Package 'relabeLoadings'

November 21, 2016

Version 1.0

Date 2016-11-20

Title Relabel Loadings from MCMC Output for Confirmatory Factor Analysis

Description In confirmatory factor analysis (CFA), structural constraints typically ensure that the model is identified up to all possible reflections, i.e., column sign changes of the matrix of loadings. Such reflection invariance is problematic for Bayesian CFA when the reflection modes are not well separated in the posterior distribution. Imposing rotational constraints -- fixing some loadings to be zero or positive in order to pick a factor solution that corresponds to one reflection mode -- may not provide a satisfactory solution for Bayesian CFA. The function 'relabel' uses the relabeling algorithm of Erosheva and Curtis to correct for sign invariance in MCMC draws from CFA models. The MCMC draws should come from Bayesian CFA models that are fit without rotational constraints.

License GPL (>= 2)

RoxygenNote 5.0.1

NeedsCompilation no

Author S. McKay Curtis [aut, cre], Elena A. Erosheva [aut]

Maintainer S. McKay Curtis <s.mckay.curtis@gmail.com>

Repository CRAN

Date/Publication 2016-11-21 21:22:46

R topics documented:

relabel *Relabel Factor Loadings from MCMC output*

Description

Uses the relabeling method of Erosheva and Curtis to correct for sign invariance in MCMC draws from confirmatory factor analysis (CFA) models.

Usage

```
relabel(obj, ...)

## Default S3 method:
relabel(obj, ...)

## S3 method for class 'matrix'
relabel(obj, random = FALSE, max.iter = 100, ...)
```

Arguments

obj	An object of the appropriate class. Currently, the method is implemented only for objects of S3 class `matrix`. The matrix should contain the MCMC draws from a Bayesian CFA model where no arbitrary restrictions are imposed on the loadings to enforce rotational invariance. For example, no loadings should have been arbitrarily constrained to be one. Columns of the matrix must have names of the form "Lam[1,1]", "Lam[1,2]", ..., where indices denote the structure of the loading matrix, i.e., missing idices indicate loadings that are constrained to be zero.
random	Logical value. If `TRUE`, random starting points will be used. If `FALSE`, starting values are chosen roughly to allow the largest loadings on each factor to have a positive sign or to allow the majority of loadings on a given factor to have a positive sign.
max.iter	Integer that specifies the maximum number of iterations of the algorithm before aborting procedure.
...	Not used.

Details

See the references.

Value

An S3 object of class `relabel` with the following components:

`Lam`	original (untransformed) draws of the loadings.
`nuLam`	transformed draws of the loadings.
`nu.init`	initial values for the sign-change parameters.
`nu`	final values of the sign change parameters used to compute `nuLam`.
`m`	final values of the mean parameters in the loss function.
`s`	final values of the standard deviation parameters in the loss function.
`factor.idx`	vector of integers indicating which columns of the posterior draws belong to which factor.
`iter`	number of iterations the algorithm took to converge.
`loss`	final value of the loss function after convergence.
`converged`	logical indicating whether the algorithm converged.

Author(s)

S. McKay Curtis and Elena Erosheva

References

Erosheva, E. A. and Curtis, S. M. (2011) "A relabeling scheme for confirmatory factor analysis." Technical report #589. University of Washington, Dept. of Statistics.

Examples

```
n <- 1000
p <- 8
set.seed(1)
mu <- as.numeric(t(cbind(
    matrix(rep(sample(c(-1, 1), size=n, replace=TRUE)*4, each=8), n, 8, byrow=TRUE),
    matrix(rep(sample(c(-1, 1), size=n, replace=TRUE)*4, each=8), n, 4, byrow=TRUE),
    matrix(rep(sample(c(-1, 1), size=n, replace=TRUE)*4, each=8), n, 4, byrow=TRUE))))
Lam <- matrix(rnorm(length(mu), mu, 1.0), n, 16, byrow=TRUE)
colnames(Lam) <- c(paste0("Lam[", 1:p, ",", 1, "]"),
                   paste0("Lam[", 1:4, ",", 2, "]"),
                   paste0("Lam[", 5:8, ",", 3, "]"))

par(mfrow=c(2, 8))
apply(Lam, 2, function(x) plot(density(x)))

## Relabeling removes the bimodality
out <- relabel(Lam)

par(mfrow=c(2, 8))
apply(out$nuLam, 2, function(x) plot(density(x)))
```

Index