Package ‘ohoegdm’

October 14, 2022

Title Ordinal Higher-Order Exploratory General Diagnostic Model for Polytomous Data

Version 0.1.0


URL https://github.com/tmsalab/ohoegdm,
    https://tmsalab.github.io/ohoegdm/

BugReports https://github.com/tmsalab/ohoegdm/issues

License GPL (>= 2)

Encoding UTF-8

RoxygenNote 7.1.2

LinkingTo Rcpp, RcppArmadillo

Imports Rcpp

Suggests edmdata, covr

NeedsCompilation yes

Author Steven Andrew Culpepper [aut, cph]
    (<https://orcid.org/0000-0003-4226-6176>),
    James Joseph Balamuta [aut, cre, cph]
    (<https://orcid.org/0000-0003-2826-8458>)

Maintainer James Joseph Balamuta <balamut2@illinois.edu>

Repository CRAN

Date/Publication 2022-02-24 09:30:02 UTC

R topics documented:

  GenerateAtable .................................................. 2
  gen_bijectionvector ............................................. 2
  ohoegdm .......................................................... 3
  sim_slcm .......................................................... 7
GenerateAtable

Generate tables that store different design elements

Description

Each table provides a "cache" of pre-computed values.

Usage

GenerateAtable(nClass, K, M, order)

Arguments

- nClass: Number of Attribute Classes
- K: Number of Attributes
- M: Number of Responses
- order: Highest interaction order to consider. Default model-specified k.

Details

This is an internal function briefly used to simulate data and, thus, has been exported into R as well as documented. Output from this function can change in future versions.

Value

Return a list containing the table caches for different parameters

gen_bijectionvector

Generate a vector to map polytomous vector to integers

Description

Converts class into a bijection to integers

Usage

gen_bijectionvector(K, M)

Arguments

- K: Number of Attributes
- M: Number of Response Categories

Value

Return a $K$-length vector containing the bijection vector.
**ohoegdm**  
*Ordinal Higher-Order General Diagnostic Model under the Exploratory Framework (OHOEGDM)*

**Description**

Performs the Gibbs sampling routine for an ordinal higher-order EGDM.

**Usage**

```r
ohoegdm(
  y,
  k,
  m = 2,
  order = k,
  sd_mh = 0.4,
  burnin = 1000L,
  chain_length = 10000L,
  l0 = c(1, rep(100, sum(choose(k, seq_len(order))))),
  l1 = c(1, rep(1, sum(choose(k, seq_len(order))))),
  m0 = 0,
  bq = 1
)
```

**Arguments**

- `y`: Ordinal Item Matrix
- `k`: Dimension to estimate for Q matrix
- `m`: Number of Item Categories. Default is 2 matching the binary case.
- `order`: Highest interaction order to consider. Default model-specified `k`.
- `sd_mh`: Metropolis-Hastings standard deviation tuning parameter.
- `burnin`: Amount of Draws to Burn
- `chain_length`: Number of Iterations for chain.
- `l0`: Spike parameter. Default 1 for intercept and 100 coefficients
- `l1`: Slab parameter. Default 1 for all values.
- `m0, bq`: Additional tuning parameters.

**Details**

The `estimates` list contains the mean information from the sampling procedure. Meanwhile, the `chain` list contains full MCMC values. Moreover, the `details` list provides information regarding the estimation call. Lastly, the `recovery` list stores values that can be used when assessing the method under a simulation study.
Value

A ohoegdm object containing four named lists:

- **estimates**: Averaged chain iterations
  - **thetas**: Average theta coefficients
  - **betas**: Average beta coefficients
  - **deltas**: Average activeness of coefficients
  - **classes**: Average class membership
  - **m2lls**: Average negative two times log-likelihood
  - **omegas**: Average omega
  - **kappas**: Average category threshold parameter
  - **taus**: Average \(K\)-vectors of factor intercept
  - **lambdas**: Average \(K\)-vectors of factor loadings
  - **guessing**: Average guessing item parameter
  - **slipping**: Average slipping item parameter
  - **QS**: Average activeness of Q matrix entries

- **chain**: Chain iterations from the underlying C++ routine.
  - **thetas**: Theta coefficients iterations
  - **betas**: Beta coefficients iterations
  - **deltas**: Activeness of coefficients iterations
  - **classes**: Class membership iterations
  - **m2lls**: Negative two times log-likelihood iterations
  - **omegas**: Omega iterations
  - **kappas**: Category threshold parameter iterations
  - **taus**: \(K\)-vector of factor intercept iterations
  - **lambdas**: \(K\)-vector of factor loadings iterations
  - **guessing**: Guessing item parameter iterations
  - **slipping**: Slipping item parameter iterations

- **details**: Properties used to estimate the model
  - **n**: Number of Subjects
  - **j**: Number of Items
  - **k**: Number of Traits
  - **m**: Number of Item Categories.
  - **order**: Highest interaction order to consider. Default model-specified \(k\).
  - **sd_mh**: Metropolis-Hastings standard deviation tuning parameter.
  - **l0**: Spike parameter
  - **l1**: Slab parameter
  - **m0, bq**: Additional tuning parameters
  - **burnin**: Number of Iterations to discard
  - **chain_length**: Number of Iterations to keep
  - **runtime**: Elapsed time algorithm run time in the C++ code.

- **recovery**: Assess recovery metrics under a simulation study.
  - **Q_item_encoded**: Per-iteration item encodings from Q matrix.
  - **MHsum**: Average acceptance from metropolis hastings sampler
Examples

# Simulation Study
if (requireNamespace("edmdata", quietly = TRUE)) {

# Obtain the full K3 Q matrix from edmdata
Q_full = qmatrix_oracle_k3_j20

# Retain only a subset of the original Q matrix
removal_idx = -c(3, 5, 9, 12, 15, 18, 19, 20)
Q = Q_full[removal_idx, ]

# Construct the beta matrix by-hand
beta = matrix(0, 20, ncol = 8)

# Intercept
beta[, 1] = 1

# Main effects
beta[1:3, 2] = 1.5
beta[4:6, 3] = 1.5
beta[7:9, 5] = 1.5

# Setup two-way effects
beta[10, c(2, 3)] = 1
beta[11, c(3, 4)] = 1
beta[12, c(2, 5)] = 1
beta[13, c(2, 5)] = 1
beta[14, c(2, 6)] = 1
beta[15, c(3, 5)] = 1
beta[16, c(3, 5)] = 1
beta[17, c(3, 7)] = 1

# Setup three-way effects
beta[18:20, c(2, 3, 5)] = 0.75

# Decrease the number of Beta rows
beta = beta[removal_idx, ]

# Construct additional parameters for data simulation
Kappa = matrix(c(0, 1, 2), nrow = 20, ncol = 3, byrow = TRUE) # mkappa
lambda = c(0.25, 1.5, -1.25) # mlambdas
tau = c(0, -0.5, 0.5) # mtaus

# Simulation conditions ----
N = 100  # Number of Observations
J = nrow(beta)  # Number of Items
M = 4  # Number of Response Categories
\texttt{Malpa = 2} \quad \# \text{Number of Classes} \\
\texttt{K = ncol(Q)} \quad \# \text{Number of Attributes} \\
\texttt{order = K} \quad \# \text{Highest interaction to consider} \\
\texttt{sdmtheta = 1} \quad \# \text{Standard deviation for theta values} \\

\textbf{# Simulate data ----} \\

\textbf{# Generate theta values} \\
\texttt{theta = rnorm(N, sd = sdmtheta)} \\

\textbf{# Generate alphas} \\
\texttt{Zs = matrix(1, N, 1) \times \tau +} \\
\texttt{\quad \text{matrix(theta, N, 1) \times \lambda +} \\
\texttt{\quad \text{matrix(rnorm(N \times K), N, K)}}} \\
\texttt{Alphas = 1 \times (Zs > 0)} \\

\texttt{vv = gen\_bijectionvector(K, Malpa)} \\
\texttt{CLS = Alphas \times vv} \\
\texttt{Atab = GenerateAtable(Malpa \times K, K, Malpa, order)\$Atable} \\

\textbf{# Simulate item-level data} \\
\texttt{Ysim = sim\_slcm(N, J, M, Malpa \times K, CLs, Atab, beta, Kappa)} \\

\textbf{# Establish chain properties} \\
\textbf{# Standard Deviation of MH. Set depending on sample size.} \\
\textbf{# If sample size is:} \\
\textbf{# - small, allow for larger standard deviation} \\
\textbf{# - large, allow for smaller standard deviation.} \\
\texttt{sd\_mh = .4} \\
\texttt{burnin = 50} \quad \# \text{Set for demonstration purposes, increase to at least 5,000 in practice.} \\
\texttt{chain\_length = 100} \quad \# \text{Set for demonstration purposes, increase to at least 40,000 in practice.} \\

\textbf{# Setup spike-slab parameters} \\
\texttt{l0s = c(1, rep(100, Malpa \times K - 1))} \\
\texttt{l1s = c(1, rep(1, Malpa \times K - 1))} \\

\texttt{my\_model = ohoegdm::ohoegdm(} \\
\texttt{\hspace{1cm} y = Ysim,} \\
\texttt{\hspace{1cm} k = K,} \\
\texttt{\hspace{1cm} m = M,} \\
\texttt{\hspace{1cm} order = order,} \\
\texttt{\hspace{1cm} l0 = l0s,} \\
\texttt{\hspace{1cm} l1 = l1s,} \\
\texttt{\hspace{1cm} m0 = 0,} \\
\texttt{\hspace{1cm} bq = 1,} \\
\texttt{\hspace{1cm} sd\_mh = sd\_mh,} \\
\texttt{\hspace{1cm} burnin = burnin,} \\
\texttt{\hspace{1cm} chain\_length = chain\_length)} \\
\texttt{)}
Simulate Ordinal Item Data from a Sparse Latent Class Model

Usage

```
sim_slcm(N, J, M, nClass, CLASS, Atable, BETA, KAPPA)
```

Arguments

- **N**: Number of Observations
- **J**: Number of Items
- **M**: Number of Item Categories (2, 3, ..., M)
- **nClass**: Number of Latent Classes
- **CLASS**: A vector of N observations containing the class ID of the subject.
- **Atable**: A matrix of dimensions $M^K \times M^O$ containing the attribute classes in bijection-form. Note, $O$ refers to the model’s highest interaction order.
- **BETA**: A matrix of dimensions $J \times M^K$ containing the coefficients of the reparameterized $\beta$ matrix.
- **KAPPA**: A matrix of dimensions $J \times M$ containing the category threshold parameters

Value

An ordinal item matrix of dimensions $N \times J$ with $M$ response levels.

See Also

`ohoegdm`
Index

gen_bijectionvector, 2
GenerateAtable, 2

ohoegdm, 3, 7

sim_slcm, 7