Package 'semTable'

April 29, 2020

Type Package

Title Structural Equation Modeling Tables

Version 1.8

Date 2020-04-29

Maintainer Paul Johnson <pauljohn@ku.edu>

Description For confirmatory factor analysis ('CFA') and structural equation
models ('SEM') estimated with the 'lavaan' package,
this package provides functions to create model summary tables and model
comparison tables for hypothesis testing. Tables can be produced in
'LaTeX', 'HTML', or comma separated variables ('CSV').

License GPL-2

Depends R (>= 3.0)

Imports stats, utils, kutils, xtable, lavaan, plyr, stationery

Suggests rockchalk

RoxygenNote 7.1.0

LazyData TRUE

VignetteBuilder stationery

NeedsCompilation no

Author Paul Johnson [aut, cre],
Benjamin Kite [aut]

Repository CRAN

Date/Publication 2020-04-29 18:20:02 UTC

R topics documented:

---

compareLavaan	*Prepare a table to compare fit measures of confirmatory factor analyses (CFA)*

Description

If the parameter `nesting` is not specified, then this function attempts to discern which models are nested and they are ordered accordingly. The user can override that by specifing a nesting structure. This uses a new notation to represent nesting of models. See the parameter `nesting`. The syntax uses the symbols ">" and "+" in an obvious way to indicate that one model is the superset or on the same level as another. If the

Usage

```
compareLavaan(models, fitmeas = c("chisq", "df", "pvalue", "rmsea",
  "cfi", "tli", "srmr", "aic", "bic"), nesting = NULL, scaled = TRUE,
  chidif = TRUE, digits = 3, ...)
```

Arguments

models	list of lavaan cfa or sem models. Model names can be supplied. See examples.
fitmeas	A vector of fit measures. One or more of these c("chisq","df","pvalue","rmsea","cfi","tli","sr Other fit measures present in the lavaan objects will be allowed; fit measures that are requested but not found are ignored.
nesting	character string indicating the nesting structure of the models. Must only contain model names, ">", and "+" separated by spaces. The model to the left of a ">" is the parent model for all models to the right of the same ">", up until another ">" is reached. When multiple models are nested in the same parent, they are separated by a "+".
scaled	should scaled versions of the fit measures requested be used if available? The scaled statistic is determined by the model estimation method. The defaul value is TRUE.
chidif	should the nested models be compared by using the anova function? The anova function may pass the model comparison on to another lavaan function. The results are added to the last three columns of the comparison table. The default value is TRUE.
digits	The digits argument that will be passed to xtable.
...	Arguments that will be passed to print.xtable. These arguments can be used to control table caption, label, and so forth. See ?print.xtable. If type = "latex" or "html", this function sets additional default values for print.xtable

that can be overridden by specifying arguments here. Default type is an R data.frame, which is printed on screen. Note the print.xtable parameter `print.results` determines whether the markup is displayed before it is returned. The `file` parameter can specify a file into which the markup is to be saved.

Details

In May 2018, the output approach was changed. The functions `xtable` and `print.xtable` are used to render the final result and any of the arguments allowed by `print.xtable` can be used here (via the `...` argument). We have some default settings for some `print.xtable`, such as type = NULL, file = NULL, print.results = TRUE, and math.style.exponents = TRUE. There are some other specific defaults for type = "latex" and type = "html", but they can all be overridden by the user. We include a model legend at the bottom of the table, indicating which models are compared by the Chi-squared statistic.

If the type argument is not specified, then the output will be a simple text display of the model table. If type is either "latex" or "html", then a marked up table will be displayed and the file argument can be used to ask for a saved version. If the user wants to simply save the result in a file, and not display on screen, insert the argument `print.results = FALSE`.

Value

If type = NULL, a data.frame object includes an attribute called "noteinfo". If type = "tex", return is a character vector created by xtable. If type = "html", a vector of HTML markup created by xtable.

Author(s)

Ben Kite <bakite@ku.edu> and Paul Johnson <pauljohn@ku.edu>

Examples

```
library(lavaan)
library(xtable)
set.seed(123)
genmodel <- "f1 =~ .7*v1 + .7*v2 + .7*v3 + .7*v4 + .7*v5 + .7*v6
f1 ~~ 1*f1"
genmodel2 <- "f1 =~ .7*v1 + .7*v2 + .7*v3 + .7*v4 + .7*v5 + .2*v6
f1 ~~ 1*f1"
##'
dat1 <- simulateData(genmodel, sample.nobs = 300)
dat2 <- simulateData(genmodel2, sample.nobs = 300)
dat1$group <- 0
dat2$group <- 1
dat <- rbind(dat1, dat2)
## In order from less constrained to restricted
## Model cc1 configural model
cc1.model <- "
            f1 =~ 1*v1 + v2 + v3 + v4 + v5 + v6
     f1 ~~ f1
     f1 ~0*1
     "
## Model2: cc2 partial weak model (AKA partial metric)
```

```
cc2.model <- "
              f1 =~ 1*v1 + c(L2,L2)*v2 + c(L3,L3)*v3 + c(L4,L4)*v4 + c(L5,L5)*v5 + v6
        f1 ~~ f1
        f1 ~0*1
      "
## Model 3: weak model (AKA metric)
cc3.model <- "
          f1 =~ 1*v1 + c(L2,L2)*v2 + c(L3,L3)*v3 + c(L4,L4)*v4 + c(L5,L5)*v5 + c(L6,L6)*v6
      f1 ~~ f1
       f1 ~0*1
        "
## Model 4: scalar model (AKA strong)
cc4.model <- "
              f1 =~ 1*v1 + c(L2,L2)*v2 + c(L3,L3)*v3 + c(L4,L4)*v4 + c(L5,L5)*v5 + v6
        f1 ~~ f1
        f1 ~ c(0,NA)*1
        v1 ~ c(I1,I1)*1
        v2 ~ c(I2,I2)*1
        v3 ~ c(I3,I3)*1
        v4 ~ c(I4,I4)*1
        v5 ~ c(I5,I5)*1
        v6 ~ c(I6,I6)*1
      "

## Reload saved models if available: avoids slow re-estimation that bothers CRAN
cc1 <-  tryCatch(readRDS(system.file("cfa/cc1.rds", package = "semTable")),
                 error = function(e) cfa(cc1.model, data=dat, group="group",
                                         meanstructure=TRUE, estimator = "MLR"))
cc2 <- tryCatch(readRDS(system.file("cfa/cc2.rds", package = "semTable")),
                error = function(e) cfa(cc2.model, data=dat, group="group",
                                        meanstructure=TRUE, estimator = "MLR"))
cc3 <- tryCatch(readRDS(system.file("cfa/cc3.rds", package = "semTable")),
                 error = function(e) cfa(cc3.model, data=dat, group="group",
                                         meanstructure=TRUE, estimator = "MLR"))
cc4 <- tryCatch(readRDS(system.file("cfa/cc4.rds", package = "semTable")),
                error = function(e) cfa(cc4.model, data=dat, group="group",
                                        meanstructure=TRUE, estimator = "MLR"))

models <- list(cc1, cc2, cc3, cc4)
## Note, nesting is not specified, so built-in nesting detection applies
compareLavaan(models)
compareLavaan(models, type = "latex")
compareLavaan(models, type = "html")
##'
## Now we specify model labels in the list
models <- list("Configural" = cc1,  "PartialMetric" = cc2, "Metric" = cc3, "Scalar" = cc4)
## The model labels are used in the nesting parameter
compareLavaan(models, nesting = "Configural > PartialMetric > Metric > Scalar")
##' Previous incorrect, treat cc2 and cc3 as children of cc1 instead:
compareLavaan(models, nesting = "Configural > PartialMetric + Metric > Scalar")
##'
compareLavaan(models, fitmeas = c("chisq", "df", "cfi", "rmsea", "tli"),
              nesting = "Configural > Metric + PartialMetric > Scalar")
```

```
compareLavaan(models, fitmeas = c("chisq", "df", "cfi", "rmsea", "tli"),
              nesting = "Configural > PartialMetric + Metric > Scalar")
##'
## Creates output file
## compareLavaan(models, fitmeas = c("chisq", "df", "cfi", "rmsea", "tli"),
## nesting = "Configural > Metric + PartialMetric > Scalar", type = "tex",
## file = "/tmp/table.tex")
```

---

detectNested	*Discern nesting pattern of SEM coefficients*

Description

Receives a list of models and orders them by best guess at intended nesting

Usage

```
detectNested(models)
```

Arguments

models   A List of lavaan-fitted SEM models

Value

matrix indicating nesting relationships

Author(s)

Ben Kite <bakite@ku.edu>

---

escape	*Text is cleaned (escaped) to prevent errors when used in LaTeX, file names, or HTML output.*

Description

This is for fixing up "untrusted text" that is to be passed into a file as content. It protects against "bad" text strings in 3 contexts, 1) LaTeX documents, 2) HTML documents, or 3) text in a file name. It converts content text to an improved string that will not cause failures in the eventual document.

Usage

```
escape(x, type = "tex")
```

Arguments

x	a string, or vector of strings (each of which is processed separately)
type	"tex" is default, could be "filename" or "html"

Details

The special in-document LaTeX symbols like percent sign or dollar sign are "%" and "\$". *Warning*: In the R session, these will appear as double-backslashed symbols, while in a saved text file, there will only be the one desired slash.

If type = "html", we only clean up <, >, / and &, and quote characters. If document is in unicode, we don't need to do the gigantic set anymore.

If type = "filename", then symbols that are not allowed in file names, such as "\", "*", are replaced. Do not use this on a full path, since it will obliterate path separators.

Value

corrected character vector

Author(s)

Paul Johnson <pauljohn@ku.edu>

Examples

```
x1 <- c("_asdf&_&$", "asd adf asd_", "^ % & $asdf_")
escape(x1)
x2 <- c("a>b", "a<b", "a < c", 'Paul "pj" Johnson')
escape(x2, type = "tex")
escape(x2, type = "html")
escape(x2, type = "filename")
```

---

markupConvert	*Convert marked-up semTable structures to latex, html, or csv*

Description

The conversion key tables are included in the code of the function.

Usage

```
markupConvert(marked, type = c("latex", "html", "csv"),
  table.float = FALSE, longtable = FALSE, caption = NULL,
  label = NULL, file = NULL, columns, centering = "siunitx")
```

Arguments

`marked`	A character string
`type`	Output type, latex", "html", or "csv".
`table.float`	TRUE if you want insertion of '\begintable'
`longtable`	should a tabular or a longtable object be created?
`caption`	A caption to use if either longtable or table is TRUE
`label`	A LaTeX label for cross-references
`file`	A file stub, to which ".tex", ".html", or ".csv" can be added
`columns`	For SEM table, the list of columns objects
`centering`	Default "siunitx". Specify "none" to return to behavior of semTable before 1.50.

Details

The semTable uses a customized markup framework that uses character sequences that begin and end with underscores, such as "_BOMC2_ for "begin of multi-column entity that will use 2 columns". These special markups need to be converted into "tex", "html", or "csv" formats.

Value

a list of marked up character objects

Author(s)

Paul Johnson

`print.kutable`	*A print method for kutable objects*

Description

A print method for kutable objects

Usage

```
## S3 method for class 'kutable'
print(x, ...)
```

Arguments

`x`	object to be printed
`...`	optional arguments, corrently ignored

Value

x unchanged

---

semTable   *Creates Structural Equation Modeling Tables*

Description

Creates LaTeX markup for structural equation modeling output tables in the style of the American
Psychological Association(APA). Input objects should be created by the "lavaan" package.

Usage

```
semTable(object, file = NULL, paramSets = "all", paramSetLabels,
  columns = c(est = "Estimate", se = "SE", z = "z", p = "p"),
  columnLabels, fits = c("chisq", "cfi", "tli", "rmsea"),
  fitLabels = toupper(fits), varLabels = NULL, groups = NULL,
  type = "latex", table.float = FALSE, caption = NULL,
  label = NULL, longtable = FALSE, print.results = TRUE,
  centering = "siunitx", alpha = c(0.05, 0.01, 0.001))
```

Arguments

object	A lavaan object (e.g., returned by cfa() or sem()), or a named list of lavaan objects, e.g., `list("Model A" = obj1, "Model B" = obj2)`. Results will be displayed side by side.
file	Base name for output file. This function will insert suffix, either "tex", "html" and "csv".
paramSets	Parameter sets to be included for each fitted object. Valid values of the vector are "all" or a any of the following: `c("composites", "loadings", "slopes", "intercepts", "residualv` Default is "all", any of the estimates present in the fitted model that are listed in the previous sentence will be included in the output. For the sake of simplicity, we now allow one vector here, which applies to all models in the object list.
paramSetLabels	Named vector, used to supply alternative pretty printing labels for parameter sets. The default values are `c("composites" = "Composites", "loadings"= "Factor Loadings", "slopes" = "Regression Slopes", "intercepts" = "Intercepts", "means"= "Means", "residualvariances" = "Residual Variances", "residualcovariances" = "Residual Covariances", "variances" = "Variances", "latentvariances" = "Latent Variances", "latentcovariances" = "Latent Covariances", "latentmeans" = "Latent Intercepts", "thresholds" = "Thresholds", "constructed" = "Constructed", "fits" = "Fit Indices")`. The paramSetLabels argument must be a named vector that overrides some or all of the default names.
columns	A vector naming estimates to appear for each model. The allowed columns are "est", "se", "z", "p", "rsquare", "estse", "eststars", "estsestars". The first 5 have the usual meanings, while "estse" (can also be written "est(se)") displays as, for example "1.21(0.23)", and the last 2 are to include "significance stars". "eststars" shows as "1.21***" and "estsestars" (or "est(se)stars") displays as "1.21(0.23)**". See parameter alpha. One may request different columns for each model by providing a named list of vectors. Use model names in the list, `list("Model A" = c("est", "se"), "Model B" = c("estse", "p"))`.

columnLabels	A named vector of "pretty labels" for the headings in the table. The default labels are c("est" = "Estimate", se = "Std. Err.", z = "z", p = "p", rsquare = "R Square", estse = "Estimate(Std.Err.")", eststars = "Estimate", estsestars = "Estimate(Std.Err.)").
fits	Summary indicators to be included. May be a list, one for each model provided, otherwise the same fit indicators will be presented for each model. Any of the fit indicators provided by lavaan::fitMeasures(object) are allowed: c("npar","fmin","chisq","df","pvalue","baseline.chisq","baseline.df","baseline.pvalu The return for "chisq" will include markup for degrees of freedom and p value. If user specifies NULL, or if "fits" is excluded from paramSets, all fit indicators are omitted.
fitLabels	Labels for some or all of the fit measures requested by the fits parameter, e.g. c(rmsea = "Root Mean Square Error of Approximation", cli = "CLI"). The default labels are the upper-case fits names (except for "chisq", where a Greek letter is supplied when possible).
varLabels	Named vector of labels to replace variable names in column 1 of SEM table.
groups	All groups will be printed, unless a subset is requested here. Estimates for all groups will be displayed side by side. If ONLY SOME groups should be included, then specify groups as either names of fit objects or as integers for elements of the groups vector.
type	Choose "latex", "html", or "csv"
table.float	If TRUE, create a LaTeX floating table object in which the tabular created here will reside. Default is FALSE.
caption	Caption for table (if table.float=TRUE) or longtable output. Ignored otherwise.
label	LaTeX label for this object (for cross-references). Only used if table.float = TRUE or longtable = TRUE.
longtable	If TRUE, use longtable for LaTeX documents. Default is FALSE. If true, table.float argument is ignored.
print.results	If TRUE, marked up result will be displayed within the session. Otherwise, result is returned silently and user can use cat to dislay it. Don't use print because it inserts unwanted decorations.
centering	Default "siunitx". For method used in previous editions, replace with "none".
alpha	Thresholds for p-values that determine number of stars. Defaults as c(0.05,0.01,0.001) for c("*","**","***").

Details

The argument paramSets determines the inclusion of estimate sections. One can specify "all", which means that all types of parameters that we can find in the fitted model are presented. Otherwise, a subset of parameter sets can be chosen by the user.

- "composites" are predictor coefficients in formative constructs
- "loadings" are the factor loadings in the model.
- "slopes" are the regression slopes in the model.
- "intercepts" are the estimated constants in the measurement models.

- "residualvariances" are the observed variable residual variances.

- "residualcovariances" are the observed covariances among residuals of observed variables.

- "latentvariances" are the variances of unobserved variables.

- "latentcovariances" are the covariances between unobserved variables.

- "latentmeans" are means of unobserved variables

- "thresholds" arise in latent response variates (non-numeric indicator data).

- "constructed" are parameters that are calculated from a formula in the model specification, such as an indirect path c=a*b.

- "fits" the summary indicators of the mismatch between the theoretical and observed covariance matrices, such as RMSEA, CLI, TFI. While the fits are not technically parameters, they are displayed in the same block style as parameters

The columns parameter is used to specify different columns, while columnLabels will alter the displayed labels for them.

Value

Markup for SEM table. Includes an attribute "markedResults", which can be converted to other markup formats by the function markupConvert.

Author(s)

Ben Kite <bakite@ku.edu> Paul Johnson <pauljohn@ku.edu>

Examples

```
## Most of the examples were moved to the semTable vignette
require(lavaan)

tempdir <- tempdir()
## The example from lavaan's docs
 HS.model <- ' visual  =~ x1 + x2 + x3
               textual =~ x4 + x5 + x6
               speed   =~ x7 + x8 + x9'
 fit1 <- cfa(HS.model, data = HolzingerSwineford1939,
             std.lv = TRUE, meanstructure = TRUE)
## Try a LaTeX file first
fit1.t1 <- semTable(fit1, columns = c("estse", "p"),
                     fits = c("chisq", "rmsea"), file = file.path(tempdir, "fit1.t1"),
                   varLabels = c("x1" = "hello"), type = "latex", print.results = FALSE)
## If you have a working version of pdflatex in your system path,
if (interactive()) testtable("fit1.t1.tex", tempdir)

model <- "factor =~ .7*y1 + .7*y2 + .7*y3 + .7*y4
                 y1 | -1*t1 + 1*t2
                 y2 | -.5*t1 + 1*t2
                 y3 | -.2*t1 + 1*t2
                 y4 | -1*t1 + 1*t2"
dat <- simulateData(model, sample.nobs = 300)
```

```
testmodel <- "ExampleFactor =~ y1 + y2 + y3 + y4"

fit4 <- cfa(testmodel, data = dat, ordered = colnames(dat),
            std.lv = FALSE)

fit4.t1 <- semTable(fit4, paramSets = c("loadings", "thresholds",
    "residualvariances"), fits = c("tli", "chisq"),
    fitLabels = c(tli = "TLI", chisq = "chisq"), type = "html",
    file=file.path(tempdir, "fit4.t1") )
if(interactive()) browseURL(attr(fit4.t1, "file"))
fit4.t2 <- semTable(fit4, fits = c("rmsea", "cfi", "chisq"),
            fitLabels = c(rmsea = "Root M.SQ.E.A", cfi = "CompFitIdx", chisq = "chisq"),
              type = "latex", file=file.path(tempdir, "fit4.t2"))
if (interactive()) testtable("fit4.t2.tex", tempdir)
```

---

starsig	*How many stars would we need for this p value?*

Description

Regression table makers need to know how many stars to attach to parameter estimates. This takes p values and a vector which indicates how many stars are deserved. It returns a required number of asterixes. Was named "stars" in previous version, but renamed due to conflict with R base function `stars`

Usage

```
starsig(pval, alpha = c(0.05, 0.01, 0.001), symbols = c("*", "**",
  "***"))
```

Arguments

`pval`	P value
`alpha`	alpha vector, defaults as c(0.05, 0.01, 0.001).
`symbols`	The default is c("*", "**", "***"), corresponding to mean that p values smaller than 0.05 receive one star, values smaller than 0.01 get two stars, and so forth. Must be same number of elements as alpha. These need not be asterixes, could be any character strings that users desire. See example.

Details

Recently, we have requests for different symbols. Some people want a "+" symbol if the p value is smaller than 0.10 but greater than 0.05, while some want tiny smiley faces if p is smaller than 0.001. We accomodate that by allowing a user specified vector of symbols, which defaults to c("*", "**", "***")

Value

a character vector of symbols (eg asterixes), same length as pval

Author(s)

Paul Johnson <pauljohn@ku.edu>

Examples

```
starsig(0.06)
starsig(0.021)
starsig(0.001)
alpha.ex <- c(0.10, 0.05, 0.01, 0.001)
symb.ex <- c("+", "*", "**", ":)!")
starsig(0.07, alpha = alpha.ex, symbols = symb.ex)
starsig(0.04, alpha = alpha.ex, symbols = symb.ex)
starsig(0.009, alpha = alpha.ex, symbols = symb.ex)
starsig(0.0009, alpha = alpha.ex, symbols = symb.ex)
```

---

testtable   *Test viewer for tex tables*

Description

Creates a small latex template file that includes a table file. Compiles it, then displays in viewer if system has xdg-open settings.

Usage

```
testtable(tablefile, dir, tmpfn = "tmp.tex")
```

Arguments

tablefile	The base name of the table file
dir	Directory where table is saved, same will be used for build.
tmpfn	File name to be used by example document

Value

LaTeX log, returned from shell function.

Author(s)

Paul Johnson <pauljohn@ku.edu>

Examples

```
require(lavaan)
tempdir <- tempdir()
HS.model <- ' visual  =~ x1 + x2 + x3
              textual =~ x4 + x5 + x6
              speed   =~ x7 + x8 + x9'
fit1 <- cfa(HS.model, data = HolzingerSwineford1939,
            std.lv = TRUE, meanstructure = TRUE)
fit1.t <- semTable(fit1, fits = c("chisq", "rmsea"),
              columns = c("estsestars", "rsquare"),
              columnLabels = c("estsestars" = "Est(SE)"),
              file = file.path(tempdir, "fit1.t"))
if (interactive()) testtable("fit1.t", tempdir)
```

Index