Package 'slca'

April 22, 2024

Type Package

Title Structural Modeling for Multiple Latent Class Variables

Version 1.0.0

Maintainer Youngsun Kim <yskstat@gmail.com>

Description Provides comprehensive tools for the implementation of Structural Latent Class Models (SLCM), including
Latent Transition Analysis (LTA; Linda M. Collins and Stephanie T. Lanza, 2009) <doi:10.1002/9780470567333>,
Latent Class Profile Analysis (LCPA; Hwan Chung et al., 2010) <doi:10.1111/j.1467-985x.2010.00674.x>, and
Joint Latent Class Analysis (JLCA; Saebom Jeon et al., 2017) <doi:10.1080/10705511.2017.1340844>, and
any other extended models involving multiple latent class variables.

License GPL (>= 3)

Encoding UTF-8

LazyData true

RoxygenNote 7.3.1

LinkingTo Rcpp

Imports DiagrammeR, magrittr, MASS, Rcpp, stats

Depends R (>= 2.10)

NeedsCompilation yes

Author Youngsun Kim [aut, cre] (<https://orcid.org/0000-0001-8003-1939>),
Hwan Chung [aut] (<https://orcid.org/0000-0002-8969-9086>)

Repository CRAN

Date/Publication 2024-04-22 18:42:39 UTC

R topics documented:

`compare` *Comparing Two Estimated* `slca` *Models*

Description

Provides relative model fit test for two fitted SLCM models with deviance statistic.

Usage

```
compare(
  model1,
  model2,
  test = c("none", "chisq", "boot"),
  nboot = 100,
  method = c("hybrid", "em", "nlm"),
  maxiter = 1000,
  tol = 1e-08,
  verbose = FALSE
)
```

Arguments

`model1`	an object of class `slca` and `estimated`.
`model2`	another object of class `slca` and `estimated`.
`test`	a character string specifying the type of test to be conducted. If "chisq", a chi-squared test is conducted. If "boot", a bootstrap test is conducted.
`nboot`	an integer specifying the number of bootstrap rounds to be performed.
`method`	estimation method for bootstrapping.
`maxiter`	an integer specifying maximum number of iterations allowed for the estimation process of each bootstrapping round.
`tol`	a numeric value setting tolerance for the convergence of each bootstrapping round.
`verbose`	a logical value indicating whether to print progress updates on the number of bootstrapping rounds completed.

Value

A `data.frame` containing the number of parameters (Df), loglikelihood, AIC, BIC, G-squared statistics, and the residual degree of freedom for each object. Depending on the `test` argument, the p-value for the corresponding statistical test may also be included.

See Also

compare

Examples

```
library(magrittr)
data <- gss7677[gss7677$COHORT == "YOUNG", ]
stat2 <- slca(status(2) ~ PAPRES + PADEG + MADEG) %>%
   estimate(data = data)
stat3 <- slca(status(3) ~ PAPRES + PADEG + MADEG) %>%
   estimate(data = data)
stat4 <- slca(status(4) ~ PAPRES + PADEG + MADEG) %>%
   estimate(data = data)

gof(stat2, stat3, stat4)
gof(stat2, stat3, stat4, test = "chisq")

gof(stat2, stat3, stat4, test = "boot")


compare(stat3, stat4)
compare(stat3, stat4, test = "chisq")

compare(stat3, stat4, test = "boot")
```

---

confint.slca *Confidence Intervals for Model Parameters*

Description

Computes confidence intervals for one or more parameters of fitted model. Package **slca** adds methods for `slca` fits.

Usage

```
## S3 method for class 'slca'
confint(object, parm, level = 0.95, type = c("param", "logit"), ...)
```

Arguments

object	an object of class slca and estimated.
parm	an integer string specifying parameters to be given confidence intervals.
level	numeric value representing the desired confidence level for the intervals, with a default of 0.95.
type	a character string specifying the format in which the results should be returned. Options include "probs" for probability format and "logit" for log-odds (logit) format, with the default being "probs".
...	additional arguments.

Value

A matrix with two columns representing the confidence intervals for the selected parameters. The columns are named based on the specified confidence level (level):

- 100 * (level / 2) %: This column shows the lower bound of the confidence interval.
- 100 * (1 - level / 2) %: This column shows the upper bound of the confidence interval.

The level parameter specifies the confidence level, with common values being 0.05 for a 95% confidence interval and 0.01 for a 99% confidence interval.

Examples

```
param(nlsy_jlcpa, index = TRUE)
confint(nlsy_jlcpa)
confint(nlsy_jlcpa, 1:4)
```

---

estimate *Estimating Parameters of* slca *Object*

Description

Estimate the parameters of model constructed using the slca function.

Usage

```
estimate(x, ...)

## S3 method for class 'slca'
estimate(x, data,
    method = c("em", "hybrid", "nlm"),
    fix2zero = NULL,
    control = slcaControl(), ...)
```

Arguments

x	an slca object defining SLCM model to be estimated.
...	additional arguments.
data	a data.frame object containing observed categorical variables incorporated in the model.
method	estimation method for SLCM parameters. The default is "em", which employs expecation-maximization (EM) algorithm for estimation; the alternative "nlm", utilizes nlm function for Newton-Raphson algorithm. The "hybrid" method begins with the EM algorithm and concludes with the nlm function for refined estimation.
fix2zero	a vector of parameters to be restricted to zero. The details of restriction is given under 'Details'
control	a list of control for the estimation procedure. Used to modify default values in slcaControl.

Details

To constrain certain parameters to zero, use the fix2zero argument. Each parameter is associated with a unique index. You can identify the index of a specific parameter by invoking the param function with the index = TRUE argument. To apply these constraints, include the relevant parameter indices in the fix2zero argument.

Value

An object of class slca and estimated with an following elements:

model	a list describing of the model.
method	the method used for estimation
arg	the brief model description used during the estimation.
mf	the data.frame used for estimation.
par	the log of the estimated paramters.
logit	the log-odds of the estimated parameters.
score	the score function for the estimated parameters.
posterior	the list of posterior probablities for each latent class variable.
convergence	a logical indicator of whether convergence was achieved.
loglikelihood	the loglikelihood of the estimated model.
control	the control values used during the estimation process.

This returned object can be further processed using the param functions to extract the estimated parameters or their respective standard errors. Additionally, the regress function enables logistic regression analysis using three-step approach to evaluate the effect of external variables on latent class variables.

See Also

slca param regress slcaControl gss7677, nlsy97

gof *Goodness of Fit Tests for Estimated* slca *Model*

Description

Provides AIC, BIC and deviance statistic (G-squared) for goodness of fit test for the fitted model. Absolute model fit can be tested with deviance statistics, if `test` argument is specified.

Usage

```
gof(object, ...)

## S3 method for class 'slca'
gof(
   object, ..., test = c("none", "chisq", "boot"), nboot = 100,
   maxiter = 100, tol = 1e-6, verbose = FALSE
)
```

Arguments

`object`	an object of class `slca` and `estimated`.
`...`	additional objects of class `slca` and `estimated`.
`test`	a character string specifying the type of test to be conducted. If "chisq", a chi-squared test is conducted. If "boot", a bootstrap test is conducted.
`nboot`	an integer specifying the number of bootstrap rounds to be performed.
`maxiter`	an integer specifying maximum number of iterations allowed for the estimation process of each bootstrapping round.
`tol`	a numeric value setting tolerance for the convergence of each bootstrapping round.
`verbose`	a logical value indicating whether to print progress updates on the number of bootstrapping rounds completed.

Value

A `data.frame` containing the number of parameters (Df), loglikelihood, AIC, BIC, G-squared statistics, and the residual degree of freedom for each object. Depending on the `test` argument, the p-value for the corresponding statistical test may also be included.

See Also

compare

Examples

```
library(magrittr)
data <- gss7677[gss7677$COHORT == "YOUNG", ]
stat2 <- slca(status(2) ~ PAPRES + PADEG + MADEG) %>%
   estimate(data = data)
stat3 <- slca(status(3) ~ PAPRES + PADEG + MADEG) %>%
   estimate(data = data)
stat4 <- slca(status(4) ~ PAPRES + PADEG + MADEG) %>%
   estimate(data = data)

gof(stat2, stat3, stat4)
gof(stat2, stat3, stat4, test = "chisq")

gof(stat2, stat3, stat4, test = "boot")


compare(stat3, stat4)
compare(stat3, stat4, test = "chisq")

compare(stat3, stat4, test = "boot")
```

---

gss7677 *GSS 1976-1977 Data on Social Status and Tolerance towards Minorities*

Description

This dataset contains responses from the General Social Survey (GSS) for the years 1976 and 1977, focusing on social status and tolerance towards minorities The latent class models can be fitted using this dataset replicate the analysis carried on McCutcheon (1985) and Bakk et al. (2014).
The data contains some covariates including year of the interview, age, sex, race, degree, and income of respondents. The variables associating social status include father's occupation and education level, and mother's education level, while the variables associating tolerance towards minorities are created by agreeing three related questions: (1) allowing public speaking, (2) allowing teaching, and (3) allowing literatures.

Format

A data frame with 2942 rows and 14 variables:

`YEAR` Interview year (1976, 1977)

`COHORT` Respondent's age
levels: (1)`YOUNG`, (2)`YOUNG-MIDDLE`, (4)`MIDDLE`, (5)`OLD`

`SEX` Respondent's sex
levels: (1)`MALE`, (2)`FEMALE`

`RACE` Respondent's race
levels: (1)`WHITE` (2)`BLACK`, (3)`OTHER`

DEGREE Respondent's degree
levels: (1)LT HS, (2)HIGH-SCH, (3)COLLEGE, (4) BACHELOR, (5)GRADUATE

REALRINC Income of respondents

PAPRES Father's prestige (occupation)
levels: (1)LOW, (2)MIDIUM, (2)HIGH

PADEG Father's degree
levels: (1)LT HS, (2)HIGH-SCH, (3)COLLEGE, (4) BACHELOR, (5)GRADUATE

MADEG Mother's degree
levels: (1)LT HS, (2)HIGH-SCH, (3)COLLEGE, (4) BACHELOR, (5)GRADUATE

TOLRAC Tolerance towards racists

TOLCOM Tolerance towards communists

TOLHOMO Tolerance towards homosexuals

TOLATH Tolerance towards atheists

TOLMIL Tolerance towards militarists

Source

General Social Survey (GSS) 1976, 1977

References

Bakk Z, Kuha J. (2021) Relating latent class membership to external variables: An overview. Br J Math Stat Psychol. 74(2):340-362.

McCutcheon, A. L. (1985). A latent class analysis of tolerance for nonconformity in the American public. Public Opinion Quarterly, 49, 474–488.

Examples

```
library(magrittr)
data <- gss7677[gss7677$RACE == "BLACK",]
model_stat <- slca(status(3) ~ PAPRES + PADEG + MADEG) %>%
   estimate(data = data)
summary(model_stat)
param(model_stat)

model_tol <- slca(tol(4) ~ TOLRAC + TOLCOM + TOLHOMO + TOLATH + TOLMIL) %>%
   estimate(data = data)
summary(model_tol)
param(model_tol)

model_lta <- slca(
   status(3) ~ PAPRES + PADEG + MADEG,
   tol(4) ~ TOLRAC + TOLCOM + TOLHOMO + TOLATH + TOLMIL,
   status ~ tol
) %>% estimate(data = data)
summary(model_lta)
param(model_lta)
```

```
regress(model_lta, status ~ SEX, data)

regress(model_lta, status ~ SEX, data, method = "BCH")
regress(model_lta, status ~ SEX, data, method = "ML")
```

`nlsy97` *NLSY97 Substance Use Data*

Description

A dataset containing substance use behavior from the National Longitudinal Survey of Youth 1997 (NLSY97) for three years: 1998, 2003, and 2008. The dataset focuses on the youth born in 1984 and tracks three substance use behaviors: tobacco/cigarette smoking, alcohol drinking, and marijuana use.

Format

A data frame with 1004 rows and 38 columns:

SEX Respondent's sex

RACE Respondent's race

ESMK_98, ESMK_03, ESMK_08 (Ever smoked) Ever smoked in 1998, 2003, and 2008 (0: No, 1: Yes)

FSMK_98, FSMK_03, FSMK_08 (Frequent smoke) Monthly smokes in 1998, 2003, and 2008 (0: No, 1: Yes)

DSMK_98, DSMK_03, DSMK_08 (Daily smoke) Daily smokes in 1998, 2003, and 2008 (0: No, 1: Yes)

HSMK_98, HSMK_03, HSMK_08 (Heavy smoke) 10+ cigarettes per day in 1998, 2003, and 2008 (0: No, 1: Yes)

EDRK_98, EDRK_03, EDRK_08 (Ever drunk) Have you ever drunk in 1998, 2003, and 2008? (0: No, 1: Yes)

CDRK_98, CDRK_03, CDRK_08 (Current drinker) Monthly drinking in 1998, 2003, and 2008 (0: No, 1: Yes)

WDRK_98, WDRK_03, WDRK_08 (Weakly drinker) 5+ days drinking in a month in 1998, 2003, and 2008 (0: No, 1: Yes)

BDRK_98, BDRK_03, BDRK_08 (Binge drinker) 5+ drinks on the same day at least one time in the last 30 day (0: No, 1: Yes)

EMRJ_98, EMRJ_03, EMRJ_08 (Ever marijuana used) Have you ever used marijuana in 1998, 2003, and 2008? (0: No, 1: Yes)

CMRJ_98, CMRJ_03, CMRJ_08 (Corrent marijuana user) Monthly marijuana use in 1998, 2003, and 2008 (0: No, 1: Yes)

OMRJ_98, OMRJ_03, OMRJ_08 (Occasional marijuana user) 10+ days marijuana use in a month in 1998, 2003, and 2008 (0: No, 1: Yes)

SMRJ_98, SMRJ_03, SMRJ_08 (School/work marijuana user) Marijuana use before/during school or work in 1998, 2003, and 2008 (0: No, 1: Yes)

Similar naming conventions apply for the years 2003 and 2008, replacing '98' with '03' and '08', respectively.

Source

National Longitudinal Survey of Youth 1997 (NLSY97)

References

Bureau of Labor Statistics, U.S. Department of Labor. National Longitudinal Survey of Youth 1997 cohort, 1997-2017 (rounds 1-18). Produced and distributed by the Center for Human Resource Research (CHRR), The Ohio State University. Columbus, OH: 2019.

Examples

```
library(magrittr)
nlsy_smoke <- slca(smk98(3) ~ ESMK_98 + FSMK_98 + DSMK_98 + HSMK_98) %>%
   estimate(data = nlsy97)
summary(nlsy_smoke)


# JLCA
model_jlca <- slca(
   smk98(3) ~ ESMK_98 + FSMK_98 + DSMK_98 + HSMK_98,
   drk98(3) ~ EDRK_98 + CDRK_98 + WDRK_98 + BDRK_98,
   mrj98(3) ~ EMRJ_98 + CMRJ_98 + OMRJ_98 + SMRJ_98,
   substance(4) ~ smk98 + drk98 + mrj98
) %>% estimate(data = nlsy97)
summary(model_jlca)
param(model_jlca)

# JLCPA
nlsy_jlcpa <- slca(
   smk98(3) ~ ESMK_98 + FSMK_98 + DSMK_98 + HSMK_98,
   drk98(3) ~ EDRK_98 + CDRK_98 + WDRK_98 + BDRK_98,
   mrj98(3) ~ EMRJ_98 + CMRJ_98 + OMRJ_98 + SMRJ_98,
   use98(5) ~ smk98 + drk98 + mrj98,
   smk03(3) ~ ESMK_03 + FSMK_03 + DSMK_03 + HSMK_03,
   drk03(3) ~ EDRK_03 + CDRK_03 + WDRK_03 + BDRK_03,
   mrj03(3) ~ EMRJ_03 + CMRJ_03 + OMRJ_03 + SMRJ_03,
   use03(5) ~ smk03 + drk03 + mrj03,
   smk08(3) ~ ESMK_08 + FSMK_08 + DSMK_08 + HSMK_08,
   drk08(3) ~ EDRK_08 + CDRK_08 + WDRK_08 + BDRK_08,
   mrj08(3) ~ EMRJ_08 + CMRJ_08 + OMRJ_08 + SMRJ_08,
   use08(5) ~ smk08 + drk08 + mrj08,
   prof(4) ~ use98 + use03 + use08,
   constraints = list(
      c("smk98", "smk03", "smk08"),
      c("drk98", "drk03", "drk08"),
```

```
        c("mrj98", "mrj03", "mrj08"),
        c("use98 ~ smk98", "use03 ~ smk03", "use08 ~ smk08"),
        c("use98 ~ drk98", "use03 ~ drk03", "use08 ~ drk08"),
        c("use98 ~ mrj98", "use03 ~ mrj03", "use08 ~ mrj08")
    )
) %>% estimate(nlsy97)
```

`nlsy_jlcpa` *JLCPA Model Estimated with NLSY97 Data*

Description

The `slca` model estimated using NLSY97 data

Format

A data frame with 1004 rows and 38 columns:

Source

National Longitudinal Survey of Youth 1997 (NLSY97)

References

Bureau of Labor Statistics, U.S. Department of Labor. National Longitudinal Survey of Youth 1997 cohort, 1997-2017 (rounds 1-18). Produced and distributed by the Center for Human Resource Research (CHRR), The Ohio State University. Columbus, OH: 2019.

Jeon, S., Seo, T. S., Anthony, J. C., & Chung, H. (2022). Latent Class Analysis for Repeatedly Measured Multiple Latent Class Variables. Multivariate Behavioral Research, 57(2–3), 341–355.

See Also

reorder.slca

`param` *Printing Estimated Parameters of* `slca` *Object*

Description

This function prints the estimated parameters of the `slca` model by accepting an `estimated slca` object.

Usage

```
param(object, ...)

## S3 method for class 'slca'
param(
   object, type = c("probs", "logit"),
   se = FALSE, index = FALSE, ...
)
```

Arguments

object	an object of class slca and estimated.
...	additional arguments.
type	a character string specifying the format in which the estimated parameters should be displayed. The options are "probs" for probability format or "logit" for log-odds (logit) format. The default setting is "probs".
se	a logical indicating whether standard errors (TRUE) or parameter estimates (FALSE) should be displayed.
index	a logical indicating whether to include (TRUE) or exclude (FALSE) the indices of the estimated parameters in the output.

Value

A list containing the specified estimated parameters or their standard errors if se is set to TRUE. The components of the list include:

pi	Membership probabilities of the root variable.
tau	Conditional probabilities between latent class variables, represented with uppercase alphabets for considering measurement invariance.
rho	Item response probabilities for each measurement model, represented with lowercase alphabets for considering measurement invariance.

---

regress *Regress Exogenous Variables on Latent Variables*

Description

This function performs regression analysis to explore the influence of exogenous (external) variables on the latent class variables within an estimated slca model. It utilizes logistic regression and employs a three-step approach.

Usage

```
regress(object, ...)

## S3 method for class 'slca'
regress(
   object, formula, data = parent.frame(),
   imputation = c("modal", "prob"),
   method = c("naive", "BCH", "ML"), ...
)
```

Arguments

object	an object of class slca and estimated
...	additional arguments.
formula	a formula defining the regression model, including both latent class variables from the estimated model and any exogenous (external) variables.
data	an optional data frame containing the exogenous variables of interest.
imputation	the imputation method for imputing (assigning) latent class variables. Possible values are: • "modal": Assigns each individual to the latent class with the highest posterior probability. • "prob": Assigns classes to individuals randomly according to the distribution of posterior probabilities.
method	the method used to adjust bias in the three-step approach, with options including "naive", "BCH", and "ML".

Value

A list with following components:

coefficients	a matrix of regression coefficients representing the odds ratios of each class against the baseline class (the last class).
std.err	a matrix of standard errors corresponding to the regression coefficients.
vcov	the calculated variance-covariance matrix for the regression coefficients.
dim	the dimensions of the coefficients matrix.
ll	the log likelihood of the regression model.

Using the summary function, you can print coefficients, standard errors, corresponding Wald statistics, and p-values for these statistics.

References

Vermunt, J. K. (2010). Latent Class Modeling with Covariates: Two Improved Three-Step Approaches. Political Analysis, 18(4), 450–469. http://www.jstor.org/stable/25792024

Examples

```
library(magrittr)
names(nlsy97)
nlsy_jlcpa %>% regress(smk98 ~ SEX, nlsy97)
nlsy_jlcpa %>% regress(drk98 ~ SEX, nlsy97)
nlsy_jlcpa %>% regress(mrj98 ~ SEX, nlsy97)

nlsy_jlcpa %>% regress(use98 ~ SEX, nlsy97)
nlsy_jlcpa %>% regress(prof ~ SEX, nlsy97)
```

`reorder.slca` *Reorder Latent Class Membership of Class Variables*

Description

This function reorders the latent class membership for specified latent class variables.

Usage

```
## S3 method for class 'slca'
reorder(x, ...)
```

Arguments

`x`	an object of class `slca` and `estimated`.
`...`	additional arguments specifying the new order for the latent class variables.

Value

Returns the modified `slca` or `estimated` object with the reordered latent classes.

Examples

```
library(magrittr)
nlsy_jlcpa %>% param

# Reorder the RHO parameters as ascending order
reordered1 <- nlsy_jlcpa %>%
   reorder(smk98 = c(2, 3, 1),
           drk98 = c(2, 1, 3),
           mrj98 = c(3, 2, 1))
reordered1 %>% param
# Label class1: nonuse
#       class2: lifetime use
#       class3: current use

# Reorder the TAU parameters for joint classes as ascending order
reordered2 <- reordered1 %>%
```

```
   reorder(use98 = c(5, 1, 4, 2, 3))
reordered2 %>% param
# Label class1: nonuse
#       class2: heavy drinking only
#       class3: not heavy use
#       class4: heavy drinking & smoking
#       class5: heavy use

# Reorder the TAU paramters for profiles as ascending order
reordered3 <- reordered2 %>%
   reorder(prof = c(4, 1, 3, 2))
reordered3 %>% param
# Label class1: nonuse stayer
#       class2: heavy drinking advancer
#       class3: heavy drk & smk advancer
#       class4: heavy use advancer
```

`simulate.slca` *Simulate data from* `slca` *model.*

Description

This function simulates data from a constructed `slca` model. If the model is not already estimated, parameters can be specified by the user or generated randomly.

Usage

```
## S3 method for class 'slca'
simulate(object, nsim = 500, seed = NULL, parm, nlevel, ...)
```

Arguments

`object`	a `slca` object representing the model from which data will be simulated.
`nsim`	the number of response observations to simulate. Defaults to 500.
`seed`	a random seed for reproducibility of the
`parm`	a set of parameters provided by the user to guide the simulation, if the model has not been estimated.
`nlevel`	the number of levels for each manifest item declared in the model. If not provided, the default is 2.
`...`	additional arguments.

Value

A `list` of two components:

`class`	A `data.frame` providing the assigned latent class for each individual across different latent class variables.
`response`	A `data.frame` containing the manifest items that were simulated.

Examples

```
library(magrittr)
model <- slca(lc1[3] ~ x1 + x2 + x3 + x4 + x5)
sim_data <- model %>% simulate(nlevel = c(3, 3, 3, 3, 3))
y <- sim_data$response
sapply(y, table)

sim_data <- model %>%
   simulate(nlevel = c(x1 = 2, x3 = 3, x4 = 4, x5 = 5))
y <- sim_data$response
sapply(y, table)

model <- slca(lc1[3] ~ x1 + x2 + x3 + x4 + x5,
              lc2[4] ~ y1 + y2 + y3 + y4 + y5)
sim_data <- model %>% simulate(1000)
sapply(sim_data$class, table)
```

---

slca *Construct Structural Latent Class Model*

Description

This function constructs a structural latent class model (SLCM) for specified latent class variables.

Usage

```
slca(formula = NULL, ..., constraints = NULL)
```

Arguments

formula	a formula specifying the latent structure. Detailed model specification is provided under 'Details'.
...	additional formulae for defining the model structure.
constraints	a list of constraints for maintaining measurement invariance. Detailed explanation of applying constraints is available under 'Details'.

Details

The formula can be broadly categorized into three main types, each serving a distinct purpose:

1. **Defining Latent Class Variables with Manifest Indicators**: Specify the relationship between a latent class variable and its manifest indicators. For example:
2. **Defining Latent Class Variables with Manifest Indicators**: Specify the relationship between a latent class variable and its manifest indicators. In these formulas, the latent class variable, denoted with square brackets or parentheses indicating the number of classes, is on the left-hand-side (lhs) and its manifest indicators are specified on right-hand-side (rhs). For example,

```
LC1[k] ~ x1 + x2 + x3
LC2[k] ~ y1 + y2 + y3
LC3(k) ~ z1 + z2 + z3
```

In these formulas, k denotes the number of latent classes for the variable.

3. **Relating Latent Class Variables to Each Other**: Define relationships where one latent class variable is influenced by another. The subsequent example implies that LC2 is conditionally affected based on LC1.

   ```
   LC1 ~ LC2
   ```

4. **Defining higher-level latent class variable**: Identify latent class variables by other latent class variables rather than manifest indicators. Following example suggests that the P is measured by LC1, LC2, and LC3 – all of which are latent class variables.

   ```
   P[k] ~ LC1 + LC2 + LC3
   ```

In all types of the formula, variables specified on the left-hand side (lhs) influence those on the right-hand side (rhs).

The constraints parameter allows you to enforce specific conditions on the model to ensure precise inference. For instance, in Longitudinal Latent Class Analysis (LTA), it's imperative that latent class variables across various time-points convey identical meanings. With the constraints option, users can uphold measurement invariance in both the measurement and structural components of the model.

1. **Measurement Invariance for Measurement Model**: Ensures that probabilities associated with latent class variables are consistent, thus maintaining semantic meaning across classes.

   ```
   c("LC1", "LC2", "LC3")
   ```

   This command ensures that variables LC1, LC2, and LC3 are semantically consistent.

2. **Measurement invariance for structural model**: Applies constraints within the structural model to ensure consistent interpretations of transition probabilities.

   ```
   c("P ~ LC1", "P -> LC2")
   ```

   This command implies that the transition probabilities from P to LC1 and from P to LC2 are consistent.

Value

An object of class slca containing various components of the model:

- tree: A data.frame that details the parent-child relationships among latent class and manifest variables.
- latent: A data.frame listing all latent class variables with details on each.
- measure: A data.frame that describes the measurement model.
- struct: A data.frame that details the structural model.

The object prints model description with four part.

1. Latent variables: lists the latent class variables incorporated in the model, along with the number of classes for each variable. The root variable is marked by asterisk (*).

2. Measurement model: Shows manifest indicators for each latent class variable and indicates measurement constraints (lowercase letters signify consistency).
3. Structural model: Describes conditional dependencies between latent class variables.
4. Dependency constraints: Outlines the constraints applied to the conditional dependencies between latent class variables. Each column marked with an uppercase alphabet symbolizes a consistent dependency structure.

Examples

```
# Standard LCA
slca(lc[3] ~ y1 + y2 + y3)
# Latent transition analysis (LTA)
slca(l1[3] ~ y11 + y21 + y31,
     l2[3] ~ y12 + y22 + y32,
     l1 ~ l2)
# LTA with measurement invariance
slca(l1[3] ~ y11 + y21 + y31,
     l2[3] ~ y12 + y22 + y32,
     l1 ~ l2, constraints = c("l1", "l2"))
# Joint latent class analysis
slca(lx[3] ~ x1 + x2 + x3, ly[3] ~ y1 + y2 + y3,
     lz[3] ~ z1 + z2 + z3, jc[3] ~ lx + ly + lz)
# Latent class profile analysis (with measurement invariance)
slca(l1[3] ~ x1 + x2 + x3, l2[3] ~ y1 + y2 + y3,
     l3[3] ~ z1 + z2 + z3, pf[3] ~ lx + ly + lz,
     constraints = c("l1", "l2", "l3"))
```

---

slcaControl *Control Parameters for slca Estimation*

Description

Control Parameters for slca Estimation

Usage

```
slcaControl(
  em.iterlim = 3000,
  em.tol = 1e-06,
  nlm.iterlim = 1000,
  nlm.tol = 1e-08,
  init.param = NULL,
  nrep = 1,
  test.iter = 200,
  na.rm = FALSE,
  verbose = TRUE
)
```

Arguments

em.iterlim	an integer specifying maximum number of iterations allowed for EM algorithm. Default is 3000.
em.tol	a numeric value setting tolerance for the convergence of EM algorithm. Default is 1e-6.
nlm.iterlim	an integer specifying maximum number of iterations allowed for estimation with nlm function. Default is 1000.
nlm.tol	a numeric value setting tolerance for the convergence of nlm function. Default is 1e-8.
init.param	initial parameters.
nrep	number of trials. Default is 1.
test.iter	an integer specifying maximum number of iterations allowed for testing parameters. Default is 200
na.rm	a logical value indicating whether to remove observations including missing values (NA)
verbose	a logical value indicating whether to display progress updates during the estimation process

Value

a list with control parameters for slca estimation.

See Also

slca

Index