Package ‘autoRasch’

February 28, 2022

**Version** 0.1.5

**Date** 2022-02-07

**Title** Semi-Automated Rasch Analysis

**Description** Performs Rasch analysis (semi-)automatically, which has been shown to be comparable with the standard Rasch analysis (Feri Wijayanto et al. (2021) <doi:10.1111/bmsp.12218>, Feri Wijayanto et al. (2022) <https://psyarxiv.com/erxuh/>).

**Depends** R (>= 3.4.0)

**Imports** doParallel, foreach, Rcpp, lavaan

**Suggests** testthat (>= 3.0.0), knitr, rmarkdown, markdown

**License** GPL-2

**VignetteBuilder** knitr

**Type** Package

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.1.2

**LinkingTo** RcppArmadillo, Rcpp

**Config/testthat/edition** 3

**NeedsCompilation** yes

**Author** Feri Wijayanto [aut, cre],
  Ioan Gabriel Bucur [aut],
  Perry Groot [aut],
  Tom Heskes [aut],
  Karlien Mul [ctb],
  Baziel G.M. van Engelen [ctb]

**Maintainer** Feri Wijayanto <f.wijayanto@cs.ru.nl>

**Repository** CRAN

**Date/Publication** 2022-02-28 20:00:02 UTC
**R topics documented:**

autoRasch ................................................. 2
autoRaschOptions ....................................... 3
check_unidim ........................................... 5
checkRel .................................................. 5
compute_score ........................................... 7
corr02_multidim ......................................... 9
corr03_multidim .......................................... 9
corr04_multidim .......................................... 10
corr05_multidim .......................................... 10
corr06_multidim .......................................... 10
createGroup ............................................. 11
dataset .................................................... 11
dicho_inh_dset ......................................... 13
dicho_md_dset ........................................... 13
fitStats .................................................... 13
generate_data ........................................... 15
generic_model ........................................... 18
gpcm ....................................................... 19
gpcm_dif .................................................. 20
pcm ........................................................ 20
pcm_dif .................................................... 22
plot_EVC ................................................... 26
plot_ICC .................................................... 27
plot_PImap ................................................ 28
polydif_inh_dset ....................................... 29
poly_inh_dset ........................................... 30
poly_md_dset ............................................ 30
residCor ................................................... 31
shortDIF ................................................... 31
short_poly_data .......................................... 32
stepwise_search ......................................... 32
testlets_dataset ........................................ 35
withinItem_multidim .................................... 35

**Index** ................................................. 36

---

**autoRasch**

**autoRasch: A package for semi-automatic Rasch analysis**

**Description**

This package helps user to do Rasch analysis (semi-)automatically which is comparable to the standard Rasch analysis using common statistics.
autoRaschOptions

---

**autoRasch Optimization Parameters Setting**

**Description**

Returns and updates the default settings used by the functions in **autoRasch** package.

**Usage**

```r
autoRaschOptions(x = NULL)
```

**Arguments**

- `x` A name of single parameter setting that is wanted to be shown. NULL means returns all parameters.

**Details**

cd_control lists the parameters used to control the coordinate descent optimization procedure. The parameters are:

- `init.step` Initial value of the delta parameters updating step. The default is 1.
- `scale.down` A constant value to scale down the updating step. The default is 0.5.
- `maxit.cd.higher` Maximum iteration in the higher level coordinate descent. The default is 500.
- `maxit.cd.lower` Maximum iteration for every coordinate optimization in the lower level coordinate descent. The default is 500.
- `abs.tol` The convergence tolerance. The algorithm stops if it is unable to reduce the negative log likelihood value by the given tolerance. The default is 1e-12.
- `max.dif.par` The convergence tolerance. The algorithm stops if it is unable to update all of the parameters’ value by the given tolerance. The default is 1e-8.

**Value**

- `fixed_par` A vector of names of the parameter types that are set to be fixed. It means that these parameters are not going to be estimated.
- `fixed_theta` A vector of theta values when theta are listed in the fixed_par. If it is not set, it will be set to zero.
- `fixed_beta` A vector of beta values when beta are listed in the fixed_par. If it is not set, it will be set to zero.
- `fixed_gamma` A vector of gamma (natural logarithm of discrimination parameters, \( \alpha = \exp(\gamma) \)) values when gamma are listed in the fixed_par. If it is not set, it will be set to zero.
- `fixed_delta` A vector of delta values when delta are listed in the fixed_par. If it is not set, it will be set to zero.
isPenalized_theta

It is a logical parameter whether, in the estimation procedure, theta is penalized or not.

isPenalized_gamma

It is a logical parameter whether, in the estimation procedure, gamma is penalized or not.

isPenalized_delta

It is a logical parameter whether, in the estimation procedure, delta is penalized or not.

groups_map

A matrix $n \times f$ to map the subject into DIF groups, where $n$ is number of subjects and $f$ is number of focal groups.

optz_method

Options of the optimization method used. The default is optim which implies on applying the PJMLE which is implemented using optim(). When it is set to mixed means that it applies the coordinate descent.

optim_control

A list of setting parameters of the optim(). For complete settings can be seen in stats::optim().

lambda_theta

The regularization parameter to the theta. The default value is 0.05

lambda_in

The regularization parameter to the gamma in the included itemset. The default value is 50.

lambda_out

The regularization parameter to the gamma in the excluded itemset. The default value is 1.

lambda_delta

The regularization parameter to the delta. The default value is 10.

randomized

A logical parameter whether the initial values of the estimated parameters are randomized or not.

random.init.th

A threshold value to limit the range of the initial values. The default value is 1e-2, means that the initial values range between $[-0.01,0.01]$

isHessian

A logical parameter whether, in the estimation procedure, need to return the Hessian matrix or not. The default value is TRUE, which means the Hessian matrix will be computed.

cd_control

A list of coordinate descent optimization setting.

Examples

```r
### To show the default values
autoRaschOptions()
autoRaschOptions(x = "isHessian")

### To change the default values
adj_setting <- autoRaschOptions()
adj_setting$isHessian <- FALSE
```
### check.unidim

**Description**
This function checks the unidimensionality status using the confirmatory factor analysis.

**Usage**
```r
check.unidim(
  x,
  is.polychor = TRUE,
  se = "robust",
  estimator = "WLSMV",
  test = "Satorra.Bentler"
)
```

**Arguments**
- `x`: The dataset of responses.
- `is.polychor`: A boolean parameter to set whether the dataset is categorical or not.
- `se`: The standard error type. Please refer to `cfa()`.
- `estimator`: The type of the estimator. Please refer to `cfa()`.
- `test`: The test used in the factor analysis. Please refer to `cfa()`.

**Value**
A list of the CFA output and the some of the goodness-of-fit indices (i.e., cfi, tli, rmsea, and srmr)

### checkRel

**Description**
This function computes the reliability index, separation and the standard error of the models estimation.

**Usage**
```r
checkRel(obj)
```  
```r
## S3 method for class 'seprel'
summary(object, ...)
```
Arguments

obj Object that resulted from any models estimation, e.g., pcm, gpcm, pcmdif, and gpcmdif.

object The object of class 'seprel'.

... Further arguments to be passed.

Details

Person reliability index

Value

A list of two objects, the reliability and the standard error.

reliability

- PRI Person reliability index.
- PSR Person separation reliability.
- IRI Item reliability index.
- ISR Item separation reliability.

stdError

- var_err_pers A matrix of variance error of the estimation.
- std_err_pers A matrix of standard error of the estimation.
- rmsse_pers Root mean square of the standard error per person.
- var_err_item A matrix of variance error of the estimation.
- std_err_item A matrix of standard error of the estimation.
- rmsse_item Root mean square of the standard error per person.
- hessian_theta Hessian matrix of theta parameter.
- hessian_beta Hessian matrix of beta parameter.

Examples

pcmObject <- pcm(shortDIF)
rel <- checkRel(pcmObject)
summary(rel)
compute_score

**compute_score**  
*Compute the In-plus-out-of-questionnaire log likelihood (with DIF)*  
(*IPOQ-LL(-DIF))*

**Description**

`compute_score` computes the IPOQ-LL/IPQ-LL-DIF score of an instrument (included set) of the given initial survey. While `compute_scores` computes the IPOQ-LL/IPQ-LL-DIF score of many (more than one) instruments (included sets) of the given initial survey simultaneously.

**Usage**

```r
compute_score(
  X,
  incl_set,
  type = c("ipoqll", "ipoqlldif"),
  groups_map = c(),
  init_par_iq = c(),
  init_par_oq = c(),
  optim_control_iq = c(),
  optim_control_oq = c(),
  setting_par_iq = c(),
  setting_par_oq = c(),
  method = c("fast", "novel")
)
```

```r
compute_scores(
  X,
  incl_sets,
  type = c("ipoqll", "ipoqlldif"),
  step_direct = c("fixed", "forward", "backward"),
  groups_map = c(),
  init_par_iq = c(),
  init_par_oq = c(),
  optim_control_iq = c(),
  optim_control_oq = c(),
  setting_par_iq = c(),
  setting_par_oq = c(),
  cores = NULL,
  method = c("fast", "novel"),
  timeLimit = 3600
)
```

```r
## S3 method for class 'score'
summary(object, ...)
```
**compute_score**

**Arguments**

- **X**
  A matrix or data.frame of the observed responses (ordinal or binary response).

- **incl_set**
  A vector of the items (columns) number in the data.frame X that are included in the included set.

- **type**
  The type of the score. `ipoql1` if we ignore the presence of the DIF and `ipoqlldif` if we want to consider the DIF effect.

- **groups_map**
  Matrix to map the respondents to the DIF groups.

- **init_par_iq**
  Initial values of the parameters in the included set before the estimation begin.

- **init_par_oq**
  Initial values of the parameters in the excluded set before the estimation begin.

- **optim_control_iq**
  The optimisation setting of the included set. See `stats::optim()` control parameter.

- **optim_control_oq**
  The optimisation setting of the excluded set. See `stats::optim()` control parameter.

- **setting_par_iq**
  The coordinate descent optimisation setting of the included set. See `autoRasch::autoRaschOptions()` cd_control parameter.

- **setting_par_oq**
  The coordinate descent optimisation setting of the excluded set. See `autoRasch::autoRaschOptions()` cd_control parameter.

- **method**
  The implementation option of log likelihood function. `fast` using a c++ implementation and `novel` using an R implementation.

- **incl_sets**
  A matrix as a results of a `rbind` of `incl_set`.

- **step_direct**
  How will you compute the criterion score. `fixed` for the given itemset, `forward` computes all the scores of the possible combination of items if an item is added to the given set, `backward` computes all the scores of the possible combination of items if an item is removed to the given set.

- **cores**
  Number of cores that is used in the paralellization.

- **timeLimit**
  The maximum execution time to compute the criterion score. Exceeding the setting time, the computation will be halted.

- **object**
  The object from the class `score`. The result of the score computation.

- **...**
  Further argument passed or from other method.

**Value**

`compute_score` will return a vector which contains in-questionnaire log likelihood (IQ-LL(-DIF)), out-of-questionnaire log likelihood (OQ-LL(-DIF)), IPOQ-LL(-DIF), included set’s items’ number in the given initial survey, the estimated theta parameters, the estimated items’ parameters in the included set, and the estimated items’ parameters in the excluded set, sequentially.

`compute_scores` will return a matrix as a result of the `rbind` operation of the `compute_score`’s result.
Examples

```r
ipoqll_score <- compute_score(shortDIF,incl_set = c(1:3),type = "ipoqll")

## Not run:
ipoqll_scores <- compute_scores(shortDIF,incl_set = rbind(c(1:3),c(2:4)),
                                 type = "ipoqll", cores = 2)

## End(Not run)
```

correl02_multidim  

Multidimensional polytomous data set with 0.2 correlation

Description

 Multidimensional polytomous data set with 0.2 correlation

Usage

```r
data(correl02_multidim)
```

Format

 An object of class `data.frame` with 301 rows and 12 columns.

correl03_multidim  

Multidimensional polytomous data set with 0.3 correlation

Description

 Multidimensional polytomous data set with 0.3 correlation

Usage

```r
data(correl03_multidim)
```

Format

 An object of class `data.frame` with 301 rows and 12 columns.
**correl04_multidim**  
*Multidimensional polytomous data set with 0.4 Correlation*

**Description**
Multidimensional polytomous data set with 0.4 Correlation

**Usage**
```r
data(correl04_multidim)
```

**Format**
An object of class `data.frame` with 301 rows and 12 columns.

---

**correl05_multidim**  
*Multidimensional polytomous data set with 0.5 Correlation*

**Description**
Multidimensional polytomous data set with 0.5 Correlation

**Usage**
```r
data(correl05_multidim)
```

**Format**
An object of class `data.frame` with 301 rows and 12 columns.

---

**correl06_multidim**  
*Multidimensional polytomous data set with 0.6 Correlation*

**Description**
Multidimensional polytomous data set with 0.6 Correlation

**Usage**
```r
data(correl06_multidim)
```

**Format**
An object of class `data.frame` with 301 rows and 12 columns.
createGroup

Create Mapping Matrix of DIF Groups

Description

This function automatically create a mapping matrix based on the existing DIF inducing covariates.

Usage

createGroup(backInfo, idxUsed = NULL, contMethod = c("mean", "median"))

Arguments

backInfo A matrix of person background information (e.g., gender, country, age, etc);
idxUsed The column number of backInfo that is used for creating the mapping matrix.
contMethod The method of how to handle a continuous variable (e.g., mean, median). This parameter is passing a function used to split the variable into binary. The default is mean.

Value

A binary matrix that maps respondents to the groups that the respondents belongs to.

dataset

The Simulated Dataset

Description

The artificial datasets used for simulation on various cases. The datasets consist of:

- inhomogenous_rasch_dataset
- inhomogenous_dataset
- uncorrel_rasch_multidim
- uncorrel_multidim
- correl02_multidim
- correl03_multidim
- correl04_multidim
- correl05_multidim
- correl06_multidim
- withinItem_multidim
- testlets_dataset
Details

inhomogenous_rasch_dataset is an artificial dataset of dichotomous responses which simulates three subscales with different predictability level (discrimination parameters). This dataset is generated by

```r
inhomogenous_rasch_dataset <- generate_data(responseType = "discriminate", ncat = 2, alpha = c(0.04, 0.045, 0.05, 0.055, 0.06, 0.065, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 2.6, 2.65, 2.7, 2.75, 2.8, 2.85, 2.9))
```

inhomogenous_dataset is an artificial dataset which simulates three subscales with different predictability level (discrimination parameters). This dataset is generated by

```r
inhomogenous_dataset <- generate_data(responseType = "discriminate", alpha = c(0.04, 0.045, 0.05, 0.055, 0.06, 0.065, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 2.6, 2.65, 2.7, 2.75, 2.8, 2.85, 2.9))
```

uncorrel_rasch_multidim is an artificial dataset of dichotomous responses which simulates three uncorrelated subscales from different dimensions. This dataset is generated by

```r
uncorrel_rasch_multidim <- generate_data(responseType = "multidim.nocorrel", ncat = 2)
```

uncorrel_multidim is an artificial dataset which simulates three uncorrelated subscales from different dimensions. This dataset is generated by

```r
uncorrel_multidim <- generate_data(responseType = "multidim.nocorrel")
```

correl02_multidim is an artificial dataset which simulates two subscales from different dimensions that having correlation of 0.2. This dataset is generated by

```r
correl02_multidim <- generate_data(responseType = "multidim.withcorrel", corLevel = 0.2)
```

Similarly, correl03_multidim, correl04_multidim, correl05_multidim, and correl06_multidim are artificial datasets which consist of two correlated subscales with correlation of 0.3, 0.4, 0.5, and 0.6, respectively.

withinItem_multidim is an artificial dataset which consist of three subscales (dimensions) with some of items relate to more than one subscale (dimension). This dataset is generated by

```r
withinItem_multidim <- generate_data(responseType = "multidim.within", ndim = 3, dim.members = list(c(1:6, 13), c(3, 7:12), c(5, 13:18)))
```

testlets_dataset is an artificial dataset which consist of two subscales with some of items relate to more than one subscale (dimension). This dataset is generated by

```r
testlets_dataset <- generate_data(responseType = "testlets", ndim = 2, sdlambda = c(0, 4))
```

See Also

pcm, pcm_dif, gpcm, gpcm_dif

Examples

```r
#res <- pcm(poly_inh_dset)
#res
#summary(res)

#pcmdif_res <- pcm_dif(polydif_inh_dset, groups_map = c(rep(1, 245), rep(0, 245)))
#fit_res <- fitStats(pcmdif_res)
#summary(fit_res)
#plot(fit_res, plotx = "gamma", ploty = "outfit")
```
**dicho_inh_dset**

**Inhomogenous Dichotomous Data Set**

**Description**

Data set with binary type responses containing three subsets with different discrimination values.

**Usage**

```r
data(dicho_inh_dset)
```

**Format**

An object of class `data.frame` with 301 rows and 18 columns.

---

**dicho_md_dset**

**Uncorrelated Multidimensional Dichotomous Data Set**

**Description**

Data set with binary type responses containing three subsets which represent different uncorrelated dimensions.

**Usage**

```r
data(dicho_md_dset)
```

**Format**

An object of class `data.frame` with 301 rows and 18 columns.

---

**fitStats**

**Fit statistics**

**Description**

The goodness-of-fit statistics of Rasch analysis for items and persons. It consists of Outfit (Unweighted) Mean Square, Infit (Weighted) Mean Square, Outfit ZSTD (Standardized Unweighted Mean Square), and Outfit ZSTD (Standardized Weighted Mean Square)
Usage

`fitStats(obj, isAlpha = TRUE, isTraced = FALSE)`

```r
## S3 method for class 'fit'
summary(object, ...)
```

`itemfit(objFit)`

`personfit(objFit)`

`plot_fitStats(objFit, toPlot = c("alpha", "infit"), useName = FALSE, ...)`

Arguments

- `obj` The object of class 'pcm' or 'pcmdif'.
- `isAlpha` Boolean value that indicates whether the discrimination parameters is needed to be estimated or not. The discrimination parameters are estimated using the corresponding models (GPCM or GPCM-DIF).
- `isTraced` A list of some matrices, i.e., the expected values, the variances, the curtosis, and the standardized residual matrix.
- `object` The object of class 'fit'.
- `...` Further arguments to be passed.
- `objFit` The object of class 'fit'.
- `toPlot` An array with length two `c(x,y)`, to choose what to plot. There are five options to plot, which are alpha, outfit, infit, outfitZ, and infitz
- `useName` A logical statement whether the name of the variable are going to be used in the plot instead of the variable order.

Value

`fitStats()` will return a list which contains:

- `alpha` A vector of estimated discrimination parameters for each items.

**i.fit** Item fit statistics.

- `i.outfitMSQ` A vector of Outfit mean square values for each items.
- `i.infitMSQ` A vector of Infit mean square values for each items.
- `i.outfitZ` A vector of OutfitZ values for each items.
- `i.infitZ` A vector of InfitZ values for each items.

**p.fit** Person fit statistics.

- `p.outfitMSQ` A vector of Outfit mean square values for each persons.
- `p.infitMSQ` A vector of Infit mean square values for each persons.
- `p.outfitZ` A vector of OutfitZ values for each persons.
generate_data

• p.infitZ A vector of InfitZ values for each person.

traceMat Some computed matrices in the process. Only if isTraced = TRUE

• emat The expected values matrix.
• vmat The variance matrix.
• cmat The curtosis matrix.
• std.res The standardized residual.

---

**generate_data**

*Generate the artificial dataset*

---

**Description**

This function generates simulated datasets with different attributes

**Usage**

```r
generate_data(
  responseType = "multidim.nocorrel",
  theta = c(-3, 3),
  sdtheta = 6,
  ntheta = 301,
  beta = c(-2.5, 2.5),
  sdbeta = 4,
  nitem = 6,
  alpha = c(1),
  sdlambda = 1,
  ncat = 5,
  thGap = 0.8,
  ndim = 3,
  randtype = "uniform",
  corLevel = 0,
  dim.members = c(),
  seed = NULL
)
```

**Arguments**

- **responseType** The type of the dataset. The types include multidim.nocorrel, multidim.withcorrel, discriminate, multidim.within, and testlets.
- **theta** A vector of the ability parameters range value, c(min.theta, max.theta). It applies when the randtype = "uniform".
- **sdtheta** Standard deviation which is used to generate theta values using stats::rnorm() with n = ntheta, mean = 0, and sd = sdtheta. It applies when the randtype = "normal".
generate_data

ntheta The number of the observations.

beta A vector of the item difficulty parameters range value, \(c(\text{min}.\beta, \text{max}.\beta)\).
It applies when the \text{randtype} = "uniform".

sdbeta Standard deviation which is used to generate item location values using \text{stats::rnorm()} with \(n = nitem\), \text{mean} = 0, and \text{sd} = sdbeta. It applies when the \text{randtype} = "normal".

nitem The number of the items in each subgroup.

alpha A vector of the discrimination parameters apply to each items.

sdlambda A vector of the standard deviation to simulate the testlet (local dependency) effect. The effect is added using \text{stats::rnorm()} with \(n = ntheta\), \text{mean} = 0, and \text{sd} = sdlambda

ncat The number of the response categories

thGap The difference between adjacent threshold.

ndim The number of subgroups (dimensions/testlets) created.

randtype The randomize type. This includes uniform and normal.

corLevel The correlation between the two dimensions.

dim.members The list of item members in each dimension.

seed Integer seed for reproducibility.

**Value**

The generated dataset as a \text{data.frame}.

**Examples**

# 1. Multidimensional Polytomous Dataset with 0.2 Correlation
# Generate multidimensional dataset which having correlation of 0.2 between the dimensions
corr02_multidim <- generate_data(
  responseType = "multidim.withcorrel", corLevel = 0.2, seed = 2021
)

# 2. Within-item Multidimensional Polytomous Dataset
# Generate multidimensional dataset with some items relate to more than one
# dimension.
withinItem_multidim <- generate_data(
  responseType = "multidim.within", ndim = 3,
  dim.members = list(c(1:6,13),c(3,7:12),c(5,13:18)), seed = 2021
)

# 3. Multi-testlets Polytomous Dataset
# Generate dataset which consist of two bundle items with different level of
# local dependency effect.
testslets_dataset <- generate_data(
  responseType = "testlets", ndim = 2, sdlambda = c(0,4), seed = 2021
)

# 4a. Inhomogenous Dichotomous Dataset
# Generate dataset with binary type responses containing three subsets
# with different discrimination values.

dicho_inh_dset <- generate_data(
  responseType = "discriminate", ncat = 2, seed = 2021,
  alpha = c(0.04,0.045,0.05,0.055,0.06,0.065,0.2,0.25,0.3,0.35,0.4,0.45,
            2.6,2.65,2.7,2.75,2.8,2.85)
)

# 4b. Inhomogenous Polytomous Dataset
# Generate dataset with polytomous responses (five categories) containing
# three subsets with different discrimination values.

poly_inh_dset <- generate_data(
  responseType = "discriminate", ncat = 5, seed = 2021,
  alpha = c(0.04,0.045,0.05,0.055,0.06,0.065,0.2,0.25,0.3,0.35,0.4,0.45,
            2.6,2.65,2.7,2.75,2.8,2.85)
)

# 4c. Shorter Inhomogenous Polytomous Dataset
short_poly_dset <- generate_data(
  alpha = c(0.02,0.5,2), nitem = 3, ndim = 3, ncat = 5,
  theta = c(-6,6), beta = c(-4,4), ntheta = 151, seed = 2021
)

# 4d. Short Dataset containing DIF items
# Generate dataset with polytomous responses (five categories) containing
# three subsets with different discrimination values and two DIF-items.
seed <- c(54748,96765)
difset_short1 <- generate_data(responseType = "discriminate", ncat = 3,
                                 ntheta = 50, nitem = 3, ndim = 1,
                                 seed = seed[1], alpha = c(2))
difset_short2 <- generate_data(responseType = "discriminate", ncat = 3,
                                 ntheta = 50, nitem = 2, ndim = 1,
                                 seed = seed[2], alpha = c(0.8),
                                 beta = c(-2.5,2.5))
shortDIF <- cbind(rbind(difset_short1,difset_short1),
                   c(difset_short2[,1],difset_short2[,2]))

# 5a. Uncorrelated Multidimensional Dichotomous Dataset
# Generate dataset with binary type responses containing three subsets which
# represent different uncorrelated dimensions.
dicho_md_dset <- generate_data(
  responseType = "multidim.nocorrel", ncat = 2, seed = 2021
)

# 5b. Uncorrelated Multidimensional Polytomous Dataset
# Generate dataset with polytomous responses (five categories) containing
# three subsets which represent different uncorrelated dimensions.
poly_md_dset <- generate_data(
  responseType = "multidim.nocorrel", ncat = 5, seed = 2021
)
generic_model  Estimation of the generic form of the models

Description

This function computes the parameter estimates of the generic form of the models by using penalized JML estimation. It allows users to adjust the default settings of the estimation.

Usage

generic_model(X, init_par = c(), setting = c())

Arguments

X  Input dataset as matrix or data frame with ordinal responses (starting from 0); rows represent individuals, column represent items.
init_par  Initial values of the estimated parameters.
setting  Parameter settings which are listed in autoRaschOptions().

Details

In the discrimination parameters estimation, instead of estimating the discrimination parameters, we are estimating the natural logarithm of the parameters to avoid negative values, $\alpha = \exp(\gamma)$.

Value

X  The dataset that is used for estimation.
name  The name of each items in the dataset.
mt_vek  A vector of the highest response category as many as the number of items.
loglik  The log likelihood of the estimation.
objtype  Type of the model that is used.
delta  A vector of the DIF parameters of each items on each groups.
gamma  A vector of the natural logarithm of discrimination parameters of each items.
beta  A vector of the difficulty parameter of each items’ categories (thresholds).
theta  A vector of the ability parameters of each individuals.
Estimation of The Generalized Partial Credit Model

Description

This function computes the parameter estimates of a generalized partial credit model for polytomous responses by using penalized JML estimation. Inputting a dichotomous responses to this model, will automatically transforms the GPCM to the 2-PL model.

Usage

\[
gpcm(X, \text{init\_par} = c(), \text{setting} = c(), \text{method} = c("fast", "novel"))
\]

## S3 method for class 'gpcm'


summary(object, ...)

## S3 method for class 'gpcm'


print(x, ...)

Arguments

- **X**: Input dataset as matrix or data frame with ordinal responses (starting from 0); rows represent individuals, columns represent items.
- **init\_par**: a vector of initial values of the estimated parameters.
- **setting**: a list of the optimization control setting parameters. See autoRaschOptions().
- **method**: The implementation option of log likelihood function. fast using a C++ implementation and novel using an R implementation.
- **object**: The object of class 'gpcm'.
- **...**: Further arguments to be passed.
- **x**: The object of class 'gpcm'.

Details

In the discrimination parameters estimation, instead of estimating the discrimination parameters \(\alpha\), we are estimating its natural logarithm to avoid negative values, \(\alpha = exp(\gamma)\).

Value

- **X**: The dataset that is used for estimation.
- **mt\_vek**: A vector of the highest response given to items.
- **itemName**: The vector of names of items (columns) in the dataset.
- **loglik**: The log likelihood of the estimation.
- **hessian**: The hessian matrix. Only when the isHessian = TRUE.
- **gamma**: A vector of the natural logarithm of discrimination parameters of each items.
- **beta**: A vector of the difficulty parameter of each items’ categories (thresholds).
- **theta**: A vector of the ability parameters of each individuals.
References

See Also
pcm, gpcm

Examples

gpcm_res <- gpcm(short_poly_data)
summary(gpcm_res, par = "alpha")

gpcm_dif
Estimation of The Generalized Partial Credit Model with DIF

Description
This function computes the parameter estimates of a generalized partial credit model with DIF for polytomous responses by using penalized JML estimation.

Usage

gpcm_dif(
  X,
  init_par = c(),
  groups_map = c(),
  setting = c(),
  method = c("fast", "novel")
)

## S3 method for class 'gpcmdif'
summary(object, ...)

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

Arguments

X A matrix or data frame as an input with ordinal responses (starting from 0); rows represent individuals, columns represent items.

init_par a vector of initial values of the estimated parameters.

groups_map Binary matrix. Respondents membership to DIF groups; rows represent individuals, column represent group partitions.

setting a list of the optimization control setting parameters. See autoRaschOptions()
method

The implementation option of log likelihood function. fast using a c++ implementation and novel using an R implementation.

object

The object of class 'gpcm_dif'.

... Further arguments to be passed.

x The object of class 'gpcm_dif'.

Details

In the discrimination parameters estimation, instead of estimating the discrimination parameters, we are estimating the natural logarithm of the parameters to avoid negative values, $\alpha = exp(\gamma)$.

Value

X The dataset that is used for estimation.

mt_vek A vector of the highest responses given to items.

itemName The vector of names of items (columns) in the dataset.

loglik The log likelihood of the estimation.

hessian The hessian matrix. Only when the isHessian = TRUE.

delta A vector of the DIF parameters of each items on each groups.

gamma A vector of the natural logarithm of discrimination parameters of each items.

beta A vector of the difficulty parameter of each items' categories (thresholds).

theta A vector of the ability parameters of each individuals.

See Also

pcm, pcm_dif, gpcm, gpcm_dif

Examples

```r
## Not run:
gpcm_dif_res <- gpcm_dif(shortDIF, groups_map = c(rep(1,50),rep(0,50)))
summary(gpcm_dif_res, par="delta")

## End(Not run)
```
**Estimation of The Partial Credit Model (PCM)**

**Description**

This function computes the parameter estimates of a partial credit model for dichotomous and polytomous responses by using penalized joint maximum likelihood estimation (PJMLE). Inputting a dichotomous responses to this model, will automatically transforms the PCM to the Rasch model.

**fitStats** compute the fit statistics (e.g., Outfit and Infit) of the PCM model estimation (items and persons).

**Usage**

```r
pcm(X, init_par = c(), setting = c(), method = c("fast", "novel"))
```

```r
## S3 method for class 'pcm'
f

```n

```r
## S3 method for class 'pcm'
fitStats(obj, isAlpha = TRUE, isTraced = FALSE)
```

```r
## S3 method for class 'pcm'
summary(object, ...)
```

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

**Arguments**

- **X**: Input dataset as matrix or data frame with ordinal responses (starting from 0); rows represent individuals, columns represent items.
- **init_par**: a vector of initial values of the estimated parameters.
- **setting**: a list of the optimization control setting parameters. See `autoRaschOptions()`.
- **method**: The implementation option of log likelihood function. `fast` using a C++ implementation and `novel` using an R implementation.
- **obj**: The object of class 'pcm'.
- **isAlpha**: Boolean value that indicates whether the discrimination parameters is needed to be estimated or not. The discrimination parameters are estimated using the corresponding models (GPCM or GPCM-DIF).
- **isTraced**: A list of some matrices, i.e., the expected values, the variances, the curtosis, and the standardized residual matrix.
- **object**: The object of class 'pcm'.
- **...**: Further arguments to be passed.
- **x**: The object of class 'pcm'.
Value

\texttt{pcm()} will return a \texttt{list} which contains:

- \texttt{X} The dataset that is used for estimation.
- \texttt{mt_vek} A vector of the highest response given to items.
- \texttt{itemName} The vector of names of items (columns) in the dataset.
- \texttt{loglik} The log likelihood of the estimation.
- \texttt{hessian} The hessian matrix. Only when the \texttt{isHessian = TRUE}.
- \texttt{beta} A vector of the difficulty parameter of each categories of items (thresholds).
- \texttt{theta} A vector of the ability parameters of each individuals.

\texttt{fitStats()} will return a \texttt{list} which contains:

- \texttt{alpha} A vector of estimated discrimination parameters for each items.

\texttt{i.fit} Item fit statistics.
- \texttt{i.outfitMSQ} A vector of Outfit mean square values for each items.
- \texttt{i.infitMSQ} A vector of Infit mean square values for each items.
- \texttt{i.outfitZ} A vector of OutfitZ values for each items.
- \texttt{i.infitZ} A vector of InfitZ values for each items.

\texttt{p.fit} Person fit statistics.
- \texttt{p.outfitMSQ} A vector of Outfit mean square values for each persons.
- \texttt{p.infitMSQ} A vector of Infit mean square values for each persons.
- \texttt{p.outfitZ} A vector of OutfitZ values for each persons.
- \texttt{p.infitZ} A vector of InfitZ values for each persons.

\texttt{traceMat} Some computed matrices in the process. Only if \texttt{isTraced = TRUE}
- \texttt{emat} The expected values matrix.
- \texttt{vmat} The variance matrix.
- \texttt{cmat} The curtosis matrix.
- \texttt{std.res} The standardized residual.

References


See Also

pcm, gpcm

Examples

pcm_res <- pcm(shortDIF)
summary(pcm_res)

#To summarize only for beta parameters
summary(pcm_res, par="beta")
fit_res <- fitStats(pcm_res, isTraced = TRUE)
itemfit(fit_res)
personfit(fit_res)
plot_fitStats(fit_res, toPlot = c("alpha","outfit"), useName = TRUE)

pcm_dif
Estimation of The Partial Credit Model with DIF

Description

This function computes the parameter estimates of a partial credit model with DIF for dichotomous and polytomous responses by implementing the coordinate descent.

fitStats compute the fit statistics (i.e., Outfit and Infit) of the PCM-DIF model estimation (items and persons).

Usage

pcm_dif(
  X,
  init_par = c(),
  groups_map = c(),
  setting = c(),
  method = c("fast", "novel")
)

## S3 method for class 'pcmdif'
fitStats(obj, isAlpha = TRUE, isTraced = FALSE)

## S3 method for class 'pcmdif'
summary(object, ...)

## S3 method for class 'pcmdif'
print(x, ...)
Arguments

- **X**: A matrix or data frame as an input with ordinal responses (starting from 0); rows represent individuals, columns represent items.
- **init_par**: a vector of initial values of the estimated parameters.
- **groups_map**: Binary matrix. Respondents membership to DIF groups; rows represent individuals, column represent group partitions.
- **setting**: a list of the optimization control setting parameters. See `autoRaschOptions()` method.
- **method**: The implementation option of log likelihood function. `fast` using a C++ implementation and `novel` using an R implementation.
- **obj**: The object of class `pcmdif`.
- **isAlpha**: Boolean value that indicates whether the discrimination parameters is needed to be estimated or not. The discrimination parameters are estimated using the corresponding models (GPCM or GPCM-DIF).
- **isTraced**: Logical statement whether returns some computed matrices, i.e., the expected values, the variances, the curtosis, and the standardized residual matrix.
- **object**: The object of class `pcmdif`.
- **...**: Further arguments to be passed.
- **x**: The object of class `pcmdif`.

Value

`pcm_dif()` will return a list which contains:

- **X**: The dataset that is used for estimation.
- **mt_vek**: A vector of the highest response given to items.
- **itemName**: The vector of names of items (columns) in the dataset.
- **loglik**: The log likelihood of the estimation.
- **hessian**: The hessian matrix. Only when the `isHessian = TRUE`.
- **beta**: A vector of the difficulty parameter of each categories of items (thresholds).
- **theta**: A vector of the ability parameters of each individuals.

`fitStats()` will return a list which contains:

- **alpha**: A vector of estimated discrimination parameters for each items.

  * **i.fit** Item fit statistics.
    - **i.outfitMSQ**: A vector of Outfit mean square values for each items.
    - **i.infitMSQ**: A vector of Infit mean square values for each items.
    - **i.outfitZ**: A vector of OutfitZ values for each items.
    - **i.infitZ**: A vector of InfitZ values for each items.

  * **p.fit** Person fit statistics.
    - **p.outfitMSQ**: A vector of Outfit mean square values for each persons.
• p.infitMSQ A vector of Infit mean square values for each persons.
• p.outfitZ A vector of OutfitZ values for each persons.
• p.infitZ A vector of InfitZ values for each persons.

traceMat Some computed matrices in the process. Only if isTraced = TRUE

• emat The expected values matrix.
• vmat The variance matrix.
• cmat The curtosis matrix.
• std.res The standardized residual.

See Also
pcm, pcm_dif, gpcm, gpcm_dif

Examples
## Not run:
pcmdif_res <- pcm_dif(shortDIF, groups_map = c(rep(1,50),rep(0,50)))
fit_res <- fitStats(pcmdif_res)
itemfit(fit_res)
personfit(fit_res)
plot_fitStats(fit_res, toPlot = c("alpha","outfit"), useName = FALSE)
## End(Not run)
Arguments

- **obj**: The object of class 'pcm'.
- **itemno**: A number of the item that is wanted to be plot.
- **xlab**: a title for the x axis.
- **ylab**: a title for the y axis.
- **xlim**: the x limits (x1, x2) of the plot. Note that x1 > x2 is allowed and leads to a 'reversed axis'. The default value, NULL, indicates that the range of the finite values to be plotted should be used; see `plot.default()`
- **col**: a vector of plotting colors
- **lty**: a vector of line types.
- **...**: Further arguments to be passed.

Value

There are no values to return. Instead, it plots expected values from the model.

Examples

```r
res <- pcm(short_poly_data)
plot_EVC(res, itemno = 4)
```

---

**plot_ICC**  
_Plot The Item Characteristic Curves_

Description

This function plots the curve(s) of the estimated ability parameters against the probabilities of responses.

Usage

```r
plot_ICC(
  obj,
  itemno = 5,
  xlab = NULL,
  ylab = NULL,
  xlim = c(-10, 10),
  col = c("green4", "darkorange2", "red2"),
  lty = c(1, 1, 1),
  main = NULL,
  ...
)
```
Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>obj</code></td>
<td>The object of class 'pcm'.</td>
</tr>
<tr>
<td><code>itemno</code></td>
<td>A number of the item that is wanted to be plot.</td>
</tr>
<tr>
<td><code>xlab</code></td>
<td>a title for the x axis.</td>
</tr>
<tr>
<td><code>ylab</code></td>
<td>a title for the y axis.</td>
</tr>
<tr>
<td><code>xlim</code></td>
<td>the x limits (x1, x2) of the plot. Note that x1 &gt; x2 is allowed and leads to a 'reversed axis'. The default value, NULL, indicates that the range of the finite values to be plotted should be used; see <code>plot.default()</code></td>
</tr>
<tr>
<td><code>col</code></td>
<td>a vector of plotting colors</td>
</tr>
<tr>
<td><code>lty</code></td>
<td>a vector of line types.</td>
</tr>
<tr>
<td><code>main</code></td>
<td>String. Plot title.</td>
</tr>
<tr>
<td><code>...</code></td>
<td>Further arguments to be passed.</td>
</tr>
</tbody>
</table>

Value

There are no values to return. Instead, it plots the curve of item characteristics from the model.

Examples

```r
res <- pcm(short_poly_data)
plot_ICC(res, itemno = 4)
```

Description

This function maps the distribution of the persons’ abilities and the items difficulties along the latent continuum.

Usage

```r
plot_PImap(
  obj,
  main = NULL,
  xlab = NULL,
  cex = NULL,
  cex.lab = NULL,
  cex.axis = NULL,
  cex.main = NULL,
  lwd = NULL,
  v = NULL,
  th_dif = 1e-05
)
```
Arguments

obj
The object of class 'pcm'.
main
main title of the plot; see `plot.default()`.
xlab
Label for the x-axis; see `plot.default()`.
cex
A numerical value giving the amount by which plotting text and symbols should be magnified relative to the default; see `par()`.
cex.lab
The magnification to be used for x and y labels relative to the current setting of cex; see `par()`.
cex.axis
The magnification to be used for axis annotation relative to the current setting of cex; see `par()`.
cex.main
The magnification to be used for main titles relative to the current setting of cex; see `par()`.
lwd
The line width, a positive number, defaulting to 1; see `par()`.
v
Variable names used
th_dif
The threshold at which a DIF effect is still considered a DIF.

Value
There are no values to return. Instead, it shows a graphical map of the estimated ability and the estimated difficulty on the same scale.

Examples

```r
## Not run:
groupsMap <- matrix(c(rep(1,50),rep(0,50)),ncol = 1, dimnames = list(c(1:100),c("V1")))
pcmdif_res <- pcm_dif(shortDIF, groups_map = groupsMap)
plot_PImap(pcmdif_res)
## End(Not run)
```

polydif_inh_dset

The Inhomogenous Polytomous Dataset containing DIF items

Description

The artificial data set of a polytomous responses (five categories) which contains three subsets with different discrimination values and two DIF-items.

Usage

data(polydif_inh_dset)

Format

An object of class `data.frame` with 490 rows and 20 columns.
**poly_inh_dset**  
*The Inhomogenous Polytomous Dataset*

**Description**

The artificial dataset of a polytomous responses (five categories) which contains three subsets with different discrimination values. To reproduce this dataset:

```r
poly_inh_dset <- generate_data(responseType = "discriminate", ncat = 5, alpha = c(0.04, 0.045, 0.05, 0.055, 0.06, 0.065, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 2.6, 2.65, 2.7, 2.75, 2.8, 2.85, 2.9))
```

will lead to similar but not the same dataset, due to the randomization.

**Usage**

```r
data(poly_inh_dset)
```

**Format**

An object of class `data.frame` with 301 rows and 18 columns.

---

**poly_md_dset**  
*Uncorrelated Multidimensional Polytomous Data Set*

**Description**

Data set with polytomous responses (five categories) containing three subsets which represent different uncorrelated dimensions.

**Usage**

```r
data(poly_md_dset)
```

**Format**

An object of class `data.frame` with 301 rows and 18 columns.
**residCor**

*Residual Correlation*

**Description**

Compute the correlation of the standardized residual to check the local dependency status.

**Usage**

```
residCor(objFit)
```

**Arguments**

- `objFit`: object of class "fit", the output of `fitStats()`.

**Value**

- `ld_correl`: Correlation matrix of the standradized residual.
- `ld_mean`: The mean of the correlation.
- `ld_lowertri`: The lower triangle of the correlation matrix.

**shortDIF**

*A Shorter Polytomous Dataset with DIF*

**Description**

The artificial dataset of a polytomous responses (three categories) which contains three non-DIF items and a DIF item.

**Usage**

```
data(shortDIF)
```

**Format**

An object of class `data.frame` with 100 rows and 4 columns.
short_poly_data  

**Description**

The artificial dataset of a polytomous responses (three categories) which contains three subsets with different discrimination values. To reproduce this dataset:

```r
short_poly_data <- generate_data(alpha = c(0.02, 0.5, 2), nitem = 3, ndim = 3, ncat = 5, theta = c(-6, 6), beta = c(-4, 4), ntheta = 151)
```

will lead to similar but not the same dataset, due to the randomization.

**Usage**

```r
data(short_poly_data)
```

**Format**

An object of class `data.frame` with 151 rows and 9 columns.

---

stepwise_search  

**Description**

To search itemset that give maximum value of the criterion

**Usage**

```r
stepwise_search(
  X,
  criterion = c("ipoqll", "ipoqlldif"),
  incl_set = c(),
  groups_map = c(),
  cores = NULL,
  optim_control_iq = c(),
  optim_control_oq = c(),
  isTracked = TRUE,
  isContinued = FALSE,
  prevData = c(),
  fileOutput = FALSE,
  tempFile = "temp_stepSearch.RData",
  isConvert = FALSE,
  setting_par_iq = c(),
  setting_par_oq = c(),
```

method = c("fast", "novel")
)

backward_search(
  X,
  criterion = c("ipoqll", "ipoqlldif"),
  incl_set = c(),
  groups_map = c(),
  cores = NULL,
  optim_control_iq = c(),
  optim_control_oq = c(),
  isTracked = TRUE,
  isContinued = FALSE,
  prevData = c(),
  fileOutput = FALSE,
  tempFile = "temp_backSearch.RData",
  isConvert = FALSE,
  setting_par_iq = c(),
  setting_par_oq = c(),
  method = c("fast", "novel")
)

## S3 method for class 'search'
summary(object, ...)

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

plot_search(obj, remOrdered = TRUE, locateMax = TRUE, ...)

Arguments

X    A matrix or data.frame of the observed responses (ordinal or binary response).
criterion    The criterion that should be used. The default is ipoqll.
incl_set    A vector of initial items in the included set to start the search. The default is to start with full items.
groups_map    A matrix or vector to map the subject to the DIFs groups.
cores    An integer value of number of cores should be used for computation. The default is 2.
optim_control_iq    The optimisation setting of the included set. See stats::optim() control parameter.
optim_control_oq    The optimisation setting of the excluded set. See stats::optim() control parameter.
isTracked    A logical value whether the progress need to be tracked or not.
isContinued    A logical value whether this search is continuing another unfinished search.
stepwise_search

prevData    The filename of the temporary .RData file of the unfinished search.
fileOutput  The filename if it is wished to save the output results in file (.RData and .csv) and FALSE if not.
tempFile    The filename of the temporary file to track the search progress. The default is "temp_stepSearch.RData" which also automatically produces "temp_stepSearch.csv".
isConvert   A logical value whether it is wanted to recompute the score of the search results using IPOQ-LL-DIF criterion.
setting_par_iq a list of the optimization control setting parameters for the included set. See setting parameter in autoRaschOptions().
setting_par_oq a list of the optimization control setting parameters for the included set. See setting parameter in autoRaschOptions().
method      The implementation option of log likelihood function. fast using a c++ implementation and novel using an R implementation.
object      The object of class 'search'.
...         Further arguments to be passed.
x           The object of class 'search'.
obj         An object of class "search".
remOrdered  A logical statement whether show the order of the items removal or not.
locateMax   A logical statement whether the location of the maximum score is needed to be marked or not.

Details

To search the itemset that give the maximum score.

Value

Matrix of the highest scores (IQ-LL, OQ-LL, and IPOQ-LL) for every number of items in the included set in the set along with the corresponding itemset.

Examples

## Not run:
search_res <- backward_search(shortDIF, criterion = "ipoqll", incl_set = c(1:4), cores = 2)
plot_search(search_res, type="l")

## End(Not run)
testlets_dataset

Multi-testlets Polytomous Data Set

Description
Generate data set which consist of two bundle items with different level of local dependency effect.

Usage
data(testlets_dataset)

Format
An object of class data.frame with 301 rows and 12 columns.

withinItem_multidim
Within-item Multidimensional Polytomous Data Set

Description
Generate multidimensional dataset with some items relate to more than one dimension.

Usage
data(withinItem_multidim)

Format
An object of class data.frame with 301 rows and 18 columns.
Index

datasets
- correl02_multidim, 9
- correl03_multidim, 9
- correl04_multidim, 10
- correl05_multidim, 10
- correl06_multidim, 10
- dicho_inh_dset, 13
- dicho_md_dset, 13
- poly_inh_dset, 30
- poly_md_dset, 30
- polydif_inh_dset, 29
- short_poly_data, 32
- shortDIF, 31
- testlets_dataset, 35
- withinItem_multidim, 35

autoRasch, 2
autoRasch::autoRaschOptions(), 8
autoRaschOptions, 3
autoRaschOptions(), 18–20, 22, 25, 34

backward_search (stepwise_search), 32

cfa(), 5
check.unidim, 5
checkRel, 5
compute_score, 7
compute_scores (compute_score), 7
correl02_multidim, 9
correl03_multidim, 9
correl04_multidim, 10
correl05_multidim, 10
correl06_multidim, 10
createGroup, 11

dataset, 11
dicho_inh_dset, 13
dicho_md_dset, 13

fitStats, 13
fitStats.pcm (pcm), 22

fitStats.pcmdif (pcm_dif), 24

generate_data, 15
generic_model, 18
gpcm, 12, 19, 20, 21, 24, 26
gpcm_dif, 12, 20, 21, 26

itemfit (fitStats), 13

list, 14, 23, 25

par(), 29
pcm, 12, 20, 21, 22, 24, 26
pcm_dif, 12, 21, 24, 26
personfit (fitStats), 13
plot.default(), 27–29
plot_EVC, 26
plot_fitStats (fitStats), 13
plot_ICC, 27
plot_PImap, 28
plot_search (stepwise_search), 32
poly_inh_dset, 30
poly_md_dset, 30
polydif_inh_dset, 29
print.gpcm (gpcm), 19
print.gpcmdif (gpcm_dif), 20
print.pcm (pcm), 22
print.pcmdif (pcm_dif), 24
print.search (stepwise_search), 32

residCor, 31

short_poly_data, 32
shortDIF, 31
stats::optim(), 4, 8, 33
stats::rnorm(), 15, 16
stepwise_search, 32
summary.fit (fitStats), 13
summary.gpcm (gpcm), 19
summary.gpcmdif (gpcm_dif), 20
summary.pcm (pcm), 22
INDEX

summary.pcmdif (pcm_dif), 24
summary.score (compute_score), 7
summary.search (stepwise_search), 32
summary.seprel (checkRel), 5

testlets_dataset, 35

withinItem_multidim, 35