usage
iterator(pars, model, sigma, matrep, fun = "fml")

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pars</td>
<td>Estimates of the free parameters.</td>
</tr>
<tr>
<td>model</td>
<td>A lavaanified model.</td>
</tr>
<tr>
<td>sigma</td>
<td>Population covariance matrix.</td>
</tr>
<tr>
<td>matrep</td>
<td>Model skeleton, as given by <code>matrix.representation</code>.</td>
</tr>
<tr>
<td>fun</td>
<td>Fitting function to be minimized.</td>
</tr>
</tbody>
</table>

Details
This function is used inside `fit.nlminb` and passed as an argument to `nlminb`.

Value
Value of the statistics to be minimized given a set of parameter estimates.

See Also
`fit.nlminb`, `nlminb`, `fml`, and `fgls`.

jumpstart

Get the starting values of the free parameters of a model.

Description
Get the starting values of the free parameters of a model.

Usage
jumpstart(model)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>model</td>
<td>A lavaanified model.</td>
</tr>
</tbody>
</table>
Value

A vector of free parameters.

Description

Covariance matrices and syntax of the path models presented in the paper by Lamb et al. 2010.

Usage

```r
data(lamb)
lamb.bacterial.cov
lamb.archaenal.cov
lamb.bacterial.model
lamb.archaenal.model
```

Format

Two covariance matrices and two character vectors with model specifications.

Note

Both syntax vectors are for the models in page 490, figure 3. Covariance matrices were built from the raw data made available in the electronic supplementary material.

Source

Article:

http://www.springerlink.com/content/f664v272642r4347/

Supplementary information:

http://www.springerlink.com/content/f664v272642r4347/supplementals/

References

Examples

```r
## Not run:
data(lamb)
cat(lamb.bacterial.model)
cat(lamb.archaeal.model)
lamb.bacterial.model.fit <- sem(lamb.bacterial.model, sample.cov =
                          lamb.bacterial.cov, sample.nobs = 129)
lamb.archaeal.model.fit <- sem(lamb.archaeal.model, sample.cov =
                          lamb.archaeal.cov, sample.nobs = 129)
summary(lamb.bacterial.model.fit, standardize = TRUE)
summary(lamb.archaeal.model.fit, standardize = TRUE)
## End(Not run)
```

---

**lamb2**  
*Yield components in oat.*

---

**Description**

Covariance matrix and syntaxes for the three first models in the paper by Lamb et al. (2011).

**Usage**

```r
data(lamb2)
lamb2
lamb2.model.fig1
lamb2.model.fig2
lamb2.model.fig3
```

**Format**

A covariance matrix and three character vectors with model specifications.

**Note**

We couldn’t make the latent variable models converge, so we included only the path models.

**Source**

Article:


**References**

### latents

Display the latent variables of a lavaanified model

#### Description

This function returns a character vector containing the names of the latent variables in a given model.

#### Usage

```r
latents(model)
```

#### Arguments

- `model` A lavaanified model.

#### Examples

```r
data(pugesek)
model <- lvnfy(pugesek.onefactor.model, data = bumpus.log)
latents(model)
```

### latta

Path analysis of natural selection via survival and fecundity across contrasting environments in Avena barbata.

#### Description

Covariance matrices and syntax of the models in the paper by Latta & McCain 2009.
Usage

```r
data(latta)
latta.greenhouse.cov
latta.hopland.cov
latta.sierra.cov
latta.greenhouse.model
latta.field.model
```

Format

Three covariance matrices and two character vectors containing data and model specification.

Note

The syntax in `latta.greenhouse.model` is that of the model in page 2463, figure 2. The vectors `latta.hopland.model` and `latta.sierra.model` represent the models in page 2465, figures 3 and 4. We get fairly similar unstandardised and standardised path coefficients, but not nearly similar maximum likelihood chi-squared values. The covariance matrices were made available by the authors in the supporting information.

Source

Article:


Supplementary information:


References


Examples

```r
## Not run:
data(latta)
cat(latta.greenhouse.model)
cat(latta.hopland.model)
cat(latta.sierra.model)

latta.greenhouse.fit <- sem(latta.greenhouse.model, sample.cov =
                               latta.greenhouse.cov, sample.nobs = 188)
latta.hopland.fit <- sem(latta.field.model, sample.cov =
                             latta.hopland.cov, sample.nobs = 188)
latta.sierra.fit <- sem(latta.field.model, sample.cov =
                             latta.sierra.cov, sample.nobs = 188)

summary(latta.greenhouse.fit, standardize = TRUE)
summary(latta.hopland.fit, standardize = TRUE)
```
Determinants of nitrification potential in a pine forest ecosystem.

Description

Raw data and model syntaxes for the paper by Laughlin (2011) determinants of nitrification potential in a pine forest.

Usage

```r
data(laughlin)
laughlin
laughlin.model
```

Format

A data frame with 82 observations on 11 variables and one character vector with the syntax for the final model.

Note

The final model is represented by figure 3. The model converges and we get the same standardized parameters and degrees of freedom. We get a slightly higher p-value for the robust maximum likelihood chi-square than reported. Data were transformed following instructions the supplementary material (Table S1).

Source

Article and supplementary info:


References


Examples

```r
data(laughlin)
fit <- sem(laughlin.model, data = laughlin, estimator="MLM")
summary(fit, standardized = TRUE)
```
Build the 4 basic matrices of the LISREL "all-y" representation given a set of starting values.

Description

These functions describe the structure of a path or latent variable model given a set of starting values. All fixed, free, and constrained (not yet implemented) go in them.

Usage

- BETA(model)
- PSI(model)
- LAMBDA.Y(model)
- THETA.EPSILON(model)

Arguments

model A lavaanified model.

Value

A structural matrix with and ids attribute indicating the positions of the parameters in the lavaanified model used as input.

References


Examples

data(albert)
tmp <- lvnfy(albert.model, data = albert.litho.cov, n = 107)
BE <- BETA(tmp)
PS <- PSI(tmp)
LY <- LAMBDA.Y(tmp)
TE <- THETA.EPSILON(tmp)
Set the ground to start analysing a structural equation model.

Description
This function uses lavaan to build an internal representation of a model to be fit.

Usage
lvnfy(model, data, n = NULL, ...)

Arguments
- model: Model syntax, as in lavaan.
- data: Either a covariance matrix or raw data.
- n: The number of observations, only used if data is a covariance matrix.
- ...: Other arguments to be passed to sem.

Value
A lavaanified model.

Examples
data(albert)
 tmp <- lvnfy(albert.model, data = albert.litho.cov, n = 107)
tmp

Display the indicator (manifest) variables of a lavaanified model

Description
This function returns a character vector containing the names of the manifest variables in a given model.

Usage
manifests(model)

Arguments
- model: A lavaanified model.
matrix.representation

Examples

data(pugesek)
model <- lvnfy(pugesek.onefactor.model, data = bumpus.log)
manifests(model)

matrix.representation(model)  # Builds the LISREL "all-y" matrix representation of a model.

Description

This function builds the "all-y" matrix representation of a lavaanified model using the starting values of the model.

Usage

matrix.representation(model)

Arguments

model  # A lavaanified model.

Value

beta  # BETA matrix
psi  # PSI matrix
lambda.y  # LAMBDA.Y matrix
theta.episilon  # THETA.EPISILON matrix

See Also

LISREL

Examples

## Not run:
data(albert)
tmp <- lvnfy(albert.model, data = albert.litho.cov, n = 107)
matrix.representation(tmp)

## End(Not run)
**mirror.tri**

*Place the values below or above the diagonal of a square matrix in the opposite triangle.*

**Description**

This function gets the values below or above the diagonal of a square matrix and transposes them.

**Usage**

```r
mirror.tri(mat, upper.on.lower = FALSE)
```

**Arguments**

- `mat`: A square matrix.
- `upper.on.lower`: Should the upper values be transposed to the triangle below the diagonal?

**ML**

*Maximum-likelihood fitting function.*

**Description**

ML fitting function to pass to an optimiser.

**Usage**

```r
fml(sigma, sigma.hat)
```

**Arguments**

- `sigma`: Population covariance matrix.
- `sigma.hat`: Model-implied covariance matrix.

**See Also**

- `fgls`

**Examples**

```r
## Not run:
data(albert)
cat(albert.model)
fit <- fit.nlminb(albert.model, albert.litho.cov, 107)
fml(albert.litho.cov, fit$sigma.hat)

## End(Not run)
```
**model.type**  
*Displays the type of a structural equations model.*

**Description**  
This function returns the type of a structural equations model.

**Usage**
```
model.type(model)
```

**Arguments**
- `model`: A lavaanified model.

**Value**
A character string with the type of the model. It can be either "pa", "cfa", or "sem".

**Examples**
```
data(albert)
model <- lvnfy(albert.model, data = albert.litho.cov, n = 107)
model.type(model)
```

---

**Multivariate Normal**  
*Random multivariate normal distribution.*

**Description**
This function returns a set of uncorrelated variables following a normal distribution.

**Usage**
```
rmnorm(covmat, means = 0, sds = NULL, n)
```

**Arguments**
- `covmat`: Covariance matrix to be decomposed.
- `means`: A vector of means. If 0 data are centred. Otherwise, its size must be equal in length to the number of variables in `covmat`.
- `sds`: Optional vector of standard deviations. Must also have length equal to the number of variables in `covmat`.
- `n`: Number of observations.
Value

A matrix.

References


Examples

data(latta)
mat <- rmnorm(latta.greenhouse.cov, n = 1000)
mat
latta.greenhouse.cov
cov(mat)

-----------------------------
observed

Display the observed variables of a lavaanified model

Description

This function returns a character vector containing the names of the observed variables in a given model.

Usage

observed(model)

Arguments

model A lavaanified model.

Examples

data(albert)
model <- lvnfy(albert.model, data = albert.litho.cov, n = 107)
exogenous(model)
<table>
<thead>
<tr>
<th>P-value</th>
<th>Returns the p-value given a z-score.</th>
</tr>
</thead>
</table>

**Description**

P-value of a z-score. Used internally to calculate the p-value of the maximum likelihood chi-squared statistic.

**Usage**

`pval(z)`

**Arguments**

`z`  
A z-score value.

---

**Description**

Pugesek and Tomer reanalyse this classic dataset using structural equation modeling.

**Usage**

```r
data(pugesek)
bumpus
bumpus.log
pugesek.onefactor.model
pugesek.threefactors.model
```

**Format**

Two data frames, one untransformed and one log-transformed, with 136 observations on 13 variables. Also, two character vectors with the syntaxes of two of the models described in the paper.

**Note**

According to the authors, they used log-transformed data in their analyses, so there are two datasets: `bumpus` and `bumpus.log`. Not all traits measured were used by the authors in the paper. All analyses used sex as the grouping variable. The one-factor model is in page 392, figure 3 and table 1. The three-factor model is in page 393 and 394, figure 5 and table 2. The survival model in page 395 could not be fitted as there is no support for categorical variables in either `lavaan` or `stremo`. Three-factor models 2 and 3 converge and we get somewhat similar loadings, but there are negative variances, which indicate Heywood cases. This might be caused by the equality constraints we imposed in some of the loadings to try to mimic the results from the paper.
pugesek

Source

Article:

https://springerlink3.metapress.com/content/m48860525r801047/resource-secured/?target=fulltext.pdf

Data:

http://fm1.fieldmuseum.org/aa/staff_page.cgi?staff=lowther&id=432

References


Examples

```r
## Not run:
data(pugesek)
cat(pugesek.onefactor.model)
cat(pugesek.threefactors.model)

pugesek.onefactor.1 <- sem(pugesek.onefactor.model, data = bumpus.log, fixed.x = FALSE, group = "sex", std.lv = 1, estimator = "MLM")
pugesek.onefactor.2 <- sem(pugesek.onefactor.model, data = bumpus.log, fixed.x = FALSE, group = "sex", std.lv = 1, group.equal = "loadings")
pugesek.onefactor.3 <- sem(pugesek.onefactor.model, data = bumpus.log, fixed.x = FALSE, group = "sex", std.lv = 1, group.equal = c("loadings", "residuals"))
pugesek.threefactors.1 <- sem(pugesek.threefactors.model, data = bumpus.log, fixed.x = FALSE, group = "sex", orthogonal = TRUE, std.lv = 1)
pugesek.threefactors.2 <- sem(pugesek.threefactors.model, data = bumpus.log, fixed.x = FALSE, group = "sex", orthogonal = TRUE, std.lv = 1, group.equal = c("loadings"))
pugesek.threefactors.3 <- sem(pugesek.threefactors.model, data = bumpus.log, fixed.x = FALSE, std.lv = TRUE, orthogonal = TRUE)

summary(pugesek.onefactor.1, standardize = TRUE)
summary(pugesek.onefactor.2, standardize = TRUE)
summary(pugesek.onefactor.3, standardize = TRUE)

summary(pugesek.threefactors.1, standardize = TRUE)
summary(pugesek.threefactors.2, standardize = TRUE)
summary(pugesek.threefactors.3, standardize = TRUE)

## End(Not run)
```
Residuals of a fitted path or structural equations model.

Description

This function returns the residuals, that is, the difference between a population covariance matrix and a model-implied one. Useful to look for misspecifications and general fit of the data to the model.

Usage

```r
## S3 method for class 'sem'
residuals(object, ...)
```

Arguments

- `object`: An object of class "sem" as returned by `fit.nlminb`.
- `...`: Not used.

Value

An nvar x nvar covariance

Examples

```r
## Not run:
data(albert)
fit <- fit.nlminb(albert.model, albert.litho.cov, 107)
residuals(fit)

## End(Not run)
```

---

RMSEA

**Description**

RMSEA approximate fit index

**Usage**

```r
RMSEA(fitted.model)
```

**Arguments**

- `fitted.model`: A fitted model of class "sem", as returned by `fit.nlminb`. 
sigma.hat

Value

RMSEA  The RMSEA index
ci      Confidence interval for the index
RMSEA  P-value

Note

This function was heavily influenced by code in lavaan.

Examples

## Not run:
data(albert)
fit <- fit.nlminb(albert.model, albert.litho.cov, 107)
RMSEA(fit)

## End(Not run)

sigma.hat          Calculates the sigma hat given a LISREL "all-y" matrix representation object.

Description

This function takes an object with all 4 basic LISREL matrices plus the identity matrix and constructs the sigma hat, that is, a predicted covariance matrix.

Usage

sigma.hat(matrep)

Arguments

matrep       The matrix representation of a lavaanified model.

Value

A covariance matrix.

See Also

sem.matrices and matrix.representation.
seNsem(hess, n)

Arguments

hess A hessian matrix.
n The number of observations.

Value

A vector of standard errors.

stiles

Fragmentation effects on remnant plant species richness.

Description

Raw, untransformed data and model syntax of the SEM tested in the paper by Stiles and Scheiner 2010.

Usage

data(stiles)
stiles
stiles.model

Format

A data frame with 22 observations on 11 variables and a character vector with model specification. See source for more details.

Source

Article:
References


Examples

---

```r
## Not run:
data(stiles)
cat(stiles.model)
stiles.trans <- stiles
stiles.trans[, c("frag.area", "isolation", "mean.plant.density")]
  <- log(stiles.trans[, c("frag.area", "isolation", 
       "mean.plant.density")])
stiles.model.fit <- sem(stiles.model, data = stiles.trans)
## Bootstrapping
boot.fit <- sem(stiles.model, sample.cov = cov(stiles.trans), sample.nobs = 22)
stiles.boot <- boot.lavaan(boot.fit, n = 1000)
## End(Not run)
```

---

Structural Matrices and Sigma

*Structural and covariance matrices describing a path or latent variables model.*

Description

This function builds the 8 basic matrices that describe a model alongside with a model-implied covariance matrix based on the starting values of the parameters in the model passed as an argument. This is the workinghorse of the package.

Usage

```
sem.matrices(model, matrep)
```

Arguments

- `model`: A lavaanned model with non NA starting values.
- `matrep`: A skeleton of the matrices that describe the model in question. Used mainly internally to avoid repetitive computations. Given by `matrix.representation`.

Value

A list with the 4 basic matrices (see `LISREL`) and a model-implied covariance matrix.

See Also

`LISREL`, `sigma.hat`, `matrix.representation`.
Index

*Topic covariances
  is.within.bounds, 15
*Topic datasets
  albert, 3
  grace, 10
  harnik, 11
  lamb, 17
  lamb2, 18
  latta, 19
  laughlin, 21
  pugesek, 28
  stiles, 32
*Topic hermitian
  is.hermitian, 13
*Topic package
  stremo-package, 2
*Topic positive-definite
  is.pd, 14
albert, 3
BETA (LISREL), 22
boot.lavaan, 4
bumpus (pugesek), 28
cfa, 4
cor2cov, 5
cov2cor, 5
degrees of freedom, 6
endogenous, 7
exogenous, 7, 7
factanal.pcomp (Factor analysis), 8
Factor analysis, 8
fgls, 16, 25
fgls (GLS), 10
fit.nlminb, 8, 16, 30
fml, 10, 16
fml (ML), 25
GLS, 10
grace, 10

harnik, 11
II, 13
is.hermitian, 13, 15
is.pd, 14, 15
is.within.bounds, 14, 15, 15
iterator, 16
jumpstart, 16
lamb, 17
lamb2, 18
LAMBDAY (LISREL), 22
latents, 7, 19
latta, 19
laughlin, 21
lavaan, 4, 8, 9, 23, 28, 31
LISREL, 22, 24, 33
lnfvy, 23
manifests, 7, 23
matrix.representation, 16, 24, 31, 33
mirror.tri, 5, 25
ML, 25
model.df (degrees of freedom), 6
model.type, 26
Multivariate Normal, 26

nlminb, 8, 9, 16

observed, 7, 27

P-value, 28
princomp, 8
PSI (LISREL), 22
pugesek, 28
pval (P-value), 28

Residuals, 30
residuals.sem (Residuals), 30
rmnorm, 8
rmnorm (Multivariate Normal), 26
RMSEA, 30

se.sem (Standard errors), 32
sem, 4, 8, 9, 23
sem.matrices, 31
sem.matrices (Structural Matrices and Sigma), 33
sigma.hat, 31, 33
solve, 14
Standard errors, 32
stiles, 32
stremo, 28
stremo (stremo-package), 2
stremo-package, 2
Structural Matrices and Sigma, 33

THETA.EPSILON (LISREL), 22