Package ‘semTools’

June 5, 2012

Type Package

Title Useful tools for structural equation modeling.

Version 0.1-4

Date 2012-06-05

Author Sunthud Pornprasertmanit <psunthud@ku.edu>, Patrick Miller
<patr1ckm@ku.edu>, Alex Schoe-
mann <schoemann@ku.edu>, Yves Rosseel <Yves.Rosseel@UGent.be>

Maintainer Sunthud Pornprasertmanit <psunthud@ku.edu>, Patrick Miller
<patr1ckm@ku.edu>, Alex Schoemann <schoemann@ku.edu>

Depends R(>= 2.14), MASS, lavaan, methods

Suggests parallel, Amelia, mice

Description This package provide useful tools for structural equation modeling analysis.

License GPL (>= 2)

LazyLoad yes

LazyData yes

URL https://github.com/simsem/semTools/wiki

R topics documented:

`kurtosis` *Finding excessive kurtosis*

Description

Finding excessive kurtosis (g2) of an object

Usage

```
kurtosis(object, population=FALSE)
```

Arguments

`object`	A vector used to find a excessive kurtosis
`population`	`TRUE` to compute the parameter formula. `FALSE` to compute the sample statistic formula.

Details

The excessive kurtosis computed is g2. The parameter excessive kurtosis γ_2 formula is

$$\gamma_2 = \frac{\mu_4}{\mu_2^2} - 3,$$

where μ_i denotes the i order central moment.

The excessive kurtosis formula for sample statistic g_2 is

$$g_2 = \frac{k_4}{k_2^2},$$

where k_i are the i order *k*-statistic.

The standard error of the excessive kurtosis is

$$Var(\hat{g}_2) = \frac{24}{N}$$

where N is the sample size.

Value

A value of an excessive kurtosis with a test statistic if the population is specified as `TRUE`

Author(s)

Sunthud Pornprasertmanit (University of Kansas; <psunthud@ku.edu>)

References

Weisstein, Eric W. (n.d.). *Kurtosis.* Retrived from MathWorld–A Wolfram Web Resource http://mathworld.wolfram.com/Kurtosis.html

Examples

```
kurtosis(1:5)
```

measurementInvariance *Measurement Invariance Tests*

Description

Testing measurement invariance across groups using a typical sequence of model comparison tests.

Usage

```
measurementInvariance(..., strict = FALSE, quiet = FALSE)
```

Arguments

`...`	The same arguments as for any lavaan model. See `cfa` for more information.
`strict`	If `TRUE`, the sequence requires 'strict' invariance. See details for more information.
`quiet`	If `TRUE`, a summary is printed out containing an overview of the different models that are fitted, together with some model comparison tests.

Details

If `strict = FALSE`, the following four models are tested in order:

1. Model 1: configural invariance. The same factor structure is imposed on all groups.
2. Model 2: weak invariance. The factor loadings are constrained to be equal across groups.
3. Model 3: strong invariance. The factor loadings and intercepts are constrained to be equal across groups.
4. Model 4: The factor loadings, intercepts and means are constrained to be equal across groups.

Each time a more restricted model is fitted, a chi-square difference test is reported, comparing the current model with the previous one, and comparing the current model to the baseline model (Model 1). In addition, the difference in cfi is also reported (delta.cfi).

If `strict = TRUE`, the following five models are tested in order:

1. Model 1: configural invariance. The same factor structure is imposed on all groups.
2. Model 2: weak invariance. The factor loadings are constrained to be equal across groups.
3. Model 3: strong invariance. The factor loadings and intercepts are constrained to be equal across groups.
4. Model 4: strict invariance. The factor loadings, intercepts and residual variances are constrained to be equal across groups.
5. Model 5: The factor loadings, intercepts, residual variances and means are constrained to be equal across groups.

Note that if the chi-square test statistic is scaled (eg. a Satorra-Bentler or Yuan-Bentler test statistic), a special version of the chi-square difference test is used as described in `http://www.statmodel.com/chidiff.shtml`

Value

Invisibly, all model fits in the sequence are returned as a list.

Author(s)

Yves Rosseel <Yves.Rosseel@UGent.be>

References

Vandenberg, R. J., and Lance, C. E. (2000). A review and synthesis of the measurement invariance literature: Suggestions, practices, and recommendations for organizational research. *Organizational Research Methods, 3,* 4-70.

Examples

```
HW.model <- ' visual =~ x1 + x2 + x3
              textual =~ x4 + x5 + x6
              speed =~ x7 + x8 + x9 '

measurementInvariance(HW.model, data=HolzingerSwineford1939, group="school")
```

---

miPowerFit — *Modification indices and their power approach for model fit evaluation*

Description

The model fit evaluation approach using modification indices and their power proposed by Saris, Satorra, and van der Veld (2009, pp. 570-573).

Usage

```
miPowerFit(lavaanObj, stdLoad=0.4, cor=0.1, stdBeta=0.1, intcept=0.2, stdDelta=NULL, delta=NULL)
```

Arguments

`lavaanObj`	The lavaan model object used to evaluate model fit
`stdLoad`	The amount of standardized factor loading that one would like to be detected (rejected). The default value is 0.4, which is suggested by Saris and colleagues (2009, p. 571).
`cor`	The amount of factor or error correlations that one would like to be detected (rejected). The default value is 0.1, which is suggested by Saris and colleagues (2009, p. 571).
`stdBeta`	The amount of standardized regression coefficients that one would like to be detected (rejected). The default value is 0.1, which is suggested by Saris and colleagues (2009, p. 571).
`intcept`	The amount of standardized intercept (similar to Cohen's *d* that one would like to be detected (rejected). The default value is 0.2, which is equivalent to a low effect size proposed by Cohen (1988, 1992).
`stdDelta`	The vector of the standardized parameters that one would like to be detected (rejected). If this argument is specified, the value here will overwrite the other arguments above. The order of the vector must be the same as the row order from modification indices from the `lavaan` object. If a single value is specified, the value will be applied to all parameters.

`delta` The vector of the unstandardized parameters that one would like to be detected (rejected). If this argument is specified, the value here will overwrite the other arguments above. The order of the vector must be the same as the row order from modification indices from the `lavaan` object. If a single value is specified, the value will be applied to all parameters.

Details

In the lavaan object, one can inspect the modification indices and expected parameter changes. Those values can be used to evaluate model fit by the method proposed by Saris and colleagues (2009). First, one should evaluate whether the modification index of each parameter is significant. Second, one should evaluate whether the power to detect a target expected parameter change is high enough. If the modification index is not significant and the power is high, there is no misspecification. If the modification index is significant and the power is low, the fixed parameter is misspecified. If the modification index is significant and the power is high, the expected parameter change is investigated. If the expected parameter change is large (greater than the the target expected parameter change), the parameter is misspecified. If the expected parameter change is low (lower than the target expected parameter change), the parameter is not misspecificied. If the modification index is not significant and the power is low, the decision is inconclusive.

Value

A data frame with these variables:

1. lhs The left-hand side variable (with respect to the lavaan operator)
2. op The lavaan syntax operator: "~~" represents covariance, "=~" represents factor loading, "~" represents regression, and "~1" represents intercept.
3. rhs The right-hand side variable (with respect to the lavaan operator)
4. group The group of the parameter
5. mi The modification index of the fixed parameter
6. epc The expected parameter change if the parameter is freely estimated
7. target.epc The target expected parameter change that represents the minimum size of misspecification that one would like to be detected by the test with a high power
8. std.epc The standardized expected parameter change if the parameter is freely estimated
9. std.target.epc The standardized target expected parameter change
10. significant.mi Represents whether the modification index value is significant
11. high.power Represents whether the power is enough to detect the target expected parameter change
12. decision The decision whether the parameter is misspecified or not: `"M"` represents the parameter is misspecified, `"NM"` represents the parameter is not misspecified, `"EPC:M"` represents the parameter is misspecified decided by checking the expected parameter change value, `"EPC:NM"` represents the parameter is not misspecified decided by checking the expected parameter change value, and `"I"` represents the decision is inconclusive.

Author(s)

Sunthud Pornprasertmanit (University of Kansas; <psunthud@ku.edu>)

References

Cohen, J. (1988). *Statistical power analysis for the behavioral sciences* (2nd ed.). Hillsdale, NJ: Erlbaum.

Cohen, J. (1992). A power primer. *Psychological Bulletin, 112*, 155-159.

Saris, W. E., Satorra, A., & van der Veld, W. M. (2009). Testing structural equation models or detection of misspecifications? *Structural Equation Modeling, 16*, 561-582.

Examples

```
library(lavaan)

HS.model <- ' visual  =~ x1 + x2 + x3
              textual =~ x4 + x5 + x6
              speed   =~ x7 + x8 + x9 '

fit <- cfa(HS.model, data=HolzingerSwineford1939, group="sex", meanstructure=TRUE)
miPowerFit(fit)

model <- '
  # latent variable definitions
     ind60 =~ x1 + x2 + x3
     dem60 =~ y1 + a*y2 + b*y3 + c*y4
     dem65 =~ y5 + a*y6 + b*y7 + c*y8

  # regressions
    dem60 ~ ind60
    dem65 ~ ind60 + dem60

  # residual correlations
    y1 ~~ y5
    y2 ~~ y4 + y6
    y3 ~~ y7
    y4 ~~ y8
    y6 ~~ y8
'
fit2 <- sem(model, data=PoliticalDemocracy, meanstructure=TRUE)
miPowerFit(fit2, stdLoad=0.3, cor=0.2, stdBeta=0.2, intcept=0.5)
```

`monteCarloMed` *Monte Carlo Confidence Intervals to Test Complex Indirect Effects*

Description

This function takes an expression for an indirect effect, the parameters and standard errors associated with the expression and returns a confidence interval based on a Monte Carlo test of mediation (MacKinnon, Lockwood, & Williams, 2004).

Usage

```
monteCarloMed(expression, ..., ACM=NULL, rep=20000, CI=95, plot=FALSE, outputValues=FALSE)
```

Arguments

`expression`	A character scalar representing the computation of an indirect effect. Different parameters in the expression should have different alphanumeric values. Expressions can use either addition (+) or multiplication (*) operators.
`...`	Parameter estimates for all parameters named in `expression`. The order of parameters should follow from `expression` (the first parameter named in `expression` should be the first parameter listed in ...). Alternatively ... can be a vector of parameter estimates.
`ACM`	A matrix representing the asymptotic covariance matrix of the parameters described in `expression`. This matrix should be a symetric matrix with dimensions equal to the number of parameters names in `expression`. Information on finding the ACOV is popular SEM software is described below.)
`rep`	The number of replications to compute. Many thousand are reccomended.
`CI`	Width of the confidence interval computed.
`plot`	Should the function output a plot of simulated values of the indirect effect?
`outputValues`	Should the function output all simulated values of the indirect effect?

Details

This function implements the Monte Carlo test of mediation first described in MacKinnon, Lockwood, & Williams (2004) and extends it to complex cases where the indirect effect is more than a function of two parameters. The function takes an expression for the indirect effect, randomly simulated values of the indirect effect based on the values of the parameters (and the associated standard errors) comprising the indirect effect, and outputs a confidence interval of the indirect effect based on the simulated values. For further information on the Monte Carlo test of mediation see MacKinnon, Lockwood, & Williams (2004), Preacher & Selig (in press), and Selig & Preacher (2008). For a Monte Carlo test of mediation with a random effects model see Selig & Preacher (2010).

The asymptotic covariance matrix can be easily found in many popular SEM software applications.

- LISRELIncluding the EC option on the OU line will print the ACM to a seperate file. The file contains the lower triangular elements of the ACM in free format and scientific notation
- MplusInclude the command TECH3; in the OUTPUT section. The ACM will be printed in the output.
- lavaan Use the command `vcov` on the fitted lavaan object to print the ACM to the screen

Value

A matrix with values for the upper and lower limits of the confidence interval generated from the Monte Carlo test of mediation. If `outputValues=TRUE`, output will be a list with a matrix with values for the upper and lower limits of the confidence interval as the first element and a vector of simulated values of the indirect effect as the second element.

Author(s)

Corbin Quick (University of Kansas; <corbinq@ku.edu>) Alexander M. Schoemann (University of Kansas; <schoemann@ku.edu>) James P. Selig (University of New Mexico; <selig@unm.edu>

References

Preacher, K. J., & Selig, J. P. (2010, July). Monte Carlo method for assessing multilevel mediation: An interactive tool for creating confidence intervals for indirect effects in 1-1-1 multilevel models [Computer software]. Available from http://quantpsy.org/.

Preacher, K. J., & Selig, J. P. (in press). Advantages of Monte Carlo confidence intervals for indirect effects. *Communication Methods and Measures*.

Selig, J. P., & Preacher, K. J. (2008, June). Monte Carlo method for assessing mediation: An interactive tool for creating confidence intervals for indirect effects [Computer software]. Available from http://quantpsy.org/.

Examples

```
#Simple two path mediation
#Write expression of indirect effect
med <- 'a*b'
#Paramter values from analyses
aparam <- 1
bparam<-2
#Asymptotic covariance matrix from analyses
AC <- matrix(c(.01,.00002,
               .00002,.02), nrow=2, byrow=TRUE)
#Compute CI, include a plot
monteCarloMed(med, coef1=aparam, coef2=bparam, outputValues=FALSE, plot=TRUE, ACM=AC)

#Use a matrix of parameter estimates as input
aparam<-c(1,2)
monteCarloMed(med, coef1=aparam, outputValues=FALSE, plot=TRUE, ACM=AC)



#complex mediation with two paths for the indirect effect
#Write expression of indirect effect
med <- 'a1*b1 + a1*b2'
#Paramter values and standard errors from analyses
aparam <- 1
b1param<-2
b2param<-1
#Asymptotic covariance matrix from analyses
AC <- matrix(c(1,.00002, .00003,
                    .00002,1, .00002,
.00003, .00002, 1), nrow=3, byrow=TRUE)
#Compute CI do not include a plot
monteCarloMed(med, coef1=aparam, coef2=b1param, coef3=b2param, ACM=AC)
```

`moreFitIndices` *Calculate more fit indices*

Description

Calculate more fit indices that are not already provided in lavaan.

Usage

```
moreFitIndices(object, nPrior = 1)
```

Arguments

`object`	The lavaan model object provided after running the `cfa` or the `sem` functions.
`nPrior`	The sample size on which prior is based. This argument is used to compute BIC*.

Details

Normed Fit Index (nfi; West, Taylor, & Wu, 2012) is one of the relative fit indices which can be computed by

$$nfi = \frac{\chi_0^2 - \chi_k^2}{\chi_0^2},$$

where χ_k^2 is the chi-square test statistic value of the target model, χ_0^2 is the chi-square test statistic value of the null model.

Incremental Fit Index (ifi; West, Taylor, & Wu, 2012) is one of the relative fit indices which can be computed by

$$ifi = \frac{\chi_0^2 - \chi_k^2}{\chi_0^2 - df_k},$$

where df_k is the degree of freedom when fitting the target model

Gamma Hat (gfi*; West, Taylor, & Wu, 2012) is one of the relative fit indices which can be computed by

$$gfi* = \frac{p}{p + 2 \times \frac{\chi_k^2 - df_k}{N-1}},$$

where N is the sample size, p is the number of variables in the model.

Adjusted Gamma Hat (agfi*; West, Taylor, & Wu, 2012) is one of the relative fit indices which can be computed by

$$agfi* = \left(1 - \frac{p \times (p+1)}{2 \times df_k}\right) \times (1 - gfi*),$$

Corrected Akaike Information Criterion (AICc; Burnham & Anderson, 2003) is the corrected version of aic for small sample size:

$$aicc = f + \frac{2k(k+1)}{N - k - 1},$$

where f is the minimized discrepancy function, which is the product of the log likelihood and -2, and k is the number of parameters in the target model.

Expected Value of Cross-Validation Index (ECVI; West, Taylor, & Wu, 2012) is the average discrepancy in the fitted covariance matrices between two samples of equal sample size across all possible combinations of two samples from the same population:

$$ecvi = f + \frac{2 \times k}{N},$$

Stochastic information criterion (sic; Preacher, 2006) is similar to aic or bic. This index will account for model complexity in the model's function form, in addition to the number of free parameters. sic can be computed by

$$sic = \frac{1}{2}\left(f - \log \det I(\hat{\theta})\right),$$

where $I(\hat{\theta})$ is the information matrix of the parameters.

Corrected Bayesian Information Criterion (BIC*; Kuha, 2004) is similar to bic but explicitly specifying the sample size on which the prior is based (N_{prior}).

$$bicc = f + k \log\left(1 + N/N_{prior}\right),$$

Hannan-Quinn Information Criterion (hqc; Hannan & Quinn, 1979) is used for model selection similar to aic or bic.

$$hqc = f + 2k \log\left(\log N\right),$$

Value

1. nfi Normed Fit Index
2. ifi Incremental Fit Index
3. gfi* Gamma Hat
4. agfi* Adjusted Gamma Hat
5. aicc Corrected Akaike Information Criterion
6. ecvi Expected Value of Cross-Validation Index
7. sic Stochastic Information Criterion
8. bic* Bayesian Information Criterion with specifying the prior sample size
9. hqc Hannan-Quinn Information Criterion

Author(s)

Sunthud Pornprasertmanit (University of Kansas; <psunthud@ku.edu>) Aaron Boulton (University of Kansas; <aboulton@ku.edu>)

References

Burnham, K., & Anderson, D. (2003). *Model selection and multimodel inference: A practical-theoretic approach.* New York, NY: Springer-Verlag.

Kuha, J. (2004). AIC and BIC: Comparisons of assumptions and performance. *Sociological Methods Research, 33,* 188-229.

Preacher, K. J. (2006). Quantifying parsimony in structural equation modeling. *Multivariate Behavioral Research, 43,* 227-259.

West, S. G., Taylor, A. B., & Wu, W. (2012). Model fit and model selection in structural equation modeling. In R. H. Hoyle (Ed.), *Handbook of Structural Equation Modeling.* New York: Guilford.

Examples

```
HS.model <- ' visual  =~ x1 + x2 + x3
              textual =~ x4 + x5 + x6
              speed   =~ x7 + x8 + x9 '

fit <- cfa(HS.model, data=HolzingerSwineford1939)
moreFitIndices(fit)
```

`orthogonalize` *Orthogonalize data for 2-way interaction in SEM*

Description

Orthogonalize indicators of a 2-way interaction between latent variables

Usage

```
orthogonalize(dat, xvars, zvars)
```

Arguments

`dat`	Matrix or data frame of item level data.
`xvars`	A vector of column numbers corresponding to indicators of the focal predictor (x).
`zvars`	A vector of column numbers corresponding to indicators of the moderator (z).

Details

This functions will take a data frame or matrix and create orthogonalized product terms to compute latent variable interactions based on the method proposed by Little, Bovaird, & Widaman. The orthogonalized product terms can be entered into a SEM as indicators of a latent interaction variable. This function will compute all possible orthogonalized product terms (e.g., x has 3 indicators and z has 4 indicators, the function will return 3*4=12 new orthognalized product terms)

Value

1. data Original data with orthogonalized product terms appended.

Author(s)

Alexander M. Schoemann (University of Kansas; `<schoemann@ku.edu>`)

References

Little, T. D., Bovaird, J. A., & Widaman, K. F. (2006). On the merits of orthogonalizing powered and product terms: Implications for modeling interactions among latent variables. *Structural Equation Modeling, 13* 497-519.

Examples

```
library(MASS)

n <- 500
means <- c(0,0)
covmat <- matrix(c(1, 0.3, 0.3, 1),nrow=2)

data <- mvrnorm(n,means,covmat)

x<-as.vector(data[,1])
z<-as.vector(data[,2])

y<-rnorm(n,0,1)+.4*x+.4*z+.2*x*z

x1<-rnorm(n,0.2,.2)+.7*x
x2<-rnorm(n,0.2,.2)+.7*x
x3<-rnorm(n,0.2,.2)+.7*x
z1<-rnorm(n,0.2,.2)+.7*z
z2<-rnorm(n,0.2,.2)+.7*z
z3<-rnorm(n,0.2,.2)+.7*z
y1<-rnorm(n,0.2,.2)+.7*y
y2<-rnorm(n,0.2,.2)+.7*y
y3<-rnorm(n,0.2,.2)+.7*y

dat<-data.frame(cbind(x1,x2,x3,z1,z2,z3,y1,y2,y3))

datOrth <-orthogonalize(dat,(1:3), (4:6))

#Fit model in Lavaan
library(lavaan)

syntax <- '
x =~ x1 + x2 +x3
z =~ z1 + z2 + z3
xz =~ x1z1 + x1z2 + x1z3 + x2z1 + x2z2 + x2z3 + x3z1 + x3z2 + x3z3
y =~ y1 + y2 + y3
x ~~ z
x ~~ 0*xz
z ~~ 0*xz
y ~ x + z +xz
'

fit <- sem(model = syntax, data=datOrth, std.lv=TRUE)
summary(fit, fit.measures=TRUE)
```

`parcelAllocation` *Random Allocation of Items to Parcels in a Structural Equation Model*

Description

This function generates a given number of randomly generated item-to-parcel allocations, fits a model to each allocation, and provides averaged results over all allocations.

Usage

```
parcelAllocation(nPerPar, facPlc, nAlloc=100, syntax, dataset, names='default', leaveout=0, ...)
```

Arguments

nPerPar	A list in which each element is a vector corresponding to each factor indicating sizes of parcels. If variables are left out of parceling, they need not be accounted for here (i.e., parcels of size "1").
facPlc	A list in which each element is a vector corresponding to each factor indicating manifest variables in respective factor. Either variable names or column numbers. Variables not included listed will not be included in output datasets.
nAlloc	The number of random allocations of items to parcels to generate.
syntax	lavaan syntax. If substituted with a file name, parcelAllocation will print output data sets to a specified folder rather than analyzing using lavaan (note for Windows users: file path must be specified using forward slashes).
dataset	Data set. Can be file path or R object (matrix or dataframe). If the data has missing values multiple imputation before parceling is recommended.
names	(Optional) A character vector containing the names of parceled variables.
leaveout	A vector of variables to be left out of randomized parceling. Either variable names or column numbers are allowed.
...	Additional arguments to be passed to lavaan

Details

This function implements the random item to parcel allocation procedure described in Sterba (2011) and Sterba and MccCallum (2010). The function takes a single data set with item level data, randomly assigns items to parcels, fits a structural equation model to the parceled data (using lavaan), and repeats this process for a user specified number of random allocations. Results from all fitted models are summarized and output. For further details on the benefits of the random allocation of itesm to parcels see Sterba (2011) and Sterba and MccCallum (2010).

Value

Estimates	A data frame containing results related to parameter estimates with columns corresponding to parameter names, average parameter estimates across allocations, the standard deviation of parameter estimates across allocations, the minimum parameter estimate across allocations, the maximum parameter estimate across allocations, the range of parameter estimates across allocations, and the proportions of allocations in which the parameter estimate is significant.
SE	A data frame containing results related to standard errors with columns corresponding to parameter names, average standard errors across allocations, the standard deviation of standard errors across allocations, the minimum standard error across allocations, the maximum standard error across allocations, and the range of standard errors across allocations.
Fit	A data frame containing results related to model fit with columns corresponding to fit index names, the average of each index across allocations, the standard deviation of each fit index across allocations, the minimum of each fit index across allocations, the maximum of each fit index across allocations, and the range of each fit index across allocations.

Author(s)

Corbin Quick (University of Kansas; <corbinq@ku.edu>) Alexander M. Schoemann (University of Kansas; <schoemann@ku.edu>)

References

Sterba, S.K. (2011). Implications of parcel-allocation variability for comparing fit of item-solutions and parcel-solutions. *Structural Equation Modeling, 18,* 554-577. Sterba, S.K. & MacCallum, R.C. (2010). Variability in parameter estimates and model fit across random allocations of items to parcels. *Multivariate Behavioral Research, 45,* 322-358.

Examples

```
#Fit 3 factor CFA to simulated data.
#Each factor has 9 indicators that are randomly parceled into 3 parcels
#Lavaan syntax for the model to be fit to parceled data
syntax <- 'La =~ V1 + V2 + V3
           Lb =~ V4 + V5 + V6
'
#Parcel and fit data 20 times. The actual parcel number should be higher than 20 times.
name1 <- colnames(simParcel)[1:9]
name2 <- colnames(simParcel)[10:18]
parcelAllocation(list(c(3,3,3),c(3,3,3)), list(name1, name2), nAlloc=20, syntax=syntax, dataset=simParcel)
```

`plotRMSEApower` *Plot power curves for RMSEA*

Description

Plots power of RMSEA over a range of sample sizes

Usage

```
plotRMSEApower(rmsea0, rmseaA, df, nlow, nhigh, steps, alpha=.05)
```

Arguments

`rmsea0`	Null RMSEA
`rmseaA`	Alternative RMSEA
`df`	Model degrees of freedom
`nlow`	Lower sample size
`nhigh`	Upper sample size
`steps`	Increase in sample size for each iteration. Smaller values of steps will lead to more precise plots. However, smaller step sizes means a longer run time.
`alpha`	Alpha level used in power calculations

Details

This function creates plot of power for RMSEA against a range of sample sizes. The plot places sample size on the horizontal axis and power on the vertical axis. The user should indicate the lower and upper values for sample size and the sample size between each estimate ("step size") We strongly urge the user to read the sources below (see References) before proceeding. A web version of this function is available at: http://quantpsy.org/rmsea/rmseaplot.htm.

Value

1. plot Plot of power for RMSEA against a range of sample sizes

Author(s)

Alexander M. Schoemann (University of Kansas; <schoemann@ku.edu>) Kristopher J. Preacher (Vanderbilt University; <kris.preacher@vanderbilt.edu>) Donna L. Coffman (Pennsylvania State University; <dlc30@psu.edu.>)

References

MacCallum, R. C., Browne, M. W., & Cai, L. (2006). Testing differences between nested covariance structure models: Power analysis and null hypotheses. *Psychological Methods, 11,* 19-35.

MacCallum, R. C., Browne, M. W., & Sugawara, H. M. (1996). Power analysis and determination of sample size for covariance structure modeling. *Psychological Methods, 1,* 130-149.

MacCallum, R. C., Lee, T., & Browne, M. W. (2010). The issue of isopower in power analysis for tests of structural equation models. *Structural Equation Modeling, 17,* 23-41.

Preacher, K. J., Cai, L., & MacCallum, R. C. (2007). Alternatives to traditional model comparison strategies for covariance structure models. In T. D. Little, J. A. Bovaird, & N. A. Card (Eds.), *Modeling contextual effects in longitudinal studies* (pp. 33-62). Mahwah, NJ: Lawrence Erlbaum Associates.

Steiger, J. H. (1998). A note on multiple sample extensions of the RMSEA fit index. *Structural Equation Modeling, 5,* 411-419.

Steiger, J. H., & Lind, J. C. (1980, June). *Statistically based tests for the number of factors.* Paper presented at the annual meeting of the Psychometric Society, Iowa City, IA.

Examples

```
plotRMSEApower(.025, .075, 23, 100, 500, 10)
```

---

runMI *Multiply impute and analyze data using lavaan*

Description

This function takes data with missing observations, multiple imputes the data, runs a SEM using lavaan and combines the results using Rubin's rules.

Usage

```
runMI(data.mat,data.model, m, miPackage="Amelia", digits=3, seed=12345,
    std.lv = FALSE, estimator = "ML", group = NULL, group.equal = "", ...)
```

Arguments

`data.mat`	Data frame with missing observations or a list of data frames where each data frame is one imputed data set (for imputed data generated outside of the function). If a list of data frames is supplied, then other options can be left at the default.
`data.model`	lavaan syntax for the the model to be analyzed.
`m`	Number of imputations wanted.
`miPackage`	Package to be used for imputation. Currently runMI only uses Amelia or mice for imputation.
`digits`	Number of digits to print in the results.
`seed`	Random number seed to be used in imputations.
`std.lv`	lavaan option. 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.
`estimator`	lavaan option. The estimator to be used. Can be one of the following: "ML" for maximum likelihood, "GLS" for generalized least squares, "WLS" for weighted least squares (sometimes called ADF estimation), "MLM" for maximum likelihood estimation with robust standard errors and a Satorra-Bentler scaled test statistic, "MLF" for maximum likelihood estimation with standard errors based on first-order derivatives and a conventional test statistic, "MLR" for maximum likelihood estimation with robust 'Huber-White' standard errors and a scaled test statistic which is asymptotically equivalent to the Yuan-Bentler T2-star test statistic. Note that the "MLM", "MLF" and "MLR" choices only affect the standard errors and the test statistic.
`group`	lavaan option. A variable name in the data frame defining the groups in a multiple group analysis.
`group.equal`	lavaan option. A vector of character strings. Only used in a multiple group analysis. Can be one or more of the following: "loadings", "intercepts", "means", "regressions", "residuals", "residual.covariances", "lv.variances" or "lv.covariances", specifying the pattern of equality constraints across multiple groups.
`...`	Other arguments to be passed to the imputation package

Value

runMI returns a list with pooled fit indices, estimates, standard errors and fraction missing information.

`fit`	Pooled fit information. The first set of fit information are simply averaged across imputations and are not trustworthy. The second set of fit information, is a pooled Chi-square statistic based on Li, Meng, Raghunathan, & Rubin (1991)
`parameters`	Pooled parameter estimates and standard errors. Wald statistics and p values are computed from the pooled estimates and standard errors. Also contains two estimates of Fraction of Missing Information (FMI). The first estimate of FMI (FMI.1) is asymptotic FMI and the second estimate of FMI (FMI.2) is corrected for small numbers of imputation

Author(s)

Alexander M. Schoemann (University of Kansas; <schoemann@ku.edu>) Patrick Miller(University of Kansas; <patr1ckm@ku.edu>) Sunthud Pornprasertmanit (University of Kansas; <psunthud@ku.edu>) Mijke Rhemtulla (University of Kansas; <mijke@ku.edu>) Alexander Robitzsch (Federal Institute for Education Research, Innovation, and Development of the Austrian School System, Salzburg, Austria; <a.robitzsch@bifie.at>) Craig Enders (Arizona State University; <Craig.Enders@asu.edu>)

References

Li, K.H., Meng, X.-L., Raghunathan, T.E. and Rubin, D.B. (1991). Significance Levels From Repeated p-values with Multiply-Imputed Data. Statistica Sinica, 1, 65-92. Rubin, D.B. (1987) Multiple Imputation for Nonresponse in Surveys. J. Wiley & Sons, New York.

Examples

```
## No Example
```

---

simParcel *Simulated Data set to Demonstrate Random Allocations of Parcels*

Description

A simulated data set with 2 factors with 9 indicators for each factor

Usage

```
data(simParcel)
```

Format

A data frame with 800 observations of 18 variables.

f1item1 Item 1 loading on factor 1

f1item2 Item 2 loading on factor 1

f1item3 Item 3 loading on factor 1

f1item4 Item 4 loading on factor 1

f1item5 Item 5 loading on factor 1

f1item6 Item 6 loading on factor 1

f1item7 Item 7 loading on factor 1

f1item8 Item 8 loading on factor 1

f1item9 Item 9 loading on factor 1

f2item1 Item 1 loading on factor 2

f2item2 Item 2 loading on factor 2

f2item3 Item 3 loading on factor 2

f2item4 Item 4 loading on factor 2

f2item5 Item 5 loading on factor 2

f2item6 Item 6 loading on factor 2

f2item7 Item 7 loading on factor 2

f2item8 Item 8 loading on factor 2

f2item9 Item 9 loading on factor 2

Source

Data was generated using the `simsem` package.

Examples

```
head(simParcel)
```

`skew` *Finding skewness*

Description

Finding skewness (g1) of an object

Usage

```
skew(object, population=FALSE)
```

Arguments

`object`	A vector used to find a skewness
`population`	`TRUE` to compute the parameter formula. `FALSE` to compute the sample statistic formula.

Details

The skewness computed is g1. The parameter skewness γ_2 formula is

$$\gamma_2 = \frac{\mu_3}{\mu_2^{3/2}},$$

where μ_i denotes the i order central moment.

The excessive kurtosis formula for sample statistic g_2 is

$$g_2 = \frac{k_3}{k_2^2},$$

where k_i are the i order k-statistic.

The standard error of the skewness is

$$Var(\hat{g}_2) = \frac{6}{N}$$

where N is the sample size.

Value

A value of a skewness with a test statistic if the population is specified as `TRUE`

Author(s)

Sunthud Pornprasertmanit (University of Kansas; <psunthud@ku.edu>)

References

Weisstein, Eric W. (n.d.). *Skewness.* Retrived from MathWorld–A Wolfram Web Resource http://mathworld.wolfram.com/Skewness.html

Examples

```
skew(1:5)
```

Index