Package 'hlt'

May 8, 2026

Type Package

Title Higher-Order Item Response Theory

Version 1.3.1

Date 2022-08-14

Maintainer Michael Kleinsasser <mjkleinsa@gmail.com>

Description Higher-order latent trait theory (item response theory). We implement the generalized partial credit model with a second-order latent trait structure. Latent regression can be done on the second-order latent trait. For a pre-print of the methods, see, ``Latent Regression in Higher-Order Item Response Theory with the R Package hlt'' <https://mkleinsa.github.io/doc/hlt_proof_draft_brmic.pdf>.

License GPL (>= 2)

Depends R (>= 3.5.0)

Imports Rcpp (>= 1.0.8), RcppDist, RcppProgress, tidyr, ggplot2, truncnorm, foreach, doParallel

LinkingTo Rcpp, RcppDist, RcppProgress

RoxygenNote 7.2.1

Encoding UTF-8

URL https://github.com/mkleinsa/hlt

BugReports https://github.com/mkleinsa/hlt/issues

LazyData true

NeedsCompilation yes

Author Michael Kleinsasser [aut, cre]

Repository CRAN

Date/Publication 2022-08-22 10:30:02 UTC

Contents

`asti` *asti data*

Description

asti data

References

Levenson, M. R., Jennings, P. A., Aldwin, C. M., & Shiraishi, R. W. (2005). Self-transcendence: conceptualization and measurement. The International Journal of Aging and Human Development, 60, 127-143.

`get_hlt_start` *hlt Starting Values*

Description

Get starting values from hlt fit object

Usage

```
get_hlt_start(x, nchains = 1)
```

Arguments

`x`	hlt model fit object
`nchains`	number of chains to get starting values

Value

a list of lists with starting values for each chain hlt *Explanatory and Descriptive Higher-Order Item Response Theory (Latent Trait Theory)*

Description

Fit a higher-order item response theory model under the generalized partial credit measurement model. The goal is to explain multiple latent dimensions by a single higher-order dimension. We extend this model with an option to perform regression on the general latent dimension.

Usage

```
hlt(
  x,
  z = NULL,
  id,
  iter,
  burn = iter/2,
  delta,
  type = "2p",
 start = list(list(lambda = c(), theta = c(), delta = c(), alpha = c(), beta = c())),
  nchains = 1,
  progress = TRUE,
  verbose = FALSE
)
```

Arguments

`x`	matrix of item responses. Responses must be integers where the lowest value is 0 and the highest value is the maximum possible response for the item with no gaps. If a question is asked with 5 possible responses, then the possible values should be c(0,1,2,3,4). For binary items, use c(0,1).
`z`	centered numeric matrix of predictors for the latent regression. Default is ‘z = NULL‘ so that no regression is performed. All columns of this matrix must be numeric. For binary items, use the values c(0,1). For continuous items, center the values on the mean and divide by the standard deviation (normalized). For factors with more than two levels, recode into multiple columns of c(0,1).
`id`	I.D. vector indexing first-order latent dimension membership for each of the first-order latent dimensions. We index starting from zero, not one. If there are three first-order . latent dimensions with 5 questions per dimension, then the vector will look like c(0,0,0,0,0,1,1,1,1,1,2,2,2,2,2).
`iter`	number of total iterations.
`burn`	number of burn in iterations.
`delta`	tuning parameter for Metropolis-Hanstings algorithm. Alter delta until acceptance.ratio =~ 0.234.

`type`	type of Partial Credit Model to fit. If the partial credit model is desired (i.e. all alpha parameters = 1), then choose ‘type = "1p"‘. If the Generalized Parial Credit Model is desired, then choose ‘type = "2p"‘. The default is ‘type = "2p"‘.
`start`	starting values for the Metropolis-Hastings algorithm.
`nchains`	number of independent MCMC chains. Default is ‘nchains = 1‘.
`progress`	boolean, show progress bar? Defaults to TRUE.
`verbose`	print verbose messages. Defaults to ‘FALSE‘. Provide a ‘list‘ with the following named arguments: ‘list(lambda = c(), theta = c(), delta = c(), alpha = c(), beta = c())‘ • lambda - vector of starting values for the latent factor loadings. • theta - vector of starting values for the abilities. • delta - vector of starting values for the difficulties. • alpha - vector of starting values for the slope parameters. • beta - vector of starting values for the latent regression parameters If you choose specify starting values, then the lengths of the starting value vectors must match the number of parameters in the model.

Value

A ‘list‘ containing:

- post - A ‘matrix‘ of posterior estimates. Rows are the draws and columns are the named parameters.
- accept.rate - acceptance rate of MH algorithm
- theta - ‘matrix‘ of mean (first column) and standard deviation (second column) of posterior estimates of ability parameters
- nT - number of latent factors estimated
- args - returns the arguments to hlt

Examples

```
# set seed
set.seed(153)

# load the asti data set
data("asti")

# shift responses to range from 0 instead of 1
x = as.matrix(asti[, 1:25]) - 1

# subset and transform predictor data
z = asti[, 26:27]
z[, 1] = (z[, 1] == "students") * 1
z[, 2] = (z[, 2] == "male") * 1
z = as.matrix(z)

# specify which items from x belong to each domain
```

```
id = c(0,0,0,0,1,1,1,1,2,2,2,2,3,3,3,3,3,3,3,4,4,4,4,4,4)

# fit the model
mod = hlt(x, z = z, id = id, iter = 20, burn = 10, delta = 0.002)

mod$accept.rate # ideally 0.234


plot(mod, param = "lambda1", type = "trace")
plot(mod, param = "lambda2", type = "trace")
plot(mod, param = "lambda3", type = "trace")
plot(mod, param = "a1", type = "trace")
plot(mod, param = "d2_3", type = "trace")
plot(mod, param = "beta1", type = "trace")

plot(mod, item = 1, type = "icc")
plot(mod, item = 2, type = "icc")
plot(mod, item = 3, type = "icc")
plot(mod, item = 4, type = "icc")
plot(mod, item = 5, type = "icc")
plot(mod, item = 6, type = "icc")
plot(mod, item = 7, type = "icc", min = -10, max = 10)

summary(mod, param = "all")
summary(mod, param = "delta", digits = 2)
summary(mod, param = "lambda")
summary(mod, param = "alpha")
summary(mod, param = "delta")
summary(mod, param = "theta", dimension = 1)
summary(mod, param = "theta", dimension = 2)
summary(mod, param = "theta", dimension = 3)
summary(mod, param = "theta", dimension = 4)

# start from a previous run's solution
post = tail(mod$post, 1)
start = list(lambda = post[1, c("lambda1", "lambda2", "lambda3", "lambda4", "lambda5")],
             theta = mod$theta_mean,
             delta = post[1, grepl("^[d]", colnames(post))],
             alpha = post[1, paste0("a", 1:25)],
             beta = post[1, c("beta1", "beta2")])

mod = hlt(x, z = z, id = id, start = start, iter = 20, burn = 10, delta = 0.002)
```

`hltsim` *Simulate the HLT model*

Description

Simulate the HLT model

Usage

```
hltsim(type, n, ntheta, lambda, id, dL, nB, beta = NULL)
```

Arguments

`type`	type of model to simulate. 'type = "1p"' for the partial credit model. 'type = "2p"' for the generalized partial credit model.
`n`	number of observations
`ntheta`	number first-level of latent dimensions
`lambda`	latent factor coefficients
`id`	number of questions
`dL`	number of levels for each question
`nB`	number of regression parameters. nB = ncol(z).
`beta`	what value to set the regression parameters.

Value

a 'list' containing

- x - matrix of simulated item responses
- theta - matrix of true latent ability
- id - I.Ds for item membership to each dimension
- namesx - vector of column names for each item
- s.cor - true correlations between latent ability dimensions
- s.delta - true difficulty parameters
- s.lambda - true loading parameters
- s.alpha - true discrimination parameters

Examples

```
xdat = hltsim(n = 250, type = "2p", ntheta = 4,
              lambda = c(0.5, 0.8, 0.9, 0.4), id = c(rep(0, 15),
              rep(1, 15), rep(2, 15), rep(3, 15)), dL = 2)
mod1 = hlt(x = xdat$x, id = xdat$id, iter = 12e1,
           burn = 6e1, delta = 0.023)

xdat = hltsim(n = 250, type = "2p", ntheta = 4,
              lambda = c(0.5, 0.8, 0.9, 0.4), id = c(rep(0, 15),
              rep(1, 15), rep(2, 15), rep(3, 15)), dL = 2,
              beta = c(0.5, -0.7), nB = 2)
mod2 = hlt(x = xdat$x, id = xdat$id, z = xdat$z,
           iter = 12e1, burn = 6e1, delta = 0.023, nchains = 1)
```

`merge_chains` *Merge Chains from hlt method*

Description

Merge Chains from hlt method

Usage

```
merge_chains(x, ...)
```

Arguments

`x`	object of class "hltObjList"
`...`	other arguments

Value

a list of class ‘hltObj‘. This class constructs a single ‘hltObj‘ from a list of model fits by merging the chains into one matrix of draws.

Index