Package 'catR'

October 12, 2022

Type Package

Title Generation of IRT Response Patterns under Computerized Adaptive Testing

Version 3.17

Date 2022-06-23

Author David Magis (U Liege, Belgium), Gilles Raiche (UQAM, Canada), Juan Ramon Barrada (U Zaragoza, Spain)

Maintainer Cheng Hua <chuabest@gmail.com>

Depends R (>= 4.0.0)

Description Provides routines for the generation of response patterns under unidimensional dichotomous and polytomous computerized adaptive testing (CAT) framework. It holds many standard functions to estimate ability, select the first item(s) to administer and optimally select the next item, as well as several stopping rules. Options to control for item exposure and content balancing are also available (Magis and Barrada (2017) <doi:10.18637/jss.v076.c01>).

License GPL (>= 3)

LazyLoad yes

RoxygenNote 7.2.0

NeedsCompilation no

Repository CRAN

Date/Publication 2022-06-24 08:00:02 UTC

R topics documented:

aStratified 2
breakBank 4
cat_pav 6
checkStopRule 7
eapEst 9
eapSem 12
EPV 16
fullDist 20

GDI . . . 23
genDichoMatrix . . . 27
genPattern . . . 30
genPolyMatrix . . . 32
Ii . . . 37
integrate.catR . . . 39
Ji . . . 40
KL . . . 43
MEI . . . 48
MWI . . . 53
nextItem . . . 57
OIi . . . 68
Pi . . . 71
randomCAT . . . 74
semTheta . . . 87
simulateRespondents . . . 95
startItems . . . 102
tcals . . . 108
test.cbList . . . 109
testList . . . 111
thetaEst . . . 113

Index **120**

`aStratified` *Item membership assessment for a-stratified sampling*

Description

This command allocates each item from the bank into its specified stratum, defined by increasing discrimination parameters for a-stratified sampling.

Usage

```
aStratified(itemBank, K, model = NULL)
```

Arguments

`itemBank`	a matrix or data frame of item parameters. Must refer to either nay dichotomous IRT model or to polytomous GRM, MGRM or GPCM models. See **Details**.
`K`	either a single integer value specifying the number of strata, or an integer vector of required numbers of items per strata.
`model`	either `NULL` (default) for dichotomous models, or any suitable acronym for allowed polytomous models. Possible values are `"GRM"`, `"MGRM"` and `"GPCM"`. See **Details**.

Details

a-stratified sampling (Chang and Ying, 1999) is a content balancing technique that splits the item bank into several strata, the latter being set by increasing discrimination (a) parameters. The first stratum holds the least discriminative items, the second stratum the second set of least discriminative items etc., and the last stratum the most discriminative items. Content balancing item selection is then performed with equiprobable item selection across the strata.

The specification of the strata can be done in two ways, through the `K` argument. First, `K` can take any positive integer value (larger than one and smaller than the number of items): it specifies then the number of strata to create. Each stratum will hold approximately the same number of items, and in case the ratio between the number of items in the bank and `K` is not an integer value, then the last stratum will hold more items than the other ones. Second, `K` can be a vector of integer values that specify the number of items per strata. In this second case, the first stratum will hold as many items as the first component of `K`, the second startum as many items as the second component of `K`, and so on. Note that the sum of `K` shoudl coincide with the number of items in the bank.

Dichotomous IRT models are considered whenever `model` is set to `NULL` (default value). In this case, `it` must be a matrix with one row per item and four columns, with the values of the discrimination, the difficulty, the pseudo-guessing and the inattention parameters (in this order). These are the parameters of the four-parameter logistic (4PL) model (Barton and Lord, 1981).

Polytomous IRT models are specified by their respective acronym: `"GRM"` for Graded Response Model, `"MGRM"` for Modified Graded Response Model and `"GPCM"` for Generalized Partial Credit Model. *Other polytomous IRT models are not allowed for this function and an error message will be returned if the input item bank does not follow one of the three aforementioned models*. The `it` still holds one row per item, end the number of columns and their content depends on the model. See `genPolyMatrix` for further information and illustrative examples of suitable polytomous item banks.

The output is a vector with character values `"Group1"` for the first startum, `"Group2"` for the second stratum etc. It has the same length as the number of items in the `itemBank` matrix and permits one-to-one identification of the items in each stratum.

Value

A vector with as many components as the number of items in the bank, with values `Group1` to `GroupG` where `G` is the number of strata (i.e. either `K`) or the length of vector `K`).

Author(s)

David Magis
Department of Psychology, University of Liege, Belgium
<david.magis@uliege.be>

References

Barton, M.A., and Lord, F.M. (1981). *An upper asymptote for the three-parameter logistic item-response model*. Research Bulletin 81-20. Princeton, NJ: Educational Testing Service.

Chang, H.-H., and Ying, Z. (1999). A-stratified multistage computerized adaptive testing. *Applied Psychological Measurement, 23*, 211-222. doi: 10.1177/01466219922031338

Magis, D. and Barrada, J. R. (2017). Computerized Adaptive Testing with R: Recent Updates of the Package *catR*. *Journal of Statistical Software*, *Code Snippets*, *76(1)*, 1-18. doi: 10.18637/jss.v076.c01

Magis, D., and Raiche, G. (2012). Random Generation of Response Patterns under Computerized Adaptive Testing with the R Package *catR*. *Journal of Statistical Software*, *48 (8)*, 1-31. doi: 10.18637/jss.v048.i08

See Also

genPolyMatrix

Examples

```
## Dichotomous models ##

 # Loading the 'tcals' parameters
 data(tcals)

 # Selecting item parameters only
 tcals <- as.matrix(tcals[,1:4])

 # Creation of five strata with equal length (17 items each)
 aStratified(tcals, 5)

 # Creation of four strata with prespecified lengths
 res <- aStratified(tcals, K = c(20, 25, 10, 30))
 table(res) # as expected


## Polytomous models ##

 # GRM with 100 items
 mat <- genPolyMatrix(100, 4, model = "GRM")

 # Creation of four strata with equal length (25 items each)
 aStratified(mat, 5, model = "GRM")
```

---

breakBank *Breaking the item bank in item parameters and group membership (for content balancing)*

Description

This command "breaks" the item bank in two parts, the item parameters as a numeric matrix, and the group membership of the items as a vector of factor levels. These two elements can be used separately for content balancing purposes, among others.

Usage

```
breakBank(itemBank)
```

Arguments

itemBank	a matrix or data frame with item parameters in the first columns and group membership in the last column.

Details

The breakBank function is useful to split the original item bank in two parts, one holding the item parameters only, and one containing the names of item group membership. The former can then directly be plugged in adequate functions such as thetaEst.

The function works with both dichotomous and polytomous IRT item banks. In both cases, the group membership must be located as the last column of the matrix or data frame.

Note that there is no check that the input is correct (that is, the group membership is located in the last column), as breakBank is mostly devoted to be used at the early stages of the randomCAT function.

Value

A list with two arguments:

itemPar	a numeric matrix with the item parameters.
cbGroup	a vector with factor names of item membership, with one name per item in itemPar element of the output list.

Author(s)

David Magis
Department of Psychology, University of Liege, Belgium
<david.magis@uliege.be>

References

Magis, D. and Barrada, J. R. (2017). Computerized Adaptive Testing with R: Recent Updates of the Package *catR*. *Journal of Statistical Software*, *Code Snippets*, *76(1)*, 1-18. doi: 10.18637/jss.v076.c01

Magis, D., and Raiche, G. (2012). Random Generation of Response Patterns under Computerized Adaptive Testing with the R Package *catR*. *Journal of Statistical Software*, *48 (8)*, 1-31. doi: 10.18637/jss.v048.i08

See Also

randomCAT

Examples

```
## Dichotomous models ##

 # Loading the 'tcals' parameters
 data(tcals)

 # Breaking 'tcals'
 breakBank(tcals)


## Polytomous models ##

 # Creation of the 'cbList' list with arbitrary proportions
 cbList <- list(names =c ("Audio1", "Audio2", "Written1", "Written2",
        "Written3"), props = c(0.1, 0.2, 0.2, 0.2, 0.3))

 # NRM with 100 items
 mat <- genPolyMatrix(100, 4, model = "NRM", cbControl = cbList)

 # Breaking 'mat'
 breakBank(mat)
```

---

cat_pav *Items parameters of the CAT_PAV data set (with item names)*

Description

The CAT-PAV questionnaire (D.O. Santos, 2017) was developed to propose a computer-adaptive assessment of productive and contextualized academic English vocabulary, piloted with students at Iowa State University (USA). Items are presented as two sentences drawn from the academic subset of the COCA corpus, with one common missing word to be identified by test takers. In case of an incorrect first answer, synonyms are exhibited as a hint. Th item is scored 2 if a correct answer was provided from first attempt, 1 if a correct answer was provided after the hint being shown, and 0 in case of an incorrect answer after the hint was shown.

Format

A matrix with 96 rows and three columns, respectively holding the discrimination, first threshold and second threshold under a GPCM calibration of the items of CAT-PAV study. Item names are available as row names, each item name corresponding to the word being tested by the item.

Source

The original items of CAT-PAV were developed by D.O. Santos (2017), field tested at Iowa State University and calibrated using the Jmetrik software (Meyer, 2015).

References

Meyer, P. (2015). *Jmetrik (version 4.0.3)* [Computer software]. Psychomeasurement Systems, LLC.

D.O. Santos, V. (2017). *A computer-adaptive test of of productive and contextualized academic vocabulary breadth in English (CAT-PAV): Development and validation*. Graduate Theses and Dissertations, 16292. Ames, IA: Iowa State University. http://lib.dr.iastate.edu/etd/16292

`checkStopRule` *Checking whether the stopping rule is satisfied*

Description

This command tests whether one of the specified stopping rules is satisfied in order to stop the CAT.

Usage

```
checkStopRule(th, se, N, it = NULL, model = NULL, D = 1, stop)
```

Arguments

`th`	numeric: the current ability estimate.
`se`	numeric: the current standard error estimate.
`N`	numeric: the current number of administered items.
`it`	the matrix of parameters of currently available items (default is `NULL`). Ignored if element `$rule` of `stop` list does not hold the value `"minInfo"`. See **Details**.
`model`	either `NULL` (default) for dichotomous models, or any suitable acronym for polytomous models. Possible values are `"GRM"`, `"MGRM"`, `"PCM"`, `"GPCM"`, `"RSM"` and `"NRM"`. Ignored if element `$rule` of `stop` list does not hold the value `"minInfo"`. See **Details**.
`D`	numeric: the metric constant. Default is `D=1` (for logistic metric); `D=1.702` yields approximately the normal metric (Haley, 1952). Ignored if `model` is not `NULL` or if element `$rule` of `stop` list does not hold the value `"minInfo"`.
`stop`	a list with valuable element names and contents to set the stopping rule, as an input of the `randomCAT` or `simulateRespondents` functions.

Details

The `checkStopRule` function checks whether at least one of the stopping rules was satisfied at the current step of the CAT test process. It mainly serves as an internal application for`randomCAT` function.

The `stop` list must be supplied accordingly to the requested list in the `randomCAT()` and in the `simulateRespondents()` functions.

Three input values must be supplied: `th` as the current ability estimate; `se` as the current standard error value related to `th` estimate; and `N` as the current test length (i.e., number of administered

items). In addition, if the `stop$rule` vector holds the option `"minInfo"`, three additional input value smust be supplied: `it` with the item parameters or all available items in the bank (i.e., previously administered items should not be set as input); `model` to specify the type of IRT model, either dichotomous or polytomous (see `Pi` fir further details); and possibly the scaling constant `D` set to one by default.

All stopping rules are being assessed and if at least one of them is satisfied, the output list will hold the vector of rules's nmaes that were satisfied through the `rule` argument). If none of the stopping rules were satisfied, this `rule` output argument is simply `NULL`.

Value

A list with two arguments:

`decision`	a logical value indicating whether at least one of the stopping rules was satisfied (`TRUE`) or not (`FALSE`).
`rule`	either a vector with the names of the stopping rules that were satisfied, or `NULL` if `decision` is `FALSE`.

Author(s)

David Magis
Department of Psychology, University of Liege, Belgium
<david.magis@uliege.be>

References

Haley, D.C. (1952). *Estimation of the dosage mortality relationship when the dose is subject to error*. Technical report no 15. Palo Alto, CA: Applied Mathematics and Statistics Laboratory, Stanford University.

Magis, D. and Barrada, J. R. (2017). Computerized Adaptive Testing with R: Recent Updates of the Package *catR*. *Journal of Statistical Software*, *Code Snippets*, *76(1)*, 1-18. doi: 10.18637/jss.v076.c01

Magis, D., and Raiche, G. (2012). Random Generation of Response Patterns under Computerized Adaptive Testing with the R Package *catR*. *Journal of Statistical Software*, *48 (8)*, 1-31. doi: 10.18637/jss.v048.i08

See Also

`randomCAT`, `simulateRespondents`, `Pi`

Examples

```
# Creation of a 'stop' list with two possible rules
 stop <- list(rule = c("length", "precision"), thr = c(20, 0.3))

# Example of successful 'length' rule
 checkStopRule(th = 0.35, se = 0.41, N = 20, stop = stop)
```

```
# Example of successful 'precision' rule
 checkStopRule(th = 0.35, se = 0.29, N = 15, stop = stop)

# Example of jointly successful 'length' and 'precision' rules
 checkStopRule(th = 0.35, se = 0.29, N = 20, stop = stop)

# Example without sucessfull rule
 checkStopRule(th = 0.35, se = 0.31, N = 18, stop = stop)

# Creation of a short bank of available items under 2PL
 it <- genDichoMatrix(items = 5, model = "2PL", seed = 1)

# Computation of maximum information at ability level 0.35
maxI <- max(Ii(0.35, it)$Ii)

# Creation of a 'stop' list with four possible rules and too large threshold for 'minInfo'
 stop <- list(rule = c("length", "precision", "classification", "minInfo"),
              thr = c(20, 0.3, 1, maxI-0.01), alpha = 0.05)

# Example with sucessfull 'classification' rule only
 checkStopRule(th = 0.35, se = 0.31, N = 18, it = it, stop = stop)

# Creation of a 'stop' list with four possible rules and too large threshold for 'minInfo'
 stop <- list(rule = c("length", "precision", "classification", "minInfo"),
              thr = c(20, 0.3, 1, maxI+0.01), alpha = 0.05)

# Example with sucessfull 'minInfo' rule only
 checkStopRule(th = 0.35, se = 0.55, N = 18, it = it, stop = stop)
```

`eapEst` *EAP ability estimation (dichotomous and polytomous IRT models)*

Description

This command returns the EAP (expected a posteriori) ability estimate for a given response pattern and a given matrix of item parameters, either under the 4PL model or any suitable polytomous IRT model.

Usage

```
eapEst(it, x, model = NULL, D = 1, priorDist = "norm", priorPar = c(0, 1),
  lower = -4, upper = 4, nqp = 33)
```

Arguments

`it`	numeric: a suitable matrix of item parameters. See **Details**.
`x`	numeric: a vector of item responses.

model	either NULL (default) for dichotomous models, or any suitable acronym for polytomous models. Possible values are "GRM", "MGRM", "PCM", "GPCM", "RSM" and "NRM". See **Details**.
D	numeric: the metric constant. Default is D=1 (for logistic metric); D=1.702 yields approximately the normal metric (Haley, 1952). Ignored if model is not NULL.
priorDist	character: specifies the prior distribution. Possible values are "norm" (default), "unif" and "Jeffreys".
priorPar	numeric: vector of two components specifying the prior parameters (default is c(0,1)). Ignored if priorDist="Jeffreys". See **Details**.
lower	numeric: the lower bound for numercal integration (default is -4).
upper	numeric: the upper bound for numercal integration (default is 4).
nqp	numeric: the number of quadrature points (default is 33).

Details

The EAP (expected a posteriori) ability estimator (Bock and Mislevy, 1982) is obtained by computing the average of the posterior distribution of ability, the latter being set as the prior distribution times the likelihood function.

Dichotomous IRT models are considered whenever model is set to NULL (default value). In this case, it must be a matrix with one row per item and four columns, with the values of the discrimination, the difficulty, the pseudo-guessing and the inattention parameters (in this order). These are the parameters of the four-parameter logistic (4PL) model (Barton and Lord, 1981).

Polytomous IRT models are specified by their respective acronym: "GRM" for Graded Response Model, "MGRM" for Modified Graded Response Model, "PCM" for Partical Credit Model, "GPCM" for Generalized Partial Credit Model, "RSM" for Rating Scale Model and "NRM" for Nominal Response Model. The it still holds one row per item, end the number of columns and their content depends on the model. See genPolyMatrix for further information and illustrative examples of suitable polytomous item banks.

Three prior distributions are available: the normal distribution, the uniform distribution and Jeffreys' prior distribution (Jeffreys, 1939, 1946). The prior distribution is specified by the argument priorPar, with values "norm", "unif" and "Jeffreys", respectively.

The argument priorPar determines either the prior mean and standard deviation of the normal prior distribution (if priorDist="norm"), or the range for defining the prior uniform distribution (if priorDist="unif"). This argument is ignored if priorDist="Jeffreys".

The required integrals are approximated by numerical adaptive quadrature. This is achieved by using the integrate.catR function. Arguments lower, upper and nqp define respectively the lower and upper bounds for numerical integration, and the number of quadrature points. By default, the numerical integration runs with 33 quadrature points on the range [-4; 4], that is, a sequence of values from -4 to 4 by steps of 0.25.

Value

The estimated EAP ability level.

Author(s)

David Magis
Department of Psychology, University of Liege, Belgium
<david.magis@uliege.be>

References

Barton, M.A., and Lord, F.M. (1981). *An upper asymptote for the three-parameter logistic item-response model*. Research Bulletin 81-20. Princeton, NJ: Educational Testing Service.

Bock, R. D., and Mislevy, R. J. (1982). Adaptive EAP estimation of ability in a microcomputer environment. *Applied Psychological Measurement, 6*, 431-444. doi: 10.1177/014662168200600405

Haley, D.C. (1952). *Estimation of the dosage mortality relationship when the dose is subject to error*. Technical report no 15. Palo Alto, CA: Applied Mathematics and Statistics Laboratory, Stanford University.

Jeffreys, H. (1939). *Theory of probability*. Oxford, UK: Oxford University Press.

Jeffreys, H. (1946). An invariant form for the prior probability in estimation problems. *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences, 186*, 453-461.

Magis, D. and Barrada, J. R. (2017). Computerized Adaptive Testing with R: Recent Updates of the Package *catR*. *Journal of Statistical Software*, *Code Snippets*, *76(1)*, 1-18. doi: 10.18637/jss.v076.c01

Magis, D., and Raiche, G. (2012). Random Generation of Response Patterns under Computerized Adaptive Testing with the R Package *catR*. *Journal of Statistical Software*, *48 (8)*, 1-31. doi: 10.18637/jss.v048.i08

See Also

thetaEst, genPolyMatrix, integrate.catR

Examples

```
## Dichotomous models ##

 # Loading the 'tcals' parameters
 data(tcals)

 # Selecting item parameters only
 tcals <- as.matrix(tcals[,1:4])

 # Creation of a response pattern (tcals item parameters,
 # true ability level 0)
 set.seed(1)
 x <- genPattern(0, tcals)

 # EAP estimation, standard normal prior distribution
 eapEst(tcals, x)

 # EAP estimation, uniform prior distribution upon range [-2,2]
```

```
eapEst(tcals, x, priorDist = "unif", priorPar = c(-2, 2))

# EAP estimation, Jeffreys' prior distribution
eapEst(tcals, x, priorDist = "Jeffreys")

# Changing the integration settings
eapEst(tcals, x, nqp = 100)


## Polytomous models ##

# Generation of an item bank under GRM with 100 items and at most 4 categories
m.GRM <- genPolyMatrix(100, 4, "GRM")
m.GRM <- as.matrix(m.GRM)

# Creation of a response pattern (true ability level 0)
set.seed(1)
x <- genPattern(0, m.GRM, model = "GRM")

# EAP estimation, standard normal prior distribution
eapEst(m.GRM, x, model = "GRM")

# EAP estimation, uniform prior distribution upon range [-2,2]
eapEst(m.GRM, x, model = "GRM", priorDist = "unif", priorPar = c(-2, 2))

# EAP estimation, Jeffreys' prior distribution
eapEst(m.GRM, x, model = "GRM", priorDist = "Jeffreys")


# Loading the cat_pav data
data(cat_pav)
cat_pav <- as.matrix(cat_pav)

# Creation of a response pattern (true ability level 0)
set.seed(1)
x <- genPattern(0, cat_pav, model = "GPCM")

# EAP estimation, standard normal prior distribution
eapEst(cat_pav, x, model = "GPCM")

# EAP estimation, uniform prior distribution upon range [-2,2]
eapEst(cat_pav, x, model = "GPCM", priorDist = "unif", priorPar = c(-2, 2))

# EAP estimation, Jeffreys' prior distribution
eapEst(cat_pav, x, model = "GPCM", priorDist = "Jeffreys")
```

---

eapSem *Standard error of EAP ability estimation (dichotomous and polytomous IRT models)*

Description

This command returns the estimated standard error of the ability estimate, for a given response pattern and a given matrix of item parameters, either under the 4PL model or any suitable polytomous IRT model.

Usage

```
eapSem(thEst, it, x, model = NULL, D = 1, priorDist = "norm",
  priorPar = c(0, 1), lower = -4, upper = 4, nqp = 33)
```

Arguments

`thEst`	numeric: the EAP ability estimate.
`it`	numeric: a suitable matrix of item parameters. See **Details**.
`x`	numeric: a vector of item responses.
`model`	either `NULL` (default) for dichotomous models, or any suitable acronym for polytomous models. Possible values are `"GRM"`, `"MGRM"`, `"PCM"`, `"GPCM"`, `"RSM"` and `"NRM"`. See **Details**.
`D`	numeric: the metric constant. Default is `D=1` (for logistic metric); `D=1.702` yields approximately the normal metric (Haley, 1952). Ignored if `model` is not `NULL`.
`priorDist`	character: specifies the prior distribution. Possible values are `"norm"` (default), `"unif"` and `"Jeffreys"`.
`priorPar`	numeric: vector of two components specifying the prior parameters (default is `c(0,1)`). Ignored if `priorDist="Jeffreys"`. See **Details**.
`lower`	numeric: the lower bound for numercal integration (default is -4).
`upper`	numeric: the upper bound for numercal integration (default is 4).
`nqp`	numeric: the number of quadrature points (default is 33).

Details

This command computes the standard error of the EAP (expected a posteriori) ability estimator (Bock and Mislevy, 1982).

Dichotomous IRT models are considered whenever `model` is set to `NULL` (default value). In this case, `it` must be a matrix with one row per item and four columns, with the values of the discrimination, the difficulty, the pseudo-guessing and the inattention parameters (in this order). These are the parameters of the four-parameter logistic (4PL) model (Barton and Lord, 1981).

Polytomous IRT models are specified by their respective acronym: `"GRM"` for Graded Response Model, `"MGRM"` for Modified Graded Response Model, `"PCM"` for Partical Credit Model, `"GPCM"` for Generalized Partial Credit Model, `"RSM"` for Rating Scale Model and `"NRM"` for Nominal Response Model. The `it` still holds one row per item, end the number of columns and their content depends on the model. See `genPolyMatrix` for further information and illustrative examples of suitable polytomous item banks.

Three prior distributions are available: the normal distribution, the uniform distribution and Jeffreys' prior distribution (Jeffreys, 1939, 1946). The prior distribution is specified by the argument `priorPar`, with values `"norm"`, `"unif"` and `"Jeffreys"`, respectively.

The argument `priorPar` determines either the prior mean and standard deviation of the normal prior distribution (if `priorDist="norm"`), or the range for defining the prior uniform distribution (if `priorDist="unif"`). This argument is ignored if `priorDist="Jeffreys"`.

The required integrals are approximated by numerical adaptive quadrature. This is achieved by using the `integrate.catR` function. Arguments `lower`, `upper` and `nqp` define respectively the lower and upper bounds for numerical integration, and the number of quadrature points. By default, the numerical integration runs with 33 quadrature points on the range [-4; 4], that is, a sequence of values from -4 to 4 by steps of 0.25.

Note that in the current version, the EAP ability estimate must be specified through the `thEst` argument.

Value

The estimated standard error of the EAP ability level.

Author(s)

David Magis
Department of Psychology, University of Liege, Belgium
<david.magis@uliege.be>

References

Barton, M.A., and Lord, F.M. (1981). *An upper asymptote for the three-parameter logistic item-response model*. Research Bulletin 81-20. Princeton, NJ: Educational Testing Service.

Bock, R. D., and Mislevy, R. J. (1982). Adaptive EAP estimation of ability in a microcomputer environment. *Applied Psychological Measurement, 6*, 431-444. doi: 10.1177/014662168200600405

Haley, D.C. (1952). *Estimation of the dosage mortality relationship when the dose is subject to error*. Technical report no 15. Palo Alto, CA: Applied Mathematics and Statistics Laboratory, Stanford University.

Jeffreys, H. (1939). *Theory of probability*. Oxford, UK: Oxford University Press.

Jeffreys, H. (1946). An invariant form for the prior probability in estimation problems. *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences, 186*, 453-461.

Magis, D. and Barrada, J. R. (2017). Computerized Adaptive Testing with R: Recent Updates of the Package *catR*. *Journal of Statistical Software*, *Code Snippets*, *76(1)*, 1-18. doi: 10.18637/jss.v076.c01

Magis, D., and Raiche, G. (2012). Random Generation of Response Patterns under Computerized Adaptive Testing with the R Package *catR*. *Journal of Statistical Software*, *48 (8)*, 1-31. doi: 10.18637/jss.v048.i08

See Also

`thetaEst`, `genPolyMatrix`, `integrate.catR`

Examples

```
## Dichotomous models ##

 # Loading the 'tcals' parameters
 data(tcals)

 # Selecting item parameters only
 tcals <- as.matrix(tcals[,1:4])

 # Creation of a response pattern (tcals item parameters,
 # true ability level 0)
 set.seed(1)
 x <- genPattern(0, tcals)

 # EAP estimation, standard normal prior distribution
 th <- eapEst(tcals, x)
 c(th, eapSem(th, tcals, x))

 # EAP estimation, uniform prior distribution upon range [-2,2]
 th <- eapEst(tcals, x, priorDist = "unif", priorPar = c(-2, 2))
 c(th, eapSem(th, tcals, x, priorDist = "unif", priorPar=c(-2, 2)))

 # EAP estimation, Jeffreys' prior distribution
 th <- eapEst(tcals, x, priorDist = "Jeffreys")
 c(th, eapSem(th, tcals, x, priorDist = "Jeffreys"))

## Not run:

## Polytomous models ##

 # Generation of an item bank under GRM with 100 items and at most 4 categories
 m.GRM <- genPolyMatrix(100, 4, "GRM")
 m.GRM <- as.matrix(m.GRM)

 # Creation of a response pattern (true ability level 0)
 set.seed(1)
 x <- genPattern(0, m.GRM, model = "GRM")

 # EAP estimation, standard normal prior distribution
 th <- eapEst(m.GRM, x, model = "GRM")
 c(th, eapSem(th, m.GRM, x, model = "GRM"))

 # EAP estimation, uniform prior distribution upon range [-2,2]
 th <- eapEst(m.GRM, x, model = "GRM", priorDist = "unif", priorPar = c(-2, 2))
 c(th, eapSem(th, m.GRM, x, model = "GRM", priorDist = "unif", priorPar = c(-2, 2)))

 # EAP estimation, Jeffreys' prior distribution
 th <- eapEst(m.GRM, x, model = "GRM", priorDist = "Jeffreys")
 c(th, eapSem(th, m.GRM, x, model = "GRM", priorDist = "Jeffreys"))
```

```
# Loading the cat_pav data
 data(cat_pav)
 cat_pav <- as.matrix(cat_pav)

 # Creation of a response pattern (true ability level 0)
 set.seed(1)
 x <- genPattern(0, cat_pav, model = "GPCM")

 # EAP estimation, standard normal prior distribution
 th <- eapEst(cat_pav, x, model = "GPCM")
 c(th, eapSem(th, cat_pav, x, model = "GPCM"))

 # EAP estimation, uniform prior distribution upon range [-2,2]
 th <- eapEst(cat_pav, x, model = "GPCM", priorDist = "unif", priorPar = c(-2, 2))
 c(th, eapSem(th, cat_pav, x, model = "GPCM", priorDist = "unif", priorPar = c(-2, 2)))

 # EAP estimation, Jeffreys' prior distribution
 th <- eapEst(cat_pav, x, model = "GPCM", priorDist = "Jeffreys")
 c(th, eapSem(th, cat_pav, x, model = "GPCM", priorDist = "Jeffreys"))

## End(Not run)
```

EPV *Expected Posterior Variance (EPV)*

Description

This command returns the expected posterior variance (EPV) for a given item under dichotomous and polytomous IRT models, as used for Minimum Expected Posterior Variance (MEPV) criterion.

Usage

```
EPV(itemBank, item, x, theta, it.given, model = NULL, priorDist = "norm",
  priorPar = c(0, 1), D = 1, parInt = c(-4, 4, 33))
```

Arguments

itemBank	numeric: a suitable matrix of item parameters. See **Details**.
item	numeric: the item (referred to as its rank in the item bank) for which the expected posterior variance must be computed.
x	numeric: a vector of item responses, coded as 0 or 1 only (for dichotomous items) or from 0 to the number of response categories minus one (for polytomous items).
theta	numeric: the provisional ability estimate.
it.given	numeric: a suitable matrix of item parameters for previously administered items. The number of rows of `it.given` must be equal to the length of `x`.

`model`	either `NULL` (default) for dichotomous models, or any suitable acronym for polytomous models. Possible values are `"GRM"`, `"MGRM"`, `"PCM"`, `"GPCM"`, `"RSM"` and `"NRM"`. See **Details**.
`priorDist`	character: specifies the prior distribution. Possible values are `"norm"` (default) and `"unif"`.
`priorPar`	numeric: vector of two components specifying the prior parameters (default is `c(0,1)`) of the prior ability distribution.
`D`	numeric: the metric constant. Default is `D=1` (for logistic metric); `D=1.702` yields approximately the normal metric (Haley, 1952). Ignored if `model` is not `NULL`.
`parInt`	numeric: vector of three components, defining the sequence of ability values for computing the posterior variance. See **Details**.

Details

The EPV can be used as a rule for selecting the next item in the CAT process (Choi and Swartz, 2009; Owen, 1975; van der Linden, 1998). This command serves as a subroutine for the `nextItem` function.

Dichotomous IRT models are considered whenever `model` is set to `NULL` (default value). In this case, `itemBank` must be a matrix with one row per item and four columns, with the values of the discrimination, the difficulty, the pseudo-guessing and the inattention parameters (in this order). These are the parameters of the four-parameter logistic (4PL) model (Barton and Lord, 1981).

Polytomous IRT models are specified by their respective acronym: `"GRM"` for Graded Response Model, `"MGRM"` for Modified Graded Response Model, `"PCM"` for Partical Credit Model, `"GPCM"` for Generalized Partial Credit Model, `"RSM"` for Rating Scale Model and `"NRM"` for Nominal Response Model. The `itemBank` still holds one row per item, end the number of columns and their content depends on the model. See `genPolyMatrix` for further information and illustrative examples of suitable polytomous item banks.

Under polytomous IRT models, let *k* be the number of administered items, and set $x_1, ..., x_k$ as the provisional response pattern (where each response x_l takes values in $\{0, 1, ..., g_l\}$). Set $\hat{\theta}_k$ as the provisional ability estimate (with the first *k* responses) and let *j* be the item of interest (not previously administered). Set also $P_{jt}(\theta)$ as the probability of answering response category t to item *j* for a given ability level θ (thus $t \in \{0, ..., g_j\}$). Finally, set $Var(\theta|x_1, ..., x_k, t)$ as the posterior variance of θ, given the provisional response pattern (updated by response t). Then, the EPV for item *j* equals

$$EPV_j = \sum_{t=0}^{g_j} P_{jt}(\hat{\theta}_k)\, Var(\theta|x_1, ..., x_k, t)$$

.

In case of dichotomous IRT models, all g_l values reduce to 1, so that item responses x_l equal either 0 or 1. Set simply $P_j(\theta)$ as the probability of answering item *j* correctly for a given ability level θ, and set $Q_j(\theta) = 1 - P_j(\theta)$. Finally, set $Var(\theta|x_1, ..., x_k, 0)$ and $Var(\theta|x_1, ..., x_k, 1)$ as the posterior variances of θ, given the provisional response pattern (updated by response 0 and 1 respectively). Then, the EPV for item *j* reduces to

$$EPV_j = P_j(\hat{\theta}_k)\, Var(\theta|x_1, ..., x_k, 1) + Q_j(\hat{\theta}_k)\, Var(\theta|x_1, ..., x_k, 0)$$

.

The posterior variances $Var(\theta|x_1, ..., x_k, x_j)$ (where x_j takes value 0 or 1 for dichotomous models, and 0, 1, ..., or g_j for polytomous models) is computed as the squared standard error of the EAP estimate of ability, using the response pattern $(x_1, ..., x_k, x_j)$. This is done by a joint use of the `eapEst` and `eapSem` functions.

The prior distribution is set up by the arguments `priorDist` and `priorPar`, with the by-default standard normal distribution. The range of integration is defined by the `parInt` argument, with by default, the sequence from -4 to 4 and of length 33 (or, by steps of 0.25). See the function `eapEst` for further details.

The provisional response pattern and the related item parameters are provided by the arguments `x` and `it.given` respectively. The target item (for which the maximum information computed) is given by its number in the item bank, through the `item` argument.

Note that the provisional response pattern `x` can also be set to `NULL` (which is useful when the number `nrItems` of starting items is set to zero). In this case, `it.given` must be a matrix with zero rows, such as e.g., `itemBank[NULL,]`.

Value

The expected posterior variance for the selected item.

Author(s)

David Magis
Department of Psychology, University of Liege, Belgium
<david.magis@uliege.be>

References

Barton, M.A., and Lord, F.M. (1981). *An upper asymptote for the three-parameter logistic item-response model*. Research Bulletin 81-20. Princeton, NJ: Educational Testing Service.

Choi, S. W., and Swartz, R. J. (2009). Comparison of CAT item selection criteria for polytomous items. *Applied Psychological Measurement, 32*, 419-440. doi: 10.1177/0146621608327801

Haley, D.C. (1952). *Estimation of the dosage mortality relationship when the dose is subject to error*. Technical report no 15. Palo Alto, CA: Applied Mathematics and Statistics Laboratory, Stanford University.

Magis, D. and Barrada, J. R. (2017). Computerized Adaptive Testing with R: Recent Updates of the Package *catR*. *Journal of Statistical Software*, *Code Snippets*, *76(1)*, 1-18. doi: 10.18637/jss.v076.c01

Magis, D., and Raiche, G. (2012). Random Generation of Response Patterns under Computerized Adaptive Testing with the R Package *catR*. *Journal of Statistical Software*, *48 (8)*, 1-31. doi: 10.18637/jss.v048.i08

Owen, R. J. (1975). A Bayesian sequential procedure for quantal response in the context of adaptive mental testing. *Journal of the American Statistical Association, 70*, 351-356. doi: 10.1080/01621459.1975.10479871

van der Linden, W. J. (1998). Bayesian item selection criteria for adaptive testing. *Psychometrika, 63*, 201-216. doi: 10.1007/BF02294775

See Also

nextItem, eapEst, eapSem, genPolyMatrix

Examples

```
## Dichotomous models ##

 # Loading the 'tcals' parameters
 data(tcals)

 # Selecting item parameters only
 bank <- as.matrix(tcals[,1:4])

 # Selection of two arbitrary items (15 and 20) of the
 # 'tcals' data set
 it.given <- bank[c(15,20),]

 # Creation of a response pattern
 x <- c(0, 1)

 # EPV for item 1, provisional ability level 0
 EPV(bank, 1, x, 0, it.given)

 # With prior standard deviation 2
 EPV(bank, 1, x, 0, it.given, priorPar = c(0,2))


## Polytomous models ##

 # Generation of an item bank under GRM with 100 items and at most 4 categories
 m.GRM <- genPolyMatrix(100, 4, "GRM")
 m.GRM <- as.matrix(m.GRM)

 # Selection of two arbitrary items (15 and 20)
 it.given <- m.GRM[c(15,20),]

 # Generation of a response pattern (true ability level 0)
 x <- genPattern(0, it.given, model = "GRM")

 # EPV for item 1, provisional ability level 0
 EPV(m.GRM, 1, x, 0, it.given, model = "GRM")

 # With prior standard deviation 2
 EPV(m.GRM, 1, x, 0, it.given, model = "GRM", priorPar = c(0, 2))


 # Loading the cat_pav data
 data(cat_pav)
 cat_pav <- as.matrix(cat_pav)

 # Selection of two arbitrary items (15 and 20)
```

```
it.given <- cat_pav[c(15, 20),]

# Generation of a response pattern (true ability level 0)
x <- genPattern(0, it.given, model = "GPCM")

# EPV for item 1, provisional ability level 0
EPV(cat_pav, 1, x, 0, it.given, model = "GPCM")

# With prior standard deviation 2
EPV(cat_pav, 1, x, 0, it.given, model = "GPCM", priorPar = c(0, 2))
```

`fullDist` *Full distribution of ability estimator (dichotomous models only)*

Description

This command returns the full distribution of the selected ability estimator, that is, the set of all possible ability estimates and related probabilities, for a given matrix of item parameters and ability estimate (or true value).

Usage

```
fullDist(th, it, method = "BM", priorDist = "norm", priorPar = c(0,1),
  weight = "Huber", tuCo  = 1, range = c(-4 ,4), parInt = c(-4, 4, 33))
```

Arguments

`th`	numeric: the ability estimate of interest (can be a vector of estimates too).
`it`	numeric: a suitable matrix of item parameters. See **Details**.
`method`	character: the ability estimator. Possible values are `"BM"` (default), `"ML"`, `"WL"`, `"EAP"` and `"ROB"`. See **Details**.
`priorDist`	character: specifies the prior distribution. Possible values are `"norm"` (default), `"unif"` and `"Jeffreys"`. Ignored if `method` is neither `"BM"` nor `"EAP"`. See **Details**.
`priorPar`	numeric: vector of two components specifying the prior parameters (default is `c(0,1)`) of the prior ability distribution. Ignored if `method` is neither `"BM"` nor `"EAP"`, or if `priorDist="Jeffreys"`. See **Details**.
`weight`	character: the type of weight function for the robust estimator. Possible values are `"Huber"` (default) and `"Tukey"`. Ignored if `method` is not `"ROB"` or if `model` is not `NULL`. See **Details**.
`tuCo`	numeric: the value of the tuning constant for the weight function (default is 1, suitable with `"Huber"` weight). Ignored if `method` is not `"ROB"` or if `model` is not `NULL`. See **Details**.

`range` numeric: vector of two components specifying the range wherein the ability estimate must be looked for (default is `c(-4,4)`). Ignored if `method=="EAP"`.

`parInt` numeric: vector of three components, holding respectively the values of the arguments `lower`, `upper` and `nqp` of the `eapEst` command. Default vector is (-4, 4, 33). Ignored if `method` is not `"EAP"`.

Details

The computation of the full distribution of an ability estimator is required to determine the exact standard error of ability, compared to the asympotic SE (ASE) displayed by former versions of the `semTheta` function. This distribution is computed as follows.

First, all possible response patterns (for the given test length fixed by the number of items specified by `it` argument) are generated through the internal `dataGen` function. Second, ability estimation is performed with each generated pattern, and the corresponding pattern probability is computed using the set of item parameters `it` and the predefined ability level `th` of interest. These to components are eventually returned as the full distribution.

In case of the Rasch (1PL) model, this long process can be shortened as the total test score is a proxy for ability estimation. Hence, with n items, only $(n + 1)$ patterns must be created, one for each test score from zero to n. Related probabilities can be derived using the Lord-Wingersky algorithm (1984) that is implemented internally through the `LW()` function.

Dichotomous IRT models are considered whenever `model` is set to `NULL` (default value). In this case, `it` must be a matrix with one row per item and four columns, with the values of the discrimination, the difficulty, the pseudo-guessing and the inattention parameters (in this order). These are the parameters of the four-parameter logistic (4PL) model (Barton and Lord, 1981).

Five ability estimators are available: the maximum likelihood (ML) estimator (Lord, 1980), the Bayes modal (BM) estimator (Birnbaum, 1969), the expected a posteriori (EAP) estimator (Bock and Mislevy, 1982), the weighted likelihood (WL) estimator (Warm, 1989) and the robust estimator (Schuster & Yuan, 2011). The selected estimator is specified by the `method` argument, with values `"ML"`, `"BM"`, `"EAP"`, `"WL"` and `"ROB"` respectively.

For the BM and EAP estimators, three prior distributions are available: the normal distribution, the uniform distribution and the Jeffreys' prior distribution (Jeffreys, 1939, 1946). The prior distribution is specified by the argument `priorPar`, with values `"norm"`, `"unif"` and `"Jeffreys"`, respectively. The `priorPar` argument is ignored if `method="ML"` or `method="WL"`.

The argument `priorPar` determines either: the prior mean and standard deviation of the normal prior distribution (if `priorDist="norm"`), or the range for defining the prior uniform distribution (if `priorDist="unif"`). This argument is ignored if `priorDist="Jeffreys"`.

The `eapPar` argument sets the range and the number of quadrature points for numerical integration in the EAP process. By default, it takes the vector value (-4, 4, 33), that is, 33 quadrature points on the range [-4; 4] (or, by steps of 0.25). See `eapEst` for further details.

Robust estimation requires an appropriate weight function that depends on an accurate tuning constant. Suggested functions are the Huber weight (Schuester and Yuan, 2011) and the Tukey weight (Mosteller and Tukey, 1977). Both can be set by the `weight` argument, with respective values `"Huber"` and `"Tukey"`. Default function is Huber. Moreover, the `tuCo` argument specifies the tuning constant for the weight function. Default value is 1 and suggested for Huber weight (also by default), and value 4 is suggested for Tukey weight (Schuester and Yuan, 2011).

The ability level of interest (that is, for which the probability distribution must be computed) is specified by the `th` argument. Note that it can hold a vector of ability levels too; in this case, the probability distribution is computed for each component of `th`.

Value

A matrix with $(t + 1)$ columns (where t is the length of vector `th`), the first column holding all observable ability estimates and columns 2 to $(t+1)$ with related probabilities of the corresponding components of `th`.

Note

Under the Rasch (1PL) model, the computation of the full distribution is very efficient even for a large test: due to the Lord-Wingersky algorithm, only $(n + 1)$ patterns (where n is the number of items) must be created, one for each possible test score. But for other models, 2^n patterns must be generated and used for ability estimation, which can become computationally intensive with long tests. Therefore it is recommended not to make use of this function with more than ten items (except under the Rasch model).

Author(s)

David Magis
Department of Psychology, University of Liege, Belgium
<david.magis@uliege.be>

Lianne Ippel
Department of Psychology, University of Liege, Belgium
<g.j.e.ippel@gmail.com>

References

Barton, M.A., and Lord, F.M. (1981). *An upper asymptote for the three-parameter logistic item-response model*. Research Bulletin 81-20. Princeton, NJ: Educational Testing Service.

Birnbaum, A. (1969). Statistical theory for logistic mental test models with a prior distribution of ability. *Journal of Mathematical Psychology, 6*, 258-276. doi: 10.1016/00222496(69)900054

Bock, R. D., and Mislevy, R. J. (1982). Adaptive EAP estimation of ability in a microcomputer environment. *Applied Psychological Measurement, 6*, 431-444. doi: 10.1177/014662168200600405

Haley, D.C. (1952). *Estimation of the dosage mortality relationship when the dose is subject to error*. Technical report no 15. Palo Alto, CA: Applied Mathematics and Statistics Laboratory, Stanford University.

Jeffreys, H. (1939). *Theory of probability*. Oxford, UK: Oxford University Press.

Jeffreys, H. (1946). An invariant form for the prior probability in estimation problems. *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences, 186*, 453-461.

Lord, F.M. (1980). *Applications of item response theory to practical testing problems*. Hillsdale, NJ: Lawrence Erlbaum.

Lord, F. M., and Wingersky, M. S. (1984). Comparison of IRT true-score and equipercentile observed-score equatings. *Applied Psychological Measurement, 8*, 453-461. doi: 10.1177/014662168400800409

Magis, D. and Barrada, J. R. (2017). Computerized Adaptive Testing with R: Recent Updates of the Package *catR*. *Journal of Statistical Software*, *Code Snippets*, *76(1)*, 1-18. doi: 10.18637/jss.v076.c01

Magis, D., and Raiche, G. (2012). Random Generation of Response Patterns under Computerized Adaptive Testing with the R Package *catR*. *Journal of Statistical Software*, *48 (8)*, 1-31. doi: 10.18637/jss.v048.i08

Mosteller, F., and Tukey, J. (1977). *Exploratory data analysis and regression*. Reading, MA: Addison-Wesley.

Schuester, C., and Yuan, K.-H. (2011). Robust estimation of latent ability in item response models. *Journal of Educational and Behavioral Statistics, 36*, 720)735.doi: 10.3102/1076998610396890

Warm, T.A. (1989). Weighted likelihood estimation of ability in item response models. *Psychometrika, 54*, 427-450. doi: 10.1007/BF02294627

See Also

semTheta, thetaEst,

Examples

```
## Dichotomous models ##

 # Generation of ten items under 1PL model
 it <- genDichoMatrix(10, model = "1PL")

 # Full distribution of ML estimator for ability level zero
 fullDist(0, it, method = "ML")

 # Idem with BM estimator (probabilities don't change, only estimated abilities)
 fullDist(0, it, method = "BM")

 # Idem with ability level 1 (only probabilities change)
 fullDist(1, it, method = "BM")

 # Distributions with two ability levels 1 and 0.5
 fullDist(c(1, 0.5), it, method = "BM")

 # Generation of ten items under 2PL model
 it2 <- genDichoMatrix(10, model = "2PL")

 # Full distribution of ML estimator for ability level zero
 fullDist(0, it2, method = "ML")
```

GDI *Global-discrimination index (GDI) and posterior global-discrimination index (GDIP) for item selection*

Description

This command returns the value of the global-discrimination index (GDI) and posterior global-discrimination index (GDIP) for a given item, an item bank and a set of previously administered items.

Usage

```
GDI(itemBank, item, x, it.given, model = NULL, lower = -4, upper = 4, nqp = 33,
  type = "GDI", priorDist="norm", priorPar = c(0, 1), D = 1, X = NULL,
  lik = NULL)
```

Arguments

itemBank	numeric: a suitable matrix of item parameters. See **Details**.
item	numeric: the item (referred to as its rank in the item bank) for which the GDI or GDIP must be computed.
x	numeric: a vector of item responses, coded as 0 or 1 only (for dichotomous items) or from 0 to the number of response categories minus one (for polytomous items).
it.given	numeric: a matrix with one row per item and four columns, with the values of the discrimination, the difficulty, the pseudo-guessing and the inattention parameters (in this order). The number of rows of it must be equal to the length of x.
model	now only NULL (default) for dichotomous models is allowed.
lower	numeric: the lower bound for numercal integration (default is -4).
upper	numeric: the upper bound for numercal integration (default is 4).
nqp	numeric: the number of quadrature points (default is 33).
type	character: the type of index to be computed. Possible values are "GDI" (default) and "GDIP". See **Details**.
priorDist	character: the prior ability distribution. Possible values are "norm" (default) for the normal distribution, and "unif" for the uniform distribution. Ignored if type is "GDI".
priorPar	numeric: a vector of two components with the prior parameters. If priorDist is "norm", then priorPar contains the mean and the standard deviation of the normal distribution. If priorDist is "unif", then priorPar contains the bounds of the uniform distribution. The default values are 0 and 1 respectively. Ignored if type is "GDI".
D	numeric: the metric constant. Default is D=1 (for logistic metric); D=1.702 yields approximately the normal metric (Haley, 1952).
X	either a vector of numeric values or NULL (default). See **Details**.
lik	either a vector of numeric values or NULL (default). See **Details**.

Details

Global-discrimination index can be used as a rule for selecting the next item in the CAT process (Kaplan, de la Torre, and Barrada, 2015). This command serves as a subroutine for the `nextItem` function.

Dichotomous IRT models are considered whenever `model` is set to `NULL` (default value). In this case, `itemBank` must be a matrix with one row per item and four columns, with the values of the discrimination, the difficulty, the pseudo-guessing and the inattention parameters (in this order). These are the parameters of the four-parameter logistic (4PL) model (Barton and Lord, 1981).

Currently both GDI and GDIP are not implemented for polytomous IRT models.

The integrals within GDI and GDIP are approximated by the `integrate.catR` function. The range of integration is set up by the arguments `lower`, `upper` and `nqp`, giving respectively the lower bound, the upper bound and the number of quadrature points. The default range goes from -4 to 4 with length 33 (that is, by steps of 0.25).

To speed up the computation, both the range of integration of values θ and the values of the likelihood function $L(\theta)$ can be directly provided to the function through the arguments `X` and `lik`. If `X` is set to `NULL` (default), the sequence of ability values for integration is determined by the arguments `lower`, `upper` and `nqp` as explained above. If `lik` is `NULL` (default), it is also internally computed from an implementation of the likelihood function.

The provisional response pattern and the related item parameters are provided by the arguments `x` and `it.given` respectively. The target item (for which the KL information is computed) is given by its rank number in the item bank, through the `item` argument.

The argument `type` defines the type of KL information to be computed. The default value, `"GDI"`, computes the GDI indexinformation, while the posterior GDI index is obtained with `type="GDIP"`. For the latter, the `priorDist` and `priorPar` arguments fix the prior ability distribution. The normal distribution is set up by `priorDist="norm"` and then, `priorPar` contains the mean and the standard deviation of the normal distribution. If `priorDist` is `"unif"`, then the uniform distribution is considered, and `priorPar` fixes the lower and upper bounds of that uniform distribution. By default, the standard normal prior distribution is assumed. These arguments are ignored whenever `method` is `"GDI"`.

Value

The required GDI or GDIP value for the selected item.

Author(s)

David Magis
Department of Psychology, University of Liege, Belgium
<david.magis@uliege.be>

Juan Ramon Barrada
Department of Psychology and Sociology, Universidad Zaragoza, Spain
<barrada@unizar.es>

References

Barton, M.A., and Lord, F.M. (1981). *An upper asymptote for the three-parameter logistic item-response model*. Research Bulletin 81-20. Princeton, NJ: Educational Testing Service.

Haley, D.C. (1952). *Estimation of the dosage mortality relationship when the dose is subject to error*. Technical report no 15. Palo Alto, CA: Applied Mathematics and Statistics Laboratory, Stanford University.

Kaplan, M., de la Torre, J., and Barrada, J. R. (2015). New item selection methods for cognitive diagnosis computerized adaptive testing. *Applied Psychological Measurement, 39*, 167-188. doi: 10.1177/0146621614554650

Magis, D. and Barrada, J. R. (2017). Computerized Adaptive Testing with R: Recent Updates of the Package *catR*. *Journal of Statistical Software*, *Code Snippets*, *76(1)*, 1-18. doi: 10.18637/jss.v076.c01

Magis, D., and Raiche, G. (2012). Random Generation of Response Patterns under Computerized Adaptive Testing with the R Package *catR*. *Journal of Statistical Software*, *48 (8)*, 1-31. doi: 10.18637/jss.v048.i08

See Also

integrate.catR, nextItem, genPolyMatrix

Examples

```
## Dichotomous models ##

 # Loading the 'tcals' parameters
 data(tcals)

 # Selecting item parameters only
 bank <- as.matrix(tcals[,1:4])

 # Selection of two arbitrary items (15 and 20) of the
 # 'tcals' data set
 it.given <- bank[c(15, 20),]

 # Creation of a response pattern
 x <- c(0, 1)

 # GDI for item 1
 GDI(bank, 1, x, it.given)

 # GDIP for item 1
 GDI(bank, 1, x, it.given, type = "GDIP")

 # GDIP for item 1, different integration range
 GDI(bank, 1, x, it.given, type = "GDIP", lower = -2, upper = 2, nqp = 20)

 # GDIP for item 1, uniform prior distribution on the range [-2,2]
 GDI(bank, 1, x, it.given, type = "GDIP", priorDist = "unif",
    priorPar = c(-2, 2))

 # Computation of likelihood function beforehand
 L <- function(th, r, param)
  prod(Pi(th, param)$Pi^r * (1 - Pi(th,param)$Pi)^(1 - r))
```

```
xx <- seq(from = -4, to = 4, length = 33)
y <- sapply(xx, L, x, it.given)
GDI(bank, 1, x, it.given, X = xx, lik = y)
```

---

genDichoMatrix *Item bank generation (dichotomous models)*

Description

This command generates an item bank from prespecified parent distributions for use with dichotomous IRT models. Subgroups of items can also be specified for content balancing purposes.

Usage

```
genDichoMatrix(items = 100, cbControl = NULL, model = "4PL",
  aPrior = c("norm", 1, 0.2), bPrior = c("norm", 0, 1),
  cPrior = c("unif", 0, 0.25), dPrior = c("unif", 0.75, 1), seed = 1)
```

Arguments

items	integer: the number of items to include in the generated item bank.
cbControl	either a list to define subgroups for content balancing or NULL (default). See **Details**.
model	character: the name of the logistic IRT model, with possible values "1PL", "2PL", "3PL" or "4PL" (default).
aPrior	vector of three components, specifying the prior distribution and item parameters for generating the item discrimination levels. See **Details**.
bPrior	vector of three components, specifying the prior distribution and item parameters for generating the item difficulty levels. See **Details**.
cPrior	vector of three components, specifying the prior distribution and item parameters for generating the item lower asymptote levels. See **Details**.
dPrior	vector of three components, specifying the prior distribution and item parameters for generating the item upper asymptote levels. See **Details**.
seed	numeric: the random seed number for the generation of item parameters (default is 1). See set.seed for further details.

Details

This function permits to generate an item bank under dichotomous IRT models that is compatible for use with randomCAT.

The number of items to be included in the bank is specified by the items argument. Corresponding item parameters are drawn from distributions to be specified by arguments aPrior, bPrior, cPrior and dPrior for respective parameters a_i, b_i, c_i and d_i (Barton and Lord, 1981). Each of these

arguments is of length 3, the first component containing the name of the distribution and the last two components coding the distribution parameters.

Possible distributions are:

- the *normal distribution* $N(\mu, \sigma^2)$, available for parameters a_i and b_i. It is specified by `"norm"` as first argument while the latter two arguments contain the values of μ and σ respectively.
- the *log-normal distribution* $\log N(\mu, \sigma^2)$, available for parameter a_i only. It is specified by `"lnorm"` as first argument while the latter two arguments contain the values of μ and σ respectively.
- the *uniform distribution* $U([a, b])$, available for all parameters. It is specified by `"unif"` as first argument while the latter two arguments contain the values of a and b respectively. Note that taking a and b equal to a common value, say t, makes all parameters to be equal to t.
- the *Beta distribution* $Beta(\alpha, \beta)$, available for parameters c_i and d_i. It is specified by `"beta"` as first argument while the latter two arguments contain the values of α and β respectively.

Inattention parameters d_i are fixed to 1 if `model` is not `"4PL"`; pseudo-guessing parameters c_i are fixed to zero if `model` is either `"1PL"` or `"2PL"`; and discrimination parameters a_i are fixed to 1 if `model="1PL"`. The random generation of item parameters can be controlled by the `seed` argument.

If required, the distribution of the items across subgroups with specified names can be performed. To do so, the `cbControl` argument must be supplied with a list of two arguments: (a) the first argument is called `$names` and contains the different names of the subgroups of items; (b) the second argument is called `$props` and contains a vector of numeric values, of the same length of `names` element, with only positive numbers but not necessarily summing to one. For instance, if `props` is set as `c(1, 2, 2)` and `items` to `100`, then the three subgroups will hold respectively 20, 40 and 40 items.

Several constraints apply to the arguments of the `cbControl` list. First, both arguments of `cbControl` must be of the same length. Second, as already explained, `cbControl$props` must either sum to 1 (in case of proportions) or to `items` (in case of integer values). Finally, if proportions are provided to `cbControl$props`, one can ensure that when multiplied by `items` they return integer values (so that they can sum up to `items`).

The random generation of item parameters and the random allocation of items to subgroups of items are both under control by the `seed` argument.

The output is a data frame with at least four arguments, with names `a`, `b`, `c` and `d` for respectively the discrimination a_i, the difficulty b_i, the lower asymptote c_i and the upper asymptote d_i parameters. A fifth argument contains optionally the subgroup names that have been randomly assigned to the generated items, in the proportions specified by the `$props` argument of the `cbControl` list.

Value

A data frame with four or five arguments:

`a`	the generated item discrimination parameters.
`b`	the generated item difficulty parameters.
`c`	the generated item lower asymptote parameters.
`d`	the generated item upper asymptote parameters.
`Group`	(optional) the distribution of subgroup names across items. Ignored if `cbControl` is `NULL`.

Note

The current version of genItemBank is only designed for dichotomous IRT models. Future extensions will hopefully provide the same tool for polytomous IRT models.

Author(s)

David Magis
Department of Psychology, University of Liege, Belgium
<david.magis@uliege.be>

References

Barton, M.A., and Lord, F.M. (1981). *An upper asymptote for the three-parameter logistic item-response model*. Research Bulletin 81-20. Princeton, NJ: Educational Testing Service.

Magis, D. and Barrada, J. R. (2017). Computerized Adaptive Testing with R: Recent Updates of the Package *catR*. *Journal of Statistical Software*, *Code Snippets*, *76(1)*, 1-18. doi: 10.18637/jss.v076.c01

Magis, D., and Raiche, G. (2012). Random Generation of Response Patterns under Computerized Adaptive Testing with the R Package *catR*. *Journal of Statistical Software*, *48 (8)*, 1-31. doi: 10.18637/jss.v048.i08

See Also

nextItem, randomCAT

Examples

```
# Item bank generation with 500 items
genDichoMatrix(items = 500)

# Item bank generation with 100 items, 2PL model and log-normal distribution with
# parameters (0, 0.1225) for discriminations
genDichoMatrix(items = 100, model = "2PL", aPrior = c("lnorm", 0, 0.1225))

# A completely identical method as for previous example
genDichoMatrix(items = 100, aPrior = c("lnorm", 0, 0.1225),
 cPrior = c("unif", 0, 0), dPrior = c("unif", 1, 1))

# Item bank generation with prespecified content balancing control options
cbList <- list(names = c("Group1", "Group2", "Group3", "Group4"),
       props = c(0.2,0.4,0.3,0.1))
genDichoMatrix(items = 100, cbControl = cbList)

# With proportions that do not sum to one
cbList <- list(names = c("Group1", "Group2", "Group3", "Group4"), props=c(2, 4, 3, 1))
genDichoMatrix(items = 100, cbControl = cbList)
```

genPattern *Random generation of item response patterns under dichotomous and polytomous IRT models*

Description

This command generates item responses patterns for a given matrix of item parameters of any specified dichotomous or polytomous IRT model and a given (set of) ability value(s).

Usage

```
genPattern(th, it, model = NULL, D = 1, seed = NULL)
```

Arguments

th	numeric: a vector of true underlynig ability values (can be a singlre value too).
it	numeric: a suitable matrix of item parameters. See **Details**.
model	either NULL (default) for dichotomous models, or any suitable acronym for polytomous models. Possible values are "GRM", "MGRM", "PCM", "GPCM", "RSM" and "NRM". See **Details**.
D	numeric: the metric constant. Default is D=1 (for logistic metric); D=1.702 yields approximately the normal metric (Haley, 1952). Ignored if model is not NULL.
seed	either the random seed value or NULL (default). See **Details**.

Details

This function permits to randomly generate item responses for a set of given ability levels, specified by the argument thr, and for a given matrix of item parameters, specified by the argument it. Both dichotomous and polytomous IRT models can be considered and item responses are generated accordingly.

For dichotomous models, item responses are generated from Bernoulli draws, using the rbinom function. For polytomous models they are generated from darws from a multinomial distribution, using the rmultinom function. In both cases, success probabilities are obtained from the Pi function.

Note that for polytomous models, item responses are coded as 0 (for the first response category), 1 (for the second category), ..., until g_j (for the last category), in agreement with the notations used in the help files of, e.g., the genPolyMatrix function.

Dichotomous IRT models are considered whenever model is set to NULL (default value). In this case, it must be a matrix with one row per item and four columns, with the values of the discrimination, the difficulty, the pseudo-guessing and the inattention parameters (in this order). These are the parameters of the four-parameter logistic (4PL) model (Barton and Lord, 1981).

Polytomous IRT models are specified by their respective acronym: "GRM" for Graded Response Model, "MGRM" for Modified Graded Response Model, "PCM" for Partical Credit Model, "GPCM" for

Generalized Partial Credit Model, "RSM" for Rating Scale Model and "NRM" for Nominal Response Model. The it still holds one row per item, end the number of columns and their content depends on the model. See genPolyMatrix for further information and illustrative examples of suitable polytomous item banks.

The random pattern generation can be fixed by setting seed to some numeric value. By default, seed is NULL and the random seed is not fixed.

Value

If th holds a single value, output is a vector with the item responses in the order of appearance of the items in the it matrix. If th is a vector of numeric values, output is a response matrix with one row per th value and one column per item.

Author(s)

David Magis
Department of Psychology, University of Liege, Belgium
<david.magis@uliege.be>

References

Barton, M.A., and Lord, F.M. (1981). *An upper asymptote for the three-parameter logistic item-response model*. Research Bulletin 81-20. Princeton, NJ: Educational Testing Service.

Haley, D.C. (1952). *Estimation of the dosage mortality relationship when the dose is subject to error*. Technical report no 15. Palo Alto, CA: Applied Mathematics and Statistics Laboratory, Stanford University.

Magis, D. and Barrada, J. R. (2017). Computerized Adaptive Testing with R: Recent Updates of the Package *catR*. *Journal of Statistical Software*, *Code Snippets*, *76(1)*, 1-18. doi: 10.18637/jss.v076.c01

Magis, D., and Raiche, G. (2012). Random Generation of Response Patterns under Computerized Adaptive Testing with the R Package *catR*. *Journal of Statistical Software*, *48 (8)*, 1-31. doi: 10.18637/jss.v048.i08

See Also

rbinom and rmultinom for random draws; genPolyMatrix, Pi

Examples

```
## Dichotomous models ##

 # Loading the 'tcals' parameters
 data(tcals)

 # Selecting item parameters only
 tcals <- as.matrix(tcals[,1:4])

 # Generation of a response pattern for ability level 0
```

```
genPattern(th = 0, tcals)

# Generation of 10 response patterns for ability levels randomly drawn from N(0,1)
genPattern(th = rnorm(10), tcals)

# Generation of a single response for the first item only
genPattern(th = 0, tcals[1,])

## Polytomous models ##

# Generation of an item bank under GRM with 100 items and at most 4 categories
m.GRM <- genPolyMatrix(100, 4, "GRM")
m.GRM <- as.matrix(m.GRM)

# Generation of a response pattern for ability level 0
genPattern(0, m.GRM, model = "GRM")

# Generation of 10 response patterns for ability levels randomly drawn from N(0,1)
genPattern(rnorm(10), m.GRM, model = "GRM")

# Generation of a single response for the first item only
genPattern(0, m.GRM[1,], model = "GRM")

# Loading the cat_pav data
data(cat_pav)
cat_pav <- as.matrix(cat_pav)

# Generation of a response pattern for ability level 0
genPattern(0, cat_pav, model = "GPCM")

# Generation of a single response for the first item only
genPattern(0, cat_pav[1,], model = "GPCM")
```

`genPolyMatrix` *Item bank generation (polytomous models)*

Description

This command generates an item bank from prespecified parent distributions for use with polytomous IRT models. Subgroups of items can also be specified for content balancing purposes.

Usage

```
genPolyMatrix(items = 100, nrCat = 3, model = "GRM", seed = 1, same.nrCat = FALSE,
  cbControl = NULL)
```

Arguments

items	integer: the number of items to generate (default is 100).
nrCat	integer: the (maximum) number of response categories to generate (default is 3).
model	character: the type of polytomous IRT model. Possible values are "GRM" (default), "MGRM", "PCM", "GPCM" and "NRM". See **Details**.
seed	numeric: the random seed for item parameter generation (default is 1).
same.nrCat	logical: should all items have the same number of response categories? (default is FALSE. Ignored if model is either "MGRM" or "RSM". See **Details**.
cbControl	either a list of accurate format to control for content balancing, or NULL. See **Details**.

Details

The genPolyMatrix permits to quickly generate a polytomous item bank in suitable format for further use in e.g. computing item response probabilities with the Pi.

The six polytomous IRT models that are supported are:

1. the *Graded Response Model* (GRM; Samejima, 1969);
2. the *Modified Graded Response Model* (MGRM; Muraki, 1990);
3. the *Partial Credit Model* (PCM; Masters, 1982);
4. the *Generalized Partial Credit Model* (GPCM; Muraki, 1992);
5. the *Rating Scale Model* (RSM; Andrich, 1978);
6. the *Nominal Response Model* (NRM; Bock, 1972).

Each model is specified through the model argument, with its accronym surrounded by double quotes (i.e. "GRM" for GRM, "PCM" for PCM, etc.). The default value is "GRM".

For any item j, set $(0, ..., g_j)$ as the $g_j + 1$ possible response categories. The maximum number of response categories can differ across items under the GRM, PCM, GPCM and NRM, but they are obviously equal across items under the MGRM and RSM. In the latter, set g as the (same) number of response categories for all items. It is possible however to require all items to have the same number of response categories, by fixing the same.nrCat argument to TRUE.

In case of GRM, PCM, GPCM or NRM with same.nrCat being FALSE, the number of response categories $g_j + 1$ per item is drawn from a Poisson distribution with parameter nrCat, and this number is restricted to the interval [2; nrCat]. This ensure at least two response categories and at most nrCat categories. In all other cases, each $g_j + 1$ is trivially fixed to $g + 1 =$ nrCat.

Denote further $P_{jk}(\theta)$ as the probability of answering response category $k \in \{0, ..., g_j\}$ of item j. For GRM and MGRM, response probabilities $P_{jk}(\theta)$ are defined through cumulative probabilities, while for PCM, GPCM, RSM and NRM they are directly computed.

For GRM and MGRM, set $P^*_{jk}(\theta)$ as the (cumulative) probability of asnwering response category k or "above", that is $P^*_{jk}(\theta) = Pr(X_j \geq k|\theta)$ where X_j is the item response. It follows obviously that for any θ, $P^*_{j0}(\theta) = 1$ and $P^*_{jk}(\theta) = 0$ when $k > g_j$. Furthermore, response category probabilities

are found back by the relationship $P_{jk}(\theta) = P^*_{jk}(\theta) - P^*_{j,k+1}(\theta)$. Then, the GRM is defined by (Samejima, 1969)

$$P^*_{jk}(\theta) = \frac{\exp\left[\alpha_j\left(\theta - \beta_{jk}\right)\right]}{1 + \exp\left[\alpha_j\left(\theta - \beta_{jk}\right)\right]}$$

and the MGRM by (Muraki, 1990)

$$P^*_{jk}(\theta) = \frac{\exp\left[\alpha_j\left(\theta - b_j + c_k\right)\right]}{1 + \exp\left[\alpha_j\left(\theta - b_j + c_k\right)\right]}.$$

The PCM, GPCM, RSM and NRM are defined as "divide-by-total" models (Embretson and Reise, 2000). The PCM has following response category probability (Masters, 1982):

$$P_{jk}(\theta) = \frac{\exp\sum_{t=0}^{k}(\theta - \delta_{jt})}{\sum_{r=0}^{g_j}\exp\sum_{t=0}^{r}(\theta - \delta_{jt})} \quad \text{with} \quad \sum_{t=0}^{0}(\theta - \delta_{jt}) = 0.$$

The GPCM has following response category probability (Muraki, 1992):

$$P_{jk}(\theta) = \frac{\exp\sum_{t=0}^{k}\alpha_j\left(\theta - \delta_{jt}\right)}{\sum_{r=0}^{g_j}\exp\sum_{t=0}^{r}\alpha_j\left(\theta - \delta_{jt}\right)} \quad \text{with} \quad \sum_{t=0}^{0}\alpha_j\left(\theta - \delta_{jt}\right) = 0.$$

The RSM has following response category probability (Andrich, 1978):

$$P_{jk}(\theta) = \frac{\exp\sum_{t=0}^{k}[\theta - (\lambda_j + \delta_t)]}{\sum_{r=0}^{g_j}\exp\sum_{t=0}^{r}[\theta - (\lambda_j + \delta_t)]} \quad \text{with} \quad \sum_{t=0}^{0}[\theta - (\lambda_j + \delta_t)] = 0.$$

Finally, the NRM has following response category probability (Bock, 1972):

$$P_{jk}(\theta) = \frac{\exp(\alpha_{jk}\,\theta + c_{jk})}{\sum_{r=0}^{g_j}\exp(\alpha_{jr}\,\theta + c_{jr})} \quad \text{with} \quad \alpha_{j0}\,\theta + c_{j0} = 0.$$

The following parent distributions are considered to generate the different item parameters. The α_j parameters of GRM, MGRM and GPCM, as well as the α_{jk} parameters of the NRM, are drawn from a log-normal distribution with mean 0 and standard deviation 0.1225. All other parameters are drawn from a standard normal distribution. Moreover, the β_{jk} parameters of the GRM and the c_k parameters of the MGRM are sorted respectively in increasing and decreasing order of k, to ensure decreasing trend in the cumulative $P^*_{jk}(\theta)$ probabilities.

The output is a matrix with one row per item and as many columns as required to hold all item parameters. In case of missing response categories, the corresponding parameters are replaced by `NA` values. Column names refer to the corresponding model parameters. See **Details** for further explanations and **Examples** for illustrative examples.

Finally, the output matrix can contain an additional vector with the names of the subgroups to be used for content balancing purposes. To do so, the argument `cbControl` (with default value is `NULL`) must contain a list of two elements: (a) the `names` element with the names of the subgroups, and (b) the `props` elements with proportions of items per subgroup (of the same length of `names` element, with only positive numbers but not necessarily summing to one). The `cbControl` argument is similar to the one in `nextItem` and `randomCAT` functions to control for content balancing. The output matrix contains then an additional column, with the names of the subgroups randomly allocated to each item by using random multinomial draws with the probabilities given by `cbControl$props`.

Value

A matrix with `items` rows and as many columns as required for the considered IRT model:

- $\max_j\ g_j + 1$ columns, holding parameters $(\alpha_j, \beta_{j1}, ..., \beta_{j,g_j})$ if `model` is `"GRM"`;
- $g + 2$ columns, holding parameters $(\alpha_j, b_j, c_1, ..., c_g)$ if `model` is `"MGRM"`;
- $\max_j\ g_j$ columns, holding parameters $(\delta_{j1}, ..., \delta_{j,g_j})$ if `model` is `"PCM"`;
- $\max_j\ g_j + 1$ columns, holding parameters $(\alpha_j, \delta_{j1}, ..., \delta_{j,g_j})$ if `model` is `"GPCM"`;
- $g + 1$ columns, holding parameters $(\lambda_j, \delta_1, ..., \delta_g)$ if `model` is `"RSM"`;
- $2\ \max_j\ g_j$ columns, holding parameters $(\alpha_{j1}, c_{j1}, \alpha_{j2}, c_{j2}, ..., \alpha_{j,g_j}, c_{j,g_j})$ if `model` is `"NRM"`.

If `cbControl` is not `NULL`, the output matrix contains an additional colum for item membership is included.

Author(s)

David Magis
Department of Psychology, University of Liege, Belgium
<david.magis@uliege.be>

References

Andrich, D. (1978). A rating formulation for ordered response categories. *Psychometrika, 43*, 561-573. doi: 10.1007/BF02293814

Bock, R. D. (1972). Estimating item parameters and latent ability when responses are scored in two or more nominal categories. *Psychometrika, 37*, 29-51. doi: 10.1007/BF02291411

Embretson, S. E., and Reise, S. P. (2000). *Item response theory for psychologists*. Mahwah, NJ: Lawrence Erlbaum Associates.

Magis, D. and Barrada, J. R. (2017). Computerized Adaptive Testing with R: Recent Updates of the Package *catR*. *Journal of Statistical Software*, *Code Snippets*, *76(1)*, 1-18. doi: 10.18637/jss.v076.c01

Magis, D., and Raiche, G. (2012). Random Generation of Response Patterns under Computerized Adaptive Testing with the R Package *catR*. *Journal of Statistical Software*, *48 (8)*, 1-31. doi: 10.18637/jss.v048.i08

Masters, G. N. (1982). A Rasch model for partial credit scoring. *Psychometrika, 47*, 149-174. doi: 10.1007/BF02296272

Muraki, E. (1990). Fitting a polytomous item response model to Likert-type data. *Applied Psychological Measurement, 14*, 59-71. doi: 10.1177/014662169001400106

Muraki, E. (1992). A generalized partial credit model: Application of an EM algorithm. *Applied Psychological Measurement, 16*, 19-176. doi: 10.1177/014662169201600206

Samejima, F. (1969). *Estimation of latent ability using a response pattern of graded scores*. Psychometrika Monograph (vol. 17).

See Also

`Pi`

Examples

```
# All generated item banks have 10 items and at most four response categories

 # GRM
 genPolyMatrix(10, 4, model = "GRM")

 # GRM with same number of response categories
 genPolyMatrix(10, 4, model = "GRM", same.nrCat = TRUE)

 # MGRM
 genPolyMatrix(10, 4, model = "MGRM")

 # MGRM with same number of response categories
 genPolyMatrix(10, 4, model = "MGRM", same.nrCat = TRUE) # same result

 # PCM
 genPolyMatrix(10, 4, model = "PCM")

 # PCM with same number of response categories
 genPolyMatrix(10, 4, model = "PCM", same.nrCat = TRUE)

 # GPCM
 genPolyMatrix(10, 4, model = "GPCM")

 # GPCM with same number of response categories
 genPolyMatrix(10, 4, model = "GPCM", same.nrCat = TRUE)

 # RSM
 genPolyMatrix(10, 4, model = "RSM")

 # RSM with same number of response categories
 genPolyMatrix(10, 4, model = "RSM", same.nrCat = TRUE) # same result

 # NRM
 genPolyMatrix(10, 4, model = "NRM")

 # NRM with same number of response categories
 genPolyMatrix(10, 4, model = "NRM", same.nrCat = TRUE)

## Content balancing

 # Creation of the 'cbList' list with arbitrary proportions
 cbList <- list(names = c("Audio1", "Audio2", "Written1", "Written2", "Written3"),
        props = c(0.1, 0.2, 0.2, 0.2, 0.3))

 # NRM with 100 items
 genPolyMatrix(100, 4, model = "NRM", cbControl = cbList)
```

`Ii` *Item information functions, first and second derivatives (dichotomous and polytomous models)*

Description

This command returns the Fisher information functions for a given ability value and a given matrix of item parameters under either the 4PL model or any suitable polytomous model. Numerical values of the first and second derivatives of the item information functions are also returned.

Usage

```
Ii(th, it, model = NULL, D = 1)
```

Arguments

`th`	numeric: the ability value.
`it`	numeric: a suitable matrix of item parameters. See **Details**.
`model`	either `NULL` (default) for dichotomous models, or any suitable acronym for polytomous models. Possible values are `"GRM"`, `"MGRM"`, `"PCM"`, `"GPCM"`, `"RSM"` and `"NRM"`. See **Details**.
`D`	numeric: the metric constant. Default is `D=1` (for logistic metric); `D=1.702` yields approximately the normal metric (Haley, 1952). Ignored if `model` is not `NULL`.

Details

The first and second derivatives are computed algebraically, either from the four-parameter logistic (4PL) model (Barton and Lord, 1981) or from the corresponding polytomous model. These derivatives are necessary for both the estimation of ability and the computation of related standard errors.

Dichotomous IRT models are considered whenever `model` is set to `NULL` (default value). In this case, `it` must be a matrix with one row per item and four columns, with the values of the discrimination, the difficulty, the pseudo-guessing and the inattention parameters (in this order). These are the parameters of the four-parameter logistic (4PL) model (Barton and Lord, 1981).

Polytomous IRT models are specified by their respective acronym: `"GRM"` for Graded Response Model, `"MGRM"` for Modified Graded Response Model, `"PCM"` for Partical Credit Model, `"GPCM"` for Generalized Partial Credit Model, `"RSM"` for Rating Scale Model and `"NRM"` for Nominal Response Model. The `it` still holds one row per item, end the number of columns and their content depends on the model. See `genPolyMatrix` for further information and illustrative examples of suitable polytomous item banks.

Value

A list with three arguments:

`Ii`	the vector with item informations (one value per item).
`dIi`	the vector with first derivatives of the item information functions (one value per item).
`d2Ii`	the vector with second derivatives of the item information functions (one value per item).

Author(s)

David Magis
Department of Psychology, University of Liege, Belgium
<david.magis@uliege.be>

References

Barton, M.A., and Lord, F.M. (1981). *An upper asymptote for the three-parameter logistic item-response model*. Research Bulletin 81-20. Princeton, NJ: Educational Testing Service.

Haley, D.C. (1952). *Estimation of the dosage mortality relationship when the dose is subject to error*. Technical report no 15. Palo Alto, CA: Applied Mathematics and Statistics Laboratory, Stanford University.

Magis, D. and Barrada, J. R. (2017). Computerized Adaptive Testing with R: Recent Updates of the Package *catR*. *Journal of Statistical Software*, *Code Snippets*, *76(1)*, 1-18. doi: 10.18637/jss.v076.c01

Magis, D., and Raiche, G. (2012). Random Generation of Response Patterns under Computerized Adaptive Testing with the R Package *catR*. *Journal of Statistical Software*, *48 (8)*, 1-31. doi: 10.18637/jss.v048.i08

See Also

`Pi`, `thetaEst`, `genPolyMatrix`

Examples

```
## Dichotomous models ##

 # Loading the 'tcals' parameters
 data(tcals)

 # Selecting item parameters only
 tcals <- as.matrix(tcals[,1:4])

 # Item information functions and derivatives
 # (various th and D values)
 Ii(th = 0, tcals)
 Ii(th = 0, tcals, D = 1.702)
 Ii(th = 1, tcals)
```

```
## Polytomous models ##

 # Generation of an item bank under GRM with 100 items and at most 4 categories
 m.GRM <- genPolyMatrix(100, 4, "GRM")
 m.GRM <- as.matrix(m.GRM)

 # Computation of item information and derivatives for ability level 0
 Ii(0, m.GRM, model = "GRM")

 # Loading the cat_pav data
 data(cat_pav)
 cat_pav <- as.matrix(cat_pav)

 # Computation of item information and derivatives for ability level 1
 Ii(1, cat_pav, model = "GPCM")
```

`integrate.catR` *Numerical integration by linear interpolation (for catR internal use)*

Description

This command computes the integral of function *f(x)* by providing values of *x* and *f(x)*, similarly to the `integrate.xy` function of the R package `sfsmisc`.

Usage

```
integrate.catR(x, y)
```

Arguments

`x`	numeric: a vector of *x* values for numerical integration.
`y`	numeric: a vector of numerical values corresponding to *f(x)* values.

Details

This function was written to compute "cheap" numerical integration by providing sequences of *x* values and corresponding computed values *f(x)*. It works similarly as the `integrate.xy` function when `use.spline=FALSE` is required. It was developed internally to eventually remove dependency of `catR` package to package `sfsmisc`.

Value

The approximated integral.

Author(s)

David Magis
Department of Psychology, University of Liege, Belgium
<david.magis@uliege.be>

References

Maechler, M. et al. (2012). *sfsmisc: Utilities from Seminar fuer Statistik ETH Zurich*. R package version 1.0-23. http://CRAN.R-project.org/package=sfsmisc

See Also

KL and the `integrate.xy` function in package `sfsmisc`

Examples

```
 # Loading the 'tcals' parameters
 x <- seq(from = -4, to = 4, length = 33)
 y <- exp(x)
 integrate.catR(x, y) # 54.86381

## Not run:
 # Comparison with integrate.xy
 require(sfsmisc)
 integrate.xy(x, y, use.spline = FALSE) # 54.86381
 integrate.xy(x, y) # 54.58058

## End(Not run)
```

Ji

Function $J(\theta)$ for weighted likelihood estimation (dichotomous and polytomous IRT models)

Description

This command returns the $J(\theta)$ function that is necessary to obtain the weighted likelihood estimation of ability with dichotomous and polytomous IRT models, as well as its asymptotic standard error.

Usage

```
Ji(th, it, model = NULL, D = 1)
```

Arguments

th	numeric: the ability value.
it	numeric: a suitable matrix of item parameters. See **Details**.
model	either NULL (default) for dichotomous models, or any suitable acronym for polytomous models. Possible values are "GRM", "MGRM", "PCM", "GPCM", "RSM" and "NRM". See **Details**.
D	numeric: the metric constant. Default is D=1 (for logistic metric); D=1.702 yields approximately the normal metric (Haley, 1952). Ignored if model is not NULL.

Details

The $J(\theta)$ fucntion is defined by (Samejima, 1998):

$$J(\theta) = \sum_{j=1}^{n} \sum_{k=0}^{g_j} \frac{P'_{jk}(\theta)\, P''_{jk}(\theta)}{P_{jk}(\theta)}$$

where n is the number of items; g_j the number of response categories for item j ($j = 1, ..., n$); $P_{jk}(\theta)$ the response category probabilities and $P'_{jk}(\theta)$ and $P''_{jk}(\theta)$ the first and second derivatives with respect to θ. In case of dichotomous IRT models, this reduces to (Warm, 1989):

$$J(\theta) = \sum_{j=1}^{n} \frac{P'_{j}(\theta)\, P''_{j}(\theta)}{P_{j}(\theta)\, Q_{j}(\theta)}$$

with $Q_j(\theta) = 1 - P_j(\theta)$.

This function is useful to compute the weighted likelihood estimates of ability with dichotomous and polytomous IRT models as well as their related asymptotic standard errors.

Dichotomous IRT models are considered whenever model is set to NULL (default value). In this case, it must be a matrix with one row per item and four columns, with the values of the discrimination, the difficulty, the pseudo-guessing and the inattention parameters (in this order). These are the parameters of the four-parameter logistic (4PL) model (Barton and Lord, 1981).

Polytomous IRT models are specified by their respective acronym: "GRM" for Graded Response Model, "MGRM" for Modified Graded Response Model, "PCM" for Partical Credit Model, "GPCM" for Generalized Partial Credit Model, "RSM" for Rating Scale Model and "NRM" for Nominal Response Model. The it still holds one row per item, end the number of columns and their content depends on the model. See genPolyMatrix for further information and illustrative examples of suitable polytomous item banks.

Value

A list with two arguments:

Ji	the vector with $J(\theta)$ values (one value per item).
dJi	the vector with first derivatives of the $J(\theta)$ values (one value per item).

Author(s)

David Magis
Department of Psychology, University of Liege, Belgium
<david.magis@uliege.be>

References

Barton, M.A., and Lord, F.M. (1981). *An upper asymptote for the three-parameter logistic item-response model*. Research Bulletin 81-20. Princeton, NJ: Educational Testing Service.

Haley, D.C. (1952). *Estimation of the dosage mortality relationship when the dose is subject to error*. Technical report no 15. Palo Alto, CA: Applied Mathematics and Statistics Laboratory, Stanford University.

Magis, D. and Barrada, J. R. (2017). Computerized Adaptive Testing with R: Recent Updates of the Package *catR*. *Journal of Statistical Software*, *Code Snippets*, *76(1)*, 1-18. doi: 10.18637/jss.v076.c01

Magis, D., and Raiche, G. (2012). Random Generation of Response Patterns under Computerized Adaptive Testing with the R Package *catR*. *Journal of Statistical Software*, *48 (8)*, 1-31. doi: 10.18637/jss.v048.i08

Samejima, F. (1998, April). *Expansion of Warm's weighted likelihood estimator of ability for the three-parameter logistic model to generate discrete responses.* PPaper presented at the annual meeting of the National Council on Measurement in Education, San Diego, CA.

Warm, T.A. (1989). Weighted likelihood estimation of ability in item response models. *Psychometrika, 54*, 427-450. doi: 10.1007/BF02294627

See Also

thetaEst, semTheta, genPolyMatrix

Examples

```
## Dichotomous models ##

 # Loading the 'tcals' parameters
 data(tcals)

 # Selecting item parameters only
 tcals <- as.matrix(tcals[,1:4])

 # Various J functions and derivatives
 # (various th and D values)
 Ji(th = 0, tcals)
 Ji(th = 0, tcals, D = 1.702)
 Ji(th = 1, tcals)

## Polytomous models ##

 # Generation of an item bank under GRM with 100 items and at most 4 categories
 m.GRM <- genPolyMatrix(100, 4, "GRM")
```

```
m.GRM <- as.matrix(m.GRM)

# Computation of J function and derivatives for ability level 0
Ji(0, m.GRM, model = "GRM")

# Loading the cat_pav data
data(cat_pav)
cat_pav <- as.matrix(cat_pav)

# Computation of J function and derivatives for ability level 1
Ji(1, cat_pav, model = "GPCM")
```

KL *Kullback-Leibler (KL) and posterior Kullback-Leibler (KLP) values for item selection*

Description

This command returns the value of the Kullback-Leibler (KL) or posterior Kullback-Leibler (KLP) for a given item, an item bank and a set of previously administered items.

Usage

```
KL(itemBank, item, x, it.given, model = NULL, theta = NULL, lower = -4,
  upper = 4, nqp = 33, type = "KL", priorDist = "norm", priorPar = c(0, 1),
  D = 1, X = NULL, lik = NULL)
```

Arguments

itemBank	numeric: a suitable matrix of item parameters. See **Details**.
item	numeric: the item (referred to as its rank in the item bank) for which the KL or KLP must be computed.
x	numeric: a vector of item responses, coded as 0 or 1 only (for dichotomous items) or from 0 to the number of response categories minus one (for polytomous items).
it.given	numeric: a matrix with one row per item and four columns, with the values of the discrimination, the difficulty, the pseudo-guessing and the inattention parameters (in this order). The number of rows of it must be equal to the length of x.
model	either NULL (default) for dichotomous models, or any suitable acronym for polytomous models. Possible values are "GRM", "MGRM", "PCM", "GPCM", "RSM" and "NRM". See **Details**.
theta	either a numeric value for provisional ability estimate or NULL. See **Details**.
lower	numeric: the lower bound for numercal integration (default is -4).

upper	numeric: the upper bound for numercal integration (default is 4).
nqp	numeric: the number of quadrature points (default is 33).
type	character: the type of information to be computed. Possible values are "KL" (default) and "KLP". See **Details**.
priorDist	character: the prior ability distribution. Possible values are "norm" (default) for the normal distribution, and "unif" for the uniform distribution. Ignored if type is "KL".
priorPar	numeric: a vector of two components with the prior parameters. If priorDist is "norm", then priorPar contains the mean and the standard deviation of the normal distribution. If priorDist is "unif", then priorPar contains the bounds of the uniform distribution. The default values are 0 and 1 respectively. Ignored if type is "KL".
D	numeric: the metric constant. Default is D=1 (for logistic metric); D=1.702 yields approximately the normal metric (Haley, 1952).
X	either a vector of numeric values or NULL (default). See **Details**.
lik	either a vector of numeric values or NULL (default). See **Details**.

Details

Kullback-Leibler information can be used as a rule for selecting the next item in the CAT process (Barrada, Olea, Ponsoda and Abad, 2010; Chang and Ying, 1996), both with dichotomous and polytomous IRT models. This command serves as a subroutine for the nextItem function.

Dichotomous IRT models are considered whenever model is set to NULL (default value). In this case, itemBank must be a matrix with one row per item and four columns, with the values of the discrimination, the difficulty, the pseudo-guessing and the inattention parameters (in this order). These are the parameters of the four-parameter logistic (4PL) model (Barton and Lord, 1981).

Polytomous IRT models are specified by their respective acronym: "GRM" for Graded Response Model, "MGRM" for Modified Graded Response Model, "PCM" for Partical Credit Model, "GPCM" for Generalized Partial Credit Model, "RSM" for Rating Scale Model and "NRM" for Nominal Response Model. The itemBank still holds one row per item, end the number of columns and their content depends on the model. See genPolyMatrix for further information and illustrative examples of suitable polytomous item banks.

Under polytomous IRT models, let k be the number of administered items, and set $x_1, ..., x_k$ as the provisional response pattern (where each response x_l takes values in $\{0, 1, ..., g_l\}$). Set $\hat{\theta}_k$ as the provisional ability estimate (with the first k responses) and let j be the item of interest (not previously administered). Set also $L(\theta|x_1, ..., x_k)$ as the likelihood function of the first k items and evaluated at θ. Set finally $P_{jt}(\theta)$ as the probability of answering response category t to item j for a given ability level θ. Then, Kullack-Leibler (KL) information is defined as

$$KL_j(\theta||\hat{\theta}_k) = \sum_{t=0}^{g_j} P_{jt}(\hat{\theta}_k) \log\left(\frac{P_{jt}(\hat{\theta}_k)}{P_{jt}(\theta)}\right).$$

In case of dichotomous IRT models, all g_l values reduce to 1, so that item responses x_l equal either 0 or 1. Set simply $P_j(\theta)$ as the probability of answering item j correctly for a given ability level θ.

Then, KL information reduces to

$$KL_j(\theta||\hat{\theta}) = P_j(\hat{\theta}) \log\left(\frac{P_j(\hat{\theta}_k)}{P_j(\theta)}\right) + [1 - P_j(\hat{\theta}_k)] \log\left(\frac{1 - P_j(\hat{\theta}_k)}{1 - P_j(\theta)}\right).$$

The quantity that is returned by this `KL` function is either: the likelihood function weighted by Kullback-Leibler information (the KL value):

$$KL_j(\hat{\theta}_k) = \int KL_j(\theta||\hat{\theta}_k)\, L(\theta|x_1, ..., x_k)\, d\theta$$

or the posterior function weighted by Kullback-Leibler information (the KLP value):

$$KLP_j(\hat{\theta}) = \int KL_j(\theta||\hat{\theta}_k)\, \pi(\theta)\, L(\theta|x_1, ..., x_k)\, d\theta$$

where $\pi(\theta)$ is the prior distribution of the ability level.

These integrals are approximated by the `integrate.catR` function. The range of integration is set up by the arguments `lower`, `upper` and `nqp`, giving respectively the lower bound, the upper bound and the number of quadrature points. The default range goes from -4 to 4 with length 33 (that is, by steps of 0.25).

To speed up the computation, both the range of integration of values θ and the values of the likelihood function $L(\theta)$ can be directly provided to the function through the arguments `X` and `lik`. If `X` is set to `NULL` (default), the sequence of ability values for integration is determined by the arguments `lower`, `upper` and `nqp` as explained above. If `lik` is `NULL` (default), it is also internally computed from an implementation of the likelihood function.

The provisional response pattern and the related item parameters are provided by the arguments `x` and `it.given` respectively. The target item (for which the KL information is computed) is given by its rank number in the item bank, through the `item` argument.

An ability level estimate must be provided to compute KL and KLP information values. Either the value is specified through the `theta` argument, or it is left equal to `NULL` (default). In this case, ability estimate is computed internally by maximum likelihood, using the `thetaEst` function with arguments `it.given` and `x`.

Note that the provisional response pattern `x` can also be set to `NULL` (which is useful when the number `nrItems` of starting items is set to zero). In this case, `it.given` must be a matrix with zero rows, such as e.g., `itemBank[NULL,]`. In this very specific configuration, the likelihood function $L(\theta|x_1, ..., x_k)$ reduces to the constant value 1 on the whole θ range (that is, a uniform likelihood).

The argument `type` defines the type of KL information to be computed. The default value, `"KL"`, computes the usual Kullback-Leibler information, while the posterior Kullback-Leibler value is obtained with `type="KLP"`. For the latter, the `priorDist` and `priorPar` arguments fix the prior ability distribution. The normal distribution is set up by `priorDist="norm"` and then, `priorPar` contains the mean and the standard deviation of the normal distribution. If `priorDist` is `"unif"`, then the uniform distribution is considered, and `priorPar` fixes the lower and upper bounds of that uniform distribution. By default, the standard normal prior distribution is assumed. These arguments are ignored whenever `method` is `"KL"`.

Value

The required KL or KLP value for the selected item.

Author(s)

David Magis
Department of Psychology, University of Liege, Belgium
<david.magis@uliege.be>

Juan Ramon Barrada
Department of Psychology and Sociology, Universidad Zaragoza, Spain
<barrada@unizar.es>

References

Barrada, J. R., Olea, J., Ponsoda, V., and Abad, F. J. (2010). A method for the comparison of item selection rules in computerized adaptive testing. *Applied Psychological Measurement*, *20*, 213-229. doi: 10.1177/0146621610370152

Barton, M.A., and Lord, F.M. (1981). *An upper asymptote for the three-parameter logistic item-response model*. Research Bulletin 81-20. Princeton, NJ: Educational Testing Service.

Chang, H.-H., and Ying, Z. (1996). A global information approach to computerized adaptive testing. *Applied Psychological Measurement, 34*, 438-452. doi: 10.1177/014662169602000303

Haley, D.C. (1952). *Estimation of the dosage mortality relationship when the dose is subject to error*. Technical report no 15. Palo Alto, CA: Applied Mathematics and Statistics Laboratory, Stanford University.

Magis, D. and Barrada, J. R. (2017). Computerized Adaptive Testing with R: Recent Updates of the Package *catR*. *Journal of Statistical Software*, *Code Snippets*, *76(1)*, 1-18. doi: 10.18637/jss.v076.c01

Magis, D., and Raiche, G. (2012). Random Generation of Response Patterns under Computerized Adaptive Testing with the R Package *catR*. *Journal of Statistical Software*, *48 (8)*, 1-31. doi: 10.18637/jss.v048.i08

See Also

integrate.catR, nextItem, genPolyMatrix

Examples

```
## Dichotomous models ##

 # Loading the 'tcals' parameters
 data(tcals)

 # Selecting item parameters only
 bank <- as.matrix(tcals[,1:4])

 # Selection of two arbitrary items (15 and 20) of the
 # 'tcals' data set
 it.given <- bank[c(15, 20),]

 # Creation of a response pattern
 x <- c(0, 1)
```

```
# KL for item 1, ML estimate of ability computed
KL(bank, 1, x, it.given)

# Current (ML) ability estimate
theta <- thetaEst(it.given, x, method = "ML")
KL(bank, 1, x, it.given, theta = theta)

# WL ability estimate instead
theta <- thetaEst(it.given, x, method = "WL")
KL(bank, 1, x, it.given, theta = theta)

# KLP for item 1
KL(bank, 1, x, it.given, theta = theta, type = "KLP")

# KLP for item 1, different integration range
KL(bank, 1, x, it.given, theta = theta, type = "KLP", lower = -2, upper = 2, nqp = 20)

# KL for item 1, uniform prior distribution on the range [-2,2]
KL(bank, 1, x, it.given, theta = theta, type = "KLP", priorDist = "unif",
   priorPar = c(-2, 2))

# Computation of likelihood function beforehand
L <- function(th, r, param)
 prod(Pi(th, param)$Pi^r * (1 - Pi(th,param)$Pi)^(1 - r))
xx <- seq(from = -4, to = 4, length = 33)
y <- sapply(xx, L, x, it.given)
KL(bank, 1, x, it.given, theta = theta, X = xx, lik = y)


## Polytomous models ##

# Generation of an item bank under GRM with 100 items and at most 4 categories
m.GRM <- genPolyMatrix(100, 4, "GRM")
m.GRM <- as.matrix(m.GRM)

# Selection of two arbitrary items (15 and 20)
it.given <- m.GRM[c(15, 20),]

# Generation of a response pattern (true ability level 0)
x <- genPattern(0, it.given, model = "GRM")

# KL for item 1, ML estimate of ability computed
KL(m.GRM, 1, x, it.given, model = "GRM")

# Current (ML) ability estimate
theta <- thetaEst(it.given, x, method = "ML", model = "GRM")
KL(m.GRM, 1, x, it.given, theta = theta, model = "GRM")

# WL ability estimate instead
theta <- thetaEst(it.given, x, method = "WL", model = "GRM")
KL(m.GRM, 1, x, it.given, theta = theta, model = "GRM")
```

```
# KLP for item 1
KL(m.GRM, 1, x, it.given, theta = theta, model = "GRM", type = "KLP")

# KLP for item 1, different integration range
KL(m.GRM, 1, x, it.given, theta = theta, model = "GRM", type = "KLP", lower = -2,
   upper = 2, nqp = 20)

# KL for item 1, uniform prior distribution on the range [-2,2]
KL(m.GRM, 1, x, it.given, theta = theta, model = "GRM", type = "KLP",
   priorDist = "unif", priorPar = c(-2, 2))


# Loading the cat_pav data
data(cat_pav)
cat_pav <- as.matrix(cat_pav)

# Selection of two arbitrary items (15 and 20)
it.given <- cat_pav[c(15, 20),]

# Generation of a response pattern (true ability level 0)
x <- genPattern(0, it.given, model = "GPCM")

 # KL for item 1, ML estimate of ability computed
KL(cat_pav, 1, x, it.given, model = "GPCM")

# Current (ML) ability estimate
theta <- thetaEst(it.given, x, method = "ML", model = "GPCM")
KL(cat_pav, 1, x, it.given, theta = theta, model = "GPCM")

# WL ability estimate instead
theta <- thetaEst(it.given, x, method = "WL", model = "GPCM")
KL(cat_pav, 1, x, it.given, theta = theta, model = "GPCM")

# KLP for item 1
KL(cat_pav, 1, x, it.given, theta = theta, model = "GPCM", type = "KLP")

# KLP for item 1, different integration range
KL(cat_pav, 1, x, it.given, theta = theta, model = "GPCM", type = "KLP", lower = -2,
   upper = 2, nqp = 20)

# KL for item 1, uniform prior distribution on the range [-2,2]
KL(cat_pav, 1, x, it.given, theta = theta, model = "GPCM", type = "KLP",
   priorDist = "unif", priorPar = c(-2, 2))
```

MEI *(Maximum) Expected Information (MEI)*

Description

This command returns the expected information (EI) for a given item (under dichotomous and polytomous IRT models), as used for Maximum Expected Information (MEI) criterion.

Usage

```
MEI(itemBank, item, x, theta, it.given, model = NULL, method = "BM",
  priorDist = "norm", priorPar = c(0, 1), D = 1, range = c(-4, 4),
  parInt = c(-4, 4, 33), infoType = "observed")
```

Arguments

itemBank	numeric: a suitable matrix of item parameters. See **Details**.
item	numeric: the item (referred to as its rank in the item bank) for which the expected information must be computed.
x	numeric: a vector of item responses, coded as 0 or 1 only (for dichotomous items) or from 0 to the number of response categories minus one (for polytomous items).
theta	numeric: the provisional ability estimate.
it.given	numeric: a suitable matrix of item parameters for previously administered items. The number of rows of `it.given` must be equal to the length of `x`.
model	either `NULL` (default) for dichotomous models, or any suitable acronym for polytomous models. Possible values are `"GRM"`, `"MGRM"`, `"PCM"`, `"GPCM"`, `"RSM"` and `"NRM"`. See **Details**.
method	character: the ability estimator. Possible values are `"BM"` (default), `"ML"` and `"WL"`. See **Details**.
priorDist	character: specifies the prior distribution. Possible values are `"norm"` (default), `"unif"` and `"Jeffreys"`. Ignored if `method` is neither `"BM"` nor `"EAP"`. See **Details**.
priorPar	numeric: vector of two components specifying the prior parameters (default is `c(0,1)`) of the prior ability distribution. Ignored if `method` is neither `"BM"` nor `"EAP"`, or if `priorDist="Jeffreys"`. See **Details**.
D	numeric: the metric constant. Default is `D=1` (for logistic metric); `D=1.702` yields approximately the normal metric (Haley, 1952).
range	numeric: vector of two components specifying the range wherein the ability estimate must be looked for (default is `c(-4,4)`). Ignored if `method=="EAP"`.
parInt	numeric: vector of three components, holding respectively the values of the arguments `lower`, `upper` and `nqp` of the `eapEst` command. Default vector is (-4, 4, 33). Ignored if `method` is not `"EAP"`.
infoType	character: the type of information function to be used. Possible values are `"observed"` (default) for observed information function, and `"Fisher"` for Fisher information function.

Details

The MEI (van der Linden, 1998; van der Linden and Pashley, 2000) can be used as a rule for selecting the next item in the CAT process (see also Choi and Swartz, 2009), both with dichotomous and polytomous IRT models. This command serves as a subroutine for the `nextItem` function.

Dichotomous IRT models are considered whenever `model` is set to `NULL` (default value). In this case, `itemBank` must be a matrix with one row per item and four columns, with the values of the discrimination, the difficulty, the pseudo-guessing and the inattention parameters (in this order). These are the parameters of the four-parameter logistic (4PL) model (Barton and Lord, 1981).

Polytomous IRT models are specified by their respective acronym: `"GRM"` for Graded Response Model, `"MGRM"` for Modified Graded Response Model, `"PCM"` for Partical Credit Model, `"GPCM"` for Generalized Partial Credit Model, `"RSM"` for Rating Scale Model and `"NRM"` for Nominal Response Model. The `itemBank` still holds one row per item, end the number of columns and their content depends on the model. See `genPolyMatrix` for further information and illustrative examples of suitable polytomous item banks.

Under polytomous IRT models, let *k* be the number of administered items, and set $x_1, ..., x_k$ as the provisional response pattern (where each response x_l takes values in $\{0, 1, ..., g_l\}$). Set $\hat{\theta}_k$ as the provisional ability estimate (with the first *k* responses) and let *j* be the item of interest (not previously administered). Set also $P_{jt}(\theta)$ as the probability of answering response category *t* to item *j* for a given ability level θ. Finally, set $\hat{\theta}^t_{k+1}$ as the ability estimates computed under the condition that the response to item *j* is *t* (with $t = 0, ..., g_j$). Then, the EI for item *j* equals

$$EI_j = \sum_{t=0}^{g_j} P_{jt}(\hat{\theta}_k)\, I_j(\hat{\theta}^t_{k+1})$$

where $I_j(\theta)$ is the information function for item *j*.

In case of dichotomous IRT models, all g_l values reduce to 1, so that item responses x_l equal either 0 or 1. Set simply $P_j(\theta)$ as the probability of answering item *j* correctly for a given ability level θ, and set $Q_j(\theta) = 1 - P_j(\theta)$. Finally, set $\hat{\theta}^0_{k+1}$ and $\hat{\theta}^1_{k+1}$ as the ability estimates computed under the condition that the response to item *j* is 0 or 1 respectively (that is, if the response pattern is updated by 0 or 1 for item *j*). Then, the EI for item *j* reduces to

$$EI_j = P_j(\hat{\theta}_k)\, I_j(\hat{\theta}^1_{k+1}) + Q_j(\hat{\theta}_k)\, I_j(\hat{\theta}^0_{k+1})$$

.

Two types of information functions are available. The first one is the observed information function, defined as

$$OI_j(\theta) = -\frac{\partial^2}{\partial\theta^2} \log L(\theta|x_j)$$

(van der Linden, 1998), where $L(\theta|x_j)$ is the likelihood related to item *j*. The second one is Fisher information function:

$$I_j(\theta) = -E\left[\frac{\partial^2}{\partial\,\theta^2} \log L(\theta|x_j))\right].$$

Under the 1PL and the 2PL models, these functions are identical (Veerkamp, 1996). See also `OIi`.

The observed and Fisher information functions are specified by the `infoType` argument, with respective values `"observed"` and `"Fisher"`. By default, the observed information function is considered (Choi and Swartz, 2009; van der Linden, 1998).

The estimator of provisional ability is defined by means of the arguments `method`, `priorDist`, `priorPar`, `D`, `range` and `parInt` of the `thetaEst` function. See the corresponding help file for further details.

The provisional response pattern and the related item parameters are provided by the arguments `x` and `it.given` respectively. The target item (for which the maximum information computed) is given by its number in the item bank, through the `item` argument.

Note that the provisional response pattern `x` can also be set to `NULL` (which is useful when the number `nrItems` of starting items is set to zero). In this case, `it.given` must be a matrix with zero rows, such as e.g., `itemBank[NULL,]`.

Value

The required expected information for the selected item.

Author(s)

David Magis
Department of Psychology, University of Liege, Belgium
<david.magis@uliege.be>

References

Barton, M.A., and Lord, F.M. (1981). *An upper asymptote for the three-parameter logistic item-response model*. Research Bulletin 81-20. Princeton, NJ: Educational Testing Service.

Choi, S. W., and Swartz, R. J. (2009). Comparison of CAT item selection criteria for polytomous items. *Applied Psychological Measurement, 32*, 419-440. doi: 10.1177/0146621608327801

Haley, D.C. (1952). *Estimation of the dosage mortality relationship when the dose is subject to error*. Technical report no 15. Palo Alto, CA: Applied Mathematics and Statistics Laboratory, Stanford University.

Magis, D. and Barrada, J. R. (2017). Computerized Adaptive Testing with R: Recent Updates of the Package *catR*. *Journal of Statistical Software*, *Code Snippets*, *76(1)*, 1-18. doi: 10.18637/jss.v076.c01

Magis, D., and Raiche, G. (2012). Random Generation of Response Patterns under Computerized Adaptive Testing with the R Package *catR*. *Journal of Statistical Software*, *48 (8)*, 1-31. doi: 10.18637/jss.v048.i08

van der Linden, W. J. (1998). Bayesian item selection criteria for adaptive testing. *Psychometrika, 63*, 201-216. doi: 10.1007/BF02294775

van der Linden, W. J., and Pashley, P. J. (2000). Item selection and ability estimation in adaptive testing. In W. J. van der Linden and C. A. W. Glas (Eds.), *Computerized adaptive testing. Theory and practice* (pp. 1-25). Boston, MA: Kluwer.

Veerkamp, W. J. J. (1996). *Statistical inference for adaptive testing*. Internal report. Enschede, The Netherlands: University of Twente.

See Also

`Ii`, `OIi`, `nextItem`, `thetaEst`, `genPolyMatrix`

Examples

```
## Dichotomous models ##

 # Loading the 'tcals' parameters
 data(tcals)

 # Selecting item parameters only
 bank <- as.matrix(tcals[,1:4])

 # Selection of two arbitrary items (15 and 20) of the
 # 'tcals' data set
 it.given <- bank[c(15, 20),]

 # Creation of a response pattern
 x <- c(0, 1)

 # MEI for item 1, provisional ability level 0
 MEI(bank, 1, x, 0, it.given)

 # With Fisher information instead
 MEI(bank, 1, x, 0, it.given, infoType = "Fisher")

 # With WL estimator instead
 MEI(bank, 1, x, 0, it.given, method = "WL")


## Polytomous models ##

 # Generation of an item bank under GRM with 100 items and at most 4 categories
 m.GRM <- genPolyMatrix(100, 4, "GRM")
 m.GRM <- as.matrix(m.GRM)

 # Selection of two arbitrary items (15 and 20)
it.given <- m.GRM[c(15, 20),]

 # Generation of a response pattern (true ability level 0)
 x <- genPattern(0, it.given, model = "GRM")

 # EPV for item 1, provisional ability level 0
 MEI(m.GRM, 1, x, 0, it.given, model = "GRM")

 # With WL method
 MEI(m.GRM, 1, x, 0, it.given, model = "GRM", method = "WL")

 # With Fisher information
 MEI(m.GRM, 1, x, 0, it.given, model = "GRM", infoType = "Fisher")

 # Loading the cat_pav data
 data(cat_pav)
 cat_pav <- as.matrix(cat_pav)
```

```
# Selection of two arbitrary items (15 and 20)
it.given <- cat_pav[c(15, 20),]

# Generation of a response pattern (true ability level 0)
x <- genPattern(0, it.given, model = "GPCM")

# EPV for item 1, provisional ability level 0
MEI(cat_pav, 1, x, 0, it.given, model = "GPCM")

# With WL method
MEI(cat_pav, 1, x, 0, it.given, model = "GPCM", method = "WL")

# With Fisher information
MEI(cat_pav, 1, x, 0, it.given, model = "GPCM", infoType = "Fisher")
```

`MWI` *Maximum likelihood weighted information (MLWI) and maximum posterior weighted information (MPWI)*

Description

This command returns the value of the likelihood (MLWI) or the posterior (MPWI) weighted information for a given item and an item bank (both under dichotomous and polytomous IRT models).

Usage

```
MWI(itemBank, item, x, it.given, model = NULL, lower = -4, upper = 4, nqp = 33,
  type = "MLWI", priorDist = "norm", priorPar = c(0, 1), D = 1)
```

Arguments

`itemBank`	numeric: a suitable matrix of item parameters. See **Details**.
`item`	numeric: the item (referred to as its rank in the item bank) for which the expected information must be computed.
`x`	numeric: a vector of item responses, coded as 0 or 1 only (for dichotomous items) or from 0 to the number of response categories minus one (for polytomous items).
`it.given`	numeric: a suitable matrix of item parameters for previously administered items. The number of rows of `it.given` must be equal to the length of `x`.
`model`	either `NULL` (default) for dichotomous models, or any suitable acronym for polytomous models. Possible values are `"GRM"`, `"MGRM"`, `"PCM"`, `"GPCM"`, `"RSM"` and `"NRM"`. See **Details**.
`lower`	numeric: the lower bound for numerical integration (default is -4).
`upper`	numeric: the upper bound for numerical integration (default is 4).

nqp	numeric: the number of quadrature points (default is 33).
type	character: the type of information to be computed. Possible values are "MLWI" (default) and "MPWI". See **Details**.
priorDist	character: the prior ability distribution. Possible values are "norm" (default) for the normal distribution, and "unif" for the uniform distribution. Ignored if type is not "MPWI".
priorPar	numeric: a vector of two components with the prior parameters. If priorDist is "norm", then priorPar contains the mean and the standard deviation of the normal distribution. If priorDist is "unif", then priorPar contains the bounds of the uniform distribution. The default values are 0 and 1 respectively. Ignored if type is not "MPWI".
D	numeric: the metric constant. Default is D=1 (for logistic metric); D=1.702 yields approximately the normal metric (Haley, 1952). Ignored if model is not NULL.

Details

Both the MLWI (Veerkamp and Berger, 1997) and the MPWI (van der Linden, 1998; van der Linden and Pashley, 2000) can be used as rules for selecting the next item in the CAT process (see also Choi and Swartz, 2009), both under dichotomous and polytomous IRT models. This command serves as a subroutine for the nextItem function.

Dichotomous IRT models are considered whenever model is set to NULL (default value). In this case, itemBank must be a matrix with one row per item and four columns, with the values of the discrimination, the difficulty, the pseudo-guessing and the inattention parameters (in this order). These are the parameters of the four-parameter logistic (4PL) model (Barton and Lord, 1981).

Polytomous IRT models are specified by their respective acronym: "GRM" for Graded Response Model, "MGRM" for Modified Graded Response Model, "PCM" for Partical Credit Model, "GPCM" for Generalized Partial Credit Model, "RSM" for Rating Scale Model and "NRM" for Nominal Response Model. The itemBank still holds one row per item, end the number of columns and their content depends on the model. See genPolyMatrix for further information and illustrative examples of suitable polytomous item banks.

Under polytomous IRT models, let *k* be the number of administered items, and set $x_1, ..., x_k$ as the provisional response pattern (where each response x_l takes values in $\{0, 1, ..., g_l\}$). Set also $I_j(\theta)$ as the information function of item *j* evaluated at θ, and set $L(\theta|x_1, ..., x_k)$ as the likelihood function evaluated at θ, given the provisional response pattern. Then, the LWI for item *j* is given by

$$LWI_j = \int I_j(\theta)L(\theta|x_1, ..., x_k)d\theta$$

and the PWI by

$$PWI_j = \int I_j(\theta)\pi(\theta)L(\theta|x_1, ..., x_k)d\theta$$

where $\pi(\theta)$ is the prior distribution of the ability level.

In case of dichotomous IRT models, all g_l values reduce to 1, so that item responses x_l equal either 0 or 1. But except from this difference, the previous definitions of LWI and PWI remain valid.

These integrals are approximated by the integrate.catR function. The range of integration is set up by the arguments lower, upper and nqp, giving respectively the lower bound, the upper bound

and the number of quadrature points. The default range goes from -4 to 4 with length 33 (that is, by steps of 0.25).

The argument `type` defines the type of information to be computed. The default value, `"MLWI"`, computes the MLWI value, while the MPWI value is obtained with `type="MPWI"`. For the latter, the `priorDist` and `priorPar` arguments fix the prior ability distribution. The normal distribution is set up by `priorDist="norm"` and then, `priorPar` contains the mean and the standard deviation of the normal distribution. If `priorDist` is `"unif"`, then the uniform distribution is considered, and `priorPar` fixes the lower and upper bounds of that uniform distribution. By default, the standard normal prior distribution is assumed. This argument is ignored whenever `method` is not `"MPWI"`.

The provisional response pattern and the related item parameters are provided by the arguments `x` and `it.given` respectively. The target item (for which the maximum information is computed) is given by its rank number in the item bank, through the `item` argument.

Note that the provisional response pattern `x` can also be set to `NULL` (which is useful when the number `nrItems` of starting items is set to zero). In this case, `it.given` must be a matrix with zero rows, such as e.g., `itemBank[NULL,]`. In this very specific configuration, the likelihood function $L(\theta|x_1, ..., x_k)$ reduces to the constant value 1 on the whole θ range (that is, a uniform likelihood).

Value

The required (likelihood or posterior) weighted information for the selected item.

Author(s)

David Magis
Department of Psychology, University of Liege, Belgium
<david.magis@uliege.be>

References

Barton, M.A., and Lord, F.M. (1981). *An upper asymptote for the three-parameter logistic item-response model*. Research Bulletin 81-20. Princeton, NJ: Educational Testing Service.

Choi, S. W., and Swartz, R. J. (2009). Comparison of CAT item selection criteria for polytomous items. *Applied Psychological Measurement, 32*, 419-440. doi: 10.1177/0146621608327801

Haley, D.C. (1952). *Estimation of the dosage mortality relationship when the dose is subject to error*. Technical report no 15. Palo Alto, CA: Applied Mathematics and Statistics Laboratory, Stanford University.

Magis, D. and Barrada, J. R. (2017). Computerized Adaptive Testing with R: Recent Updates of the Package *catR*. *Journal of Statistical Software*, *Code Snippets*, *76(1)*, 1-18. doi: 10.18637/jss.v076.c01

Magis, D., and Raiche, G. (2012). Random Generation of Response Patterns under Computerized Adaptive Testing with the R Package *catR*. *Journal of Statistical Software*, *48 (8)*, 1-31. doi: 10.18637/jss.v048.i08

van der Linden, W. J. (1998). Bayesian item selection criteria for adaptive testing. *Psychometrika, 63*, 201-216. doi: 10.1007/BF02294775

van der Linden, W. J., and Pashley, P. J. (2000). Item selection and ability estimation in adaptive testing. In W. J. van der Linden and C. A. W. Glas (Eds.), *Computerized adaptive testing. Theory and practice* (pp. 1-25). Boston, MA: Kluwer.

Veerkamp, W. J. J., and Berger, M. P. F. (1997). Some new item selection criteria for adaptive testing. *Journal of Educational and Behavioral Statistics, 22*, 203-226. doi: 10.3102/10769986022002203

See Also

Ii, nextItem, integrate.catR, genPolyMatrix

Examples

```
## Dichotomous models ##

 # Loading the 'tcals' parameters
 data(tcals)

 # Selecting item parameters only
 bank <- as.matrix(tcals[,1:4])

  # Selection of two arbitrary items (15 and 20) of the
 # 'tcals' data set
 it.given <- bank[c(15, 20),]

 # Creation of a response pattern
 x <- c(0, 1)

 # MLWI for item 1
 MWI(bank, 1, x, it.given)

 # MPWI for item 1
 MWI(bank, 1, x, it.given, type = "MPWI")

 # MLWI for item 1, different integration range
 MWI(bank, 1, x, it.given, lower = -2, upper = 2, nqp = 20)

 # MPWI for item 1, uniform prior distribution on the range [-2,2]
 MWI(bank, 1, x, it.given, type = "MPWI", priorDist = "unif", priorPar = c(-2, 2))


## Polytomous models ##

 # Generation of an item bank under GRM with 100 items and at most 4 categories
 m.GRM <- genPolyMatrix(100, 4, "GRM")
 m.GRM <- as.matrix(m.GRM)

 # Selection of two arbitrary items (15 and 20)
 it.given <- m.GRM[c(15, 20),]

 # Generation of a response pattern (true ability level 0)
 x <- genPattern(0, it.given, model = "GRM")

 # MLWI for item 1
 MWI(m.GRM, 1, x, it.given, model = "GRM")
```

```
# MPWI for item 1
MWI(m.GRM, 1, x, it.given, model = "GRM", type = "MPWI")

# MLWI for item 1, different integration range
MWI(m.GRM, 1, x, it.given, model = "GRM", lower = -2, upper = 2, nqp = 20)

# MPWI for item 1, uniform prior distribution on the range [-2,2]
MWI(m.GRM, 1, x, it.given, model = "GRM", type = "MPWI", priorDist = "unif",
priorPar = c(-2, 2))


# Loading the cat_pav data
data(cat_pav)
cat_pav <- as.matrix(cat_pav)

# Selection of two arbitrary items (15 and 20)
it.given <- cat_pav[c(15, 20),]

# Generation of a response pattern (true ability level 0)
x <- genPattern(0, it.given, model = "GPCM")

 # MLWI for item 1
MWI(cat_pav, 1, x, it.given, model = "GPCM")

# MPWI for item 1
MWI(cat_pav, 1, x, it.given, model = "GPCM", type = "MPWI")

# MLWI for item 1, different integration range
MWI(cat_pav, 1, x, it.given, model = "GPCM", lower = -2, upper = 2, nqp = 20)

# MPWI for item 1, uniform prior distribution on the range [-2,2]
MWI(cat_pav, 1, x, it.given, model = "GPCM", type = "MPWI", priorDist = "unif",
priorPar = c(-2, 2))
```

`nextItem` *Selection of the next item*

Description

This command selects the next item to be administered, given the list of previously administered items and the current ability estimate, with several possible criteria. Item exposure and content balancing can also be controlled.

Usage

```
nextItem(itemBank, model = NULL, theta = 0, out = NULL, x = NULL,
  criterion = "MFI", method = "BM", priorDist = "norm", priorPar = c(0, 1),
  D = 1, range = c(-4, 4), parInt = c(-4, 4, 33), infoType = "observed",
  randomesque = 1, random.seed = NULL, rule = "length", thr = 20, SETH = NULL,
```

```
AP = 1, nAvailable = NULL, maxItems = 50, cbControl = NULL, cbGroup = NULL)
```

Arguments

itemBank	numeric: a suitable matrix of item parameters. See **Details**.
model	either NULL (default) for dichotomous models, or any suitable acronym for polytomous models. Possible values are "GRM", "MGRM", "PCM", "GPCM", "RSM" and "NRM". See **Details**.
theta	numeric: the current value of the ability estimate (default is 0). Ignored if criterion is either "MLWI", "MPWI" or "random". See **Details**.
out	either a vector of integer values specifying the items previously administered, or NULL (default).
x	numeric: the provisional response pattern, with the same length as out (and NULL by default). Ignored if criterion is either "MFI", "bOtp", "thOpt", "proportional", "progressive" or "random". See **Details**.
criterion	character: the method for next item selection. Possible values are "MFI" (default), "bOpt", "thOpt", "MLWI", "MPWI", "MEI", "MEPV", "progressive", "proportional", "KL", "KLP", "GDI", "GDIP" and random. See **Details**.
method	character: the ability estimator. Possible values are "BM" (default), "ML" and "WL". Ignored if method is not "MEI". See **Details**.
priorDist	character: the prior ability distribution. Possible values are "norm" (default) for the normal distribution, and "unif" for the uniform distribution. Ignored if type is not "MPWI", "KLP" or "GDIP".
priorPar	numeric: a vector of two components with the prior parameters. If priorDist is "norm", then priorPar contains the mean and the standard deviation of the normal distribution. If priorDist is "unif", then priorPar contains the bounds of the uniform distribution. The default values are 0 and 1 respectively. Ignored if type is neither "MPWI" nor "KLP".
D	numeric: the metric constant. Default is D=1 (for logistic metric); D=1.702 yields approximately the normal metric (Haley, 1952).
range	numeric: vector of two components specifying the range wherein the ability estimate must be looked for (default is c(-4,4)).
parInt	numeric: a vector of three numeric values, specifying respectively the lower bound, the upper bound and the number of quadrature points for numerical integration (default is c(-4,4,33)). Ignored if method is either "MFI", "bOpt", "thOpt", "progressive", "proportional" or "random". See **Details**.
infoType	character: the type of information function to be used. Possible values are "observed" (default) and "Fisher". Ignored if criterion is not "MEI". See **Details**.
randomesque	integer: the number of items to be chosen from the next item selection rule, among those the next item to be administered will be randomly picked up. Default value is 1 and leads to usual selection of the optimal item for the specified criterion. See **Details**.

random.seed	either NULL (default) or a numeric value to fix the random seed of randomesque selection of the items. Ignored if randomesque is equal to one.
rule	character: the type of stopping rule for next item selection. Possible values are "length" (default) or "precision". Ignored if criterion is any other value than "progressive" or "proportional". See **Details**.
thr	numeric: the threshold related to the stopping rule. Ignored if criterion is neither "progressive" nor "proportional". See **Details**.
SETH	either a numeric value for the provisional standard errorNULL (default). Ignored if criterion is neither "progressive" nor "proportional", or if rule is not "precision". See **Details**.
AP	numeric: the value of the acceleration parameter (default value is 1). Ignored if criterion is neither "progressive" nor "proportional". See **Details**.
nAvailable	either a numeric vector of zero and one entries to denote respectively which items are not available and are availbale, or NULL (default). Used for content balancing purposes only. See **Details**.
maxItems	integer: the maximum number of items to be administered during the adaptive test (default value is 50). See **Details**.
cbControl	either a list of accurate format to control for content balancing, or NULL. See **Details**.
cbGroup	either a factor vector of accurate format to control for content balancing, or NULL. See **Details**.

Details

Currently twelve methods are available for selecting the next item to be administered in the adaptive test. All are avilable with dichotomous items and ten out of the twelve are also available for polytomous items. For a given current ability estimate, the next item is selected (among the available items) by using:

1. the maximum Fisher information (MFI) criterion,
2. the so-called bOpt procedure (Urry, 1970) (**not** for polytomous items),
3. the so-called thOpt procedure (see e.g., Barrada, Mazuela and Olea, 2006; Magis, 2013) (**not** for polytomous items),
4. the maximum likelihood weighted information (MLWI) (Veerkamp and Berger, 1997),
5. the maximum posterior weighted information (MPWI) (van der Linden, 1998),
6. the maximum expected information (MEI) criterion (van der Linden, 1998),
7. the minimum expected posterior variance (MEPV),
8. the Kullback-Leibler (KL) divergency criterion (Chang and Ying, 1996),
9. the posterior Kullback-Leibler (KLP) criterion (Chang and Ying, 1996),
10. the progressive method (Barrada, Olea, Ponsoda, and Abad, 2008, 2010; Revuelta and Ponsoda, 1998),
11. the proportional method (Barrada, Olea, Ponsoda, and Abad, 2008, 2010; Segall, 2004),
12. the global-discrimination index (GDI) (Kaplan, de la Torre, and Barrada, 2015),

13. the posterior global-discrimination index (GDIP) (Kaplan, de la Torre, and Barrada, 2015),
14. or by selecting the next item completely *randomly* among the available items.

The MFI criterion selects the next item as the one which maximizes the item information function (Baker, 1992). The most informative item is selected from the item informations computed from the bank of items specified with `itemBank`.

The so-called `bOpt` method (formerly referred to as *Urry's procedure*) consists in selecting as next the item whose difficulty level is closest to the current ability estimate. Under the 1PL model, both bOpt and MFI methods are equivalent. This method is not available with polytomous items.

The so-called `thOpt` method consists in selecting the item for which optimal θ value (that is, the value θ^* for which item information is maximal) is closest to the current ability estimate. Under the 1PL and 2PL models, both bOpt and thOpt methods are equivalent. This method is not available with polytomous items.

The MLWI and MPWI criteria select the next item as the one with maximal information, weighted either by the likelihood function or the posterior distribution. See the function `MWI` for further details.

The MEI criterion selects the item with maximum expected information, computed with the `MEI` function.

The MEPV criterion selects the item with minimum expected posterior variance, computed with the `EPV` function.

The KL and KLP criteria select the item with maximum Kullback-Leibler (KL) information (or the posterior KL information in case of KLP criterion). This information is computed by the `KL` function.

With the progressive method the item selected is the one that maximizes the sum of two elements, a random part and a part determined by the Fisher information. At the beginning of the test, the importance of the random element is maximum; as the test advances, the information increases its relevance in the item selection. The speed for the transition from purely random selection to purely information based selection is determined by the acceleration parameter, set by the argument `AP`, where higher values imply a greater importance of the random element during the test. This method is not available with `rule="classification"`.

In the proportional method the items are randomly selected with probabilities of selection determined by their Fisher information raised to a given power. This power is equal to 0 at the beginning of the test and increases as the test advances. This implies that the test starts with completely random selection and approaches the MFI at the end of the test. Here, the acceleration parameter, set by the argument `AP`, plays a similar role than with the progressive method. This method is not available with `rule="classification"`.

The GDI and GDIP criteria select the item with maximum global-discrimination index (GDI), or the posterior GDI in case of GDIP criterion. This index is computed by the `GDI` function.

The method for next item selection is specified by the `criterion` argument. Possible values are `"MFI"` for maximum Fisher information criterion, `"bOpt"` for bOpt's method, `"thOpt"` for the eponym method, `"MLWI"` for maximum likelihood weighted information criterion, `"MPWI"` for the maximum posterior weighted information criterion, `"MEI"` for the maximum expected information criterion, `"MEPV"` for minimum expected posterior variance, `"KL"` for Kullback-Leibler information method, `"KLP"` for posterior Kullback-Leibler information method, `"progressive"` for the progressive method, `"proportional"` for the proportional method, `"GDI"` for global-discrimination

index method, "KLP" for posterior global-discrimination index method, and "random" for random selection. Other values return an error message.

For all methods but MLWI, MPWI, GDI, GDIP and random criteria, the provisional ability estimate must be supplied throught the theta argument (by default, it is equal to zero). For MLWI, MPWI, G and random criteria, this argument is ignored.

The available items are those that are not specified in the out argument. By default, out is NULL, which means that all items are available. Typically out contains the item numbers that have been already administered. Alternatively one can reduce the number of available items by specifying the argument nAvailable appropriately. The latter is a vector of 0 and 1 values, where 1 corresponds to an available item and 0 to a non-availbale item. It works similarly as for the function startItems. If both out and nAvailable are supplied, then all items from both vectors are discarded for next item selection.

For MEI, MEPV, MLWI, MPWI, KL, KLP, GDI, and GDIP methods, the provisional response pattern must be provided through the x argument. It must be of 0/1 entries and of the same length as the out argument. It is ignored with MFI, bOpt, thOpt, progressive, proportional and random criteria. Moreover, the range of integration (or posterior variance computation) is specified by the triplet parInt, where the first, second, and third value correspond to the arguments lower, upper and nqp of e.g., the MWI function, respectively.

The method, priorDist, priorPar, D, range and intPar arguments fix the choice of the ability estimator for MEI method. See the thetaEst function for further details. Note that priorDist and priorPar are also used to select the prior distribution with the MPWI and KLP methods. Finally, parInt is also used for numerical integration, among others for MLWI, MPWI, MEPV, KL and KLP methods.

Finally, for MEI criterion, the type of information function must be supplied through the infoType argument. It is equal to "observed" by default, which refers to the observed information function, and the other possible value is "Fisher" for Fisher information function. See the MEI funtion for further details. This argument is ignored if criterion is not "MEI".

The so-called *randomesque* approach is used to improve item exposure control (Kingsbury and Zara, 1989), which consists in selecting more than one item as the best items to be administered (according to the specified criterion). The final item that is administered is randomly chosen among this set of optimal items. The argument randomesque controls for the number of optimal items to be selected. The default value is 1, which corresponds to the usual framework of selecting the optimal item for next administration. Note that, for compatibility issues, if the number of remaining items is smaller than randomesque, the latter is replaced by this number of remaining items. The *randomesque* approach is not considered with the progressive or proportional item selection rules.

Dichotomous IRT models are considered whenever model is set to NULL (default value). In this case, itemBank must be a matrix with one row per item and four columns, with the values of the discrimination, the difficulty, the pseudo-guessing and the inattention parameters (in this order). These are the parameters of the four-parameter logistic (4PL) model (Barton and Lord, 1981).

Polytomous IRT models are specified by their respective acronym: "GRM" for Graded Response Model, "MGRM" for Modified Graded Response Model, "PCM" for Partical Credit Model, "GPCM" for Generalized Partial Credit Model, "RSM" for Rating Scale Model and "NRM" for Nominal Response Model. The itemBank still holds one row per item, end the number of columns and their content depends on the model. See genPolyMatrix for further information and illustrative examples of suitable polytomous item banks.

Control for content balancing is also possible, given two conditions: (a) the cbGroup argument is a vector with the names of the subgroups of items for content balancing (one value per item), and (b) the argument cbControl is a correctly specified list. The correct format for cbControl is a list with two elements. The first one is called names and holds the names of the subgroups of items (in the order that is prespecified by the user). The second element is called props and contains the (theoretical) proportions of items to be administered from each subgroup for content balancing. These proportions must be strictly positive but may not sum to one; in this case they are internally normalized to sum to one. Note that cbControl should be tested with the test.cbList function prior to using nextItem.

Under content balancing, the selection of the next item is done in several steps.

1. If no item was administered yet, one subgroup is randomly picked up and the optimal item from this subgroup is selected.
2. If at least one subgroup wasn't targeted yet by item selection, one of these subgroups is randomly picked up and the optimal item from this subgroup is selected.
3. If at least one item per subgroup was already administered, the empirical relative proportions of items administered per subgroup are computed, and the subgroup(s) whose difference between empirical and theoretical (i.e. given by cbControl$props) proportions is (are) selected. The optimal item is then selected from this subgroup for next administration (in case of several such groups, one group is randomly picked up first).

See Kingsbury and Zara (1989) for further details.

In case of content balancing control, three vectors of proportions are returned in the output list: $prior.prop contains the empirical relative proportions for items already administered (i.e. passed through the out argument); $post.prop contains the same empirical relative proportions but including the optimal item that was just selected; and $th.prop contains the theoretical proportions (i.e. those from cbControl$props or the normalized values). Note that NA values are returned when no control for content balancing is specified.

Value

A list with nine arguments:

item	the selected item (identified by its number in the item bank).
par	the vector of item parameters of the selected item.
info	the value of the MFI, Fisher's information, the MLWI, the MPWI, the MEI, the EPV, the unsigned distance between estimated ability and difficulty parameter (or the ability value were maximum Fisher information is provided) or NA (for "random" criterion) for the selected item and the current ability estimate.
criterion	the value of the criterion argument.
randomesque	the value of the randomesque argument.
name	either the name of the selected item (provided as the row name of the appropriate row in the item bank matrix) or NULL.
prior.prop	a vector with empirical proportions of items previously administered for each subgroup of items set by cbControl.
post.prop	a vector with empirical proportions of items previously administered, together with the one currently selected, for each subgroup of items set by cbControl.
th.prop	a vector with theoretical proportions given by cbControl$props.

Note

van der linden (1998) also introduced the Maximum Expected Posterior Weighted Information (MEPWI) criterion, as a mix of both MEI and MPWI methods (see also van der Linden and Pashley, 2000). However, Choi and Swartz (2009) established that this method is completely equivalent to MPWI. For this reason, MEPWI was not implemented here.

Author(s)

David Magis
Department of Psychology, University of Liege, Belgium
<david.magis@uliege.be>

Juan Ramon Barrada
Department of Psychology and Sociology, Universidad Zaragoza, Spain
<barrada@unizar.es>

Guido Corradi
Department of Psychology and Sociology, Universidad Zaragoza, Spain
<guidocor@gmail.com>

References

Baker, F.B. (1992). *Item response theory: parameter estimation techniques*. New York, NY: Marcel Dekker.

Barrada, J. R., Mazuela, P., and Olea, J. (2006). Maximum information stratification method for controlling item exposure in computerized adaptive testing. *Psicothema, 18*, 156-159.

Barrada, J. R., Olea, J., Ponsoda, V., and Abad, F. J. (2008). Incorporating randomness to the Fisher information for improving item exposure control in CATS. *British Journal of Mathematical and Statistical Psychology, 61*, 493-513. doi: 10.1348/000711007X230937

Barrada, J. R., Olea, J., Ponsoda, V., and Abad, F. J. (2010). A method for the comparison of item selection rules in computerized adaptive testing. *Applied Psychological Measurement*, *34*, 438-452. doi: 10.1177/0146621610370152

Barton, M.A., and Lord, F.M. (1981). *An upper asymptote for the three-parameter logistic item-response model*. Research Bulletin 81-20. Princeton, NJ: Educational Testing Service.

Chang, H.-H., and Ying, Z. (1996). A global information approach to computerized adaptive testing. *Applied Psychological Measurement, 20*, 213-229. doi: 10.1177/014662169602000303

Choi, S. W., and Swartz, R. J. (2009). Comparison of CAT item selection criteria for polytomous items. *Applied Psychological Measurement, 32*, 419-440. doi: 10.1177/0146621608327801

Haley, D.C. (1952). *Estimation of the dosage mortality relationship when the dose is subject to error*. Technical report no 15. Palo Alto, CA: Applied Mathematics and Statistics Laboratory, Stanford University.

Kaplan, M., de la Torre, J., and Barrada, J. R. (2015). New item selection methods for cognitive diagnosis computerized adaptive testing. *Applied Psychological Measurement, 39*, 167-188. doi: 10.1177/0146621614554650

Kingsbury, G. G., and Zara, A. R. (1989). Procedures for selecting items for computerized adaptive tests. *Applied Measurement in Education, 2*, 359-375. doi: 10.1207/s15324818ame0204_6

Leroux, A. J., Lopez, M., Hembry, I. and Dodd, B. G. (2013). A comparison of exposure control procedures in CATs using the 3PL model. *Educational and Psychological Measurement, 73*, 857-874. doi: 10.1177/0013164413486802

Magis, D. (2013). A note on the item information function of the four-parameter logistic model. *Applied Psychological Measurement, 37*, 304-315. doi: 10.1177/0146621613475471

Magis, D. and Barrada, J. R. (2017). Computerized Adaptive Testing with R: Recent Updates of the Package *catR*. *Journal of Statistical Software*, *Code Snippets*, *76(1)*, 1-18. doi: 10.18637/jss.v076.c01

Magis, D., and Raiche, G. (2012). Random Generation of Response Patterns under Computerized Adaptive Testing with the R Package *catR*. *Journal of Statistical Software*, *48 (8)*, 1-31. doi: 10.18637/jss.v048.i08

Revuelta, J., and Ponsoda, V. (1998). A comparison of item exposure control methods in computerized adaptive testing. *Journal of Educational Measurement, 35*, 311-327. doi: 10.1111/j.1745-3984.1998.tb00541.x

Segall, D. O. (2004). A sharing item response theory model for computerized adaptive testing. *Journal of Educational and Behavioral Statistics, 29*, 439-460. doi: 10.3102/10769986029004439

Urry, V. W. (1970). *A Monte Carlo investigation of logistic test models.* Unpublished doctoral dissertation. West Lafayette, IN: Purdue University.

van der Linden, W. J. (1998). Bayesian item selection criteria for adaptive testing. *Psychometrika, 63*, 201-216. doi: 10.1007/BF02294775

van der Linden, W. J., and Pashley, P. J. (2000). Item selection and ability estimation in adaptive testing. In W. J. van der Linden and C. A. W. Glas (Eds.), *Computerized adaptive testing. Theory and practice* (pp. 1-25). Boston, MA: Kluwer.

Veerkamp, W. J. J., and Berger, M. P. F. (1997). Some new item selection criteria for adaptive testing. *Journal of Educational and Behavioral Statistics, 22*, 203-226. doi: 10.3102/10769986022002203

See Also

MWI, MEI, KL, thetaEst, test.cbList, randomCAT, genPolyMatrix

Examples

```
## Dichotomous models ##

 # Loading the 'tcals' parameters
 data(tcals)

 # Item bank creation with 'tcals' item parameters
 prov <- breakBank(tcals)
 bank <- prov$itemPar
 cbGroup <- prov$cbGroup

 ## MFI criterion

 # Selecting the next item, current ability estimate is 0
 nextItem(bank, theta = 0) # item 63 is selected
```

```
# Selecting the next item, current ability estimate is 0 and item 63 is removed
nextItem(bank, theta = 0, out = 63) # item 10 is selected

# Selecting the next item, current ability estimate is 0 and items 63 and 10 are
# removed
nextItem(bank, theta = 0, out = c(63, 10)) # item 62 is selected

# Item exposure control by selecting three items (selected item will be either 10, 62
# or 63)
nextItem(bank, theta = 0, randomesque = 3)

# Fixing the random seed for randomesque selection
nextItem(bank, theta = 0, randomesque = 3, random.seed = 1)


## bOpt method

# Selecting the next item, current ability estimate is 0
nextItem(bank, theta = 0, criterion = "bOpt") # item 24 is selected

# Selecting the next item, current ability estimate is 0 and item 24 is removed
nextItem(bank, theta = 0, out = 24, criterion = "bOpt")


## thOpt method

# Selecting the next item, current ability estimate is 0
nextItem(bank, theta = 0, criterion = "thOpt") # item 76 is selected

# Selecting the next item, current ability estimate is 0 and item 76 is removed
nextItem(bank, theta = 0, out = 76, criterion = "thOpt") # item 70 is selected


## MLWI and MPWI methods

# Selecting the next item, current response pattern is 0 and item 63 was administered
# first
nextItem(bank, x = 0, out = 63, criterion = "MLWI")
nextItem(bank, x = 0, out = 63, criterion = "MPWI")

# Selecting the next item, current response pattern is (0,1) and item 19 is removed
nextItem(bank, x = c(0, 1), out = c(63, 19), criterion = "MLWI")
nextItem(bank, x = c(0, 1), out = c(63, 19), criterion = "MPWI")

## Not run:

## MEI method

# Selecting the next item, current response pattern is 0 and item 63 was administered
# first
# Ability estimation by WL method
th <- thetaEst(rbind(bank[63,]), 0, method = "WL")
```

```
nextItem(bank, x = 0, out = 63, theta = th, criterion = "MEI") # item 49 is selected

# With Fisher information
nextItem(bank, x = 0, out = 63, theta = th, criterion = "MEI", infoType = "Fisher")
  # item 10 is selected


## MEPV method

# Selecting the next item, current response pattern is 0 and item 63 was administered
# first
# Ability estimation by WL method
nextItem(bank, x = 0, out = 63, theta = th, criterion = "MEPV") # item 19 is selected


## KL and KLP methods

# Selecting the next item, current response pattern is 0 and item 63 was administered
# first
# Ability estimation by WL method
nextItem(bank, x = 0, out = 63, theta = th, criterion = "KL") # item 19 is selected
nextItem(bank, x = 0, out = 63, theta = th, criterion = "KLP") # item 44 is selected


## GDI and GDIP methods

# Selecting the next item, current response pattern is 0 and item 63 was administered
# first
nextItem(bank, x = 0, out = 63, criterion = "GDI") # item 49 is selected
nextItem(bank, x = 0, out = 63, criterion = "GDIP") # item 44 is selected


## Progressive method

# Selecting the next item, current ability estimate is 0 and item 63 was administered
# first
# (default options: "length" rule with "thr = 20")
nextItem(bank, out = 63, theta = 0, criterion = "progressive")
nextItem(bank, out = 63, theta = 0, criterion = "progressive")
 # result can be different!


## Proportional method

# Selecting the next item, current ability estimate is 0 and item 63 was administered
# first
# (default options: "length" rule with "thr = 20")
nextItem(bank, out = 63, theta = 0, criterion = "proportional")
nextItem(bank, out = 63, theta = 0, criterion = "proportional")
 # result can be different!


## Random method
```

```
# Selecting the next item, item 63 was administered first
nextItem(bank, out = 63, criterion = "random")
nextItem(bank, out = 63, criterion = "random")  # may produce a different result


## Content balancing

# Creation of the 'cbList' list with arbitrary proportions
cbList <- list(names = c("Audio1", "Audio2", "Written1", "Written2", "Written3"),
       props = c(0.1, 0.2, 0.2, 0.2, 0.3))

# Selecting the next item, MFI criterion, current ability estimate is 0, items 12, 33,
# 46 and 63 previously administered
nextItem(bank, theta = 0, out = c(12, 33, 46, 63), cbControl = cbList,
         cbGroup = cbGroup)  # item 70 is selected


## Polytomous models ##

# Generation of an item bank under GRM with 100 items and at most 4 categories
m.GRM <- genPolyMatrix(100, 4, "GRM")
m.GRM <- as.matrix(m.GRM)

# Current ability estimate is 0
# Selecting the next item, current response pattern is 1 and item 70 was administered
# first

## MFI method
nextItem(m.GRM, model = "GRM", theta = 0, criterion = "MFI", out = 70)

## Progressive method
nextItem(m.GRM, model = "GRM", theta = 0, criterion = "progressive", out = 70)

## KL method
nextItem(m.GRM, model = "GRM", theta = 0, criterion = "KL", out = 70, x = 1)

## MFI with content balancing
cbList <- list(names = c("Audio1","Audio2","Written1","Written2", "Written3"),
       props = c(0.1,0.2,0.2,0.2,0.3))
m.GRM<-genPolyMatrix(100, 4, model = "GRM", cbControl = cbList)
bank<-breakBank(m.GRM)
nextItem(bank$itemPar, model = "GRM", theta = 0, criterion = "MFI", out = 70,
 cbControl = cbList, cbGroup = bank$cbGroup)


# Loading the cat_pav data
data(cat_pav)
cat_pav <- as.matrix(cat_pav)

# Current ability estimate is 0
# Selecting the next item, current response pattern is 1 and item 15 was administered
# first
```

```
## MFI method
nextItem(cat_pav, model = "GPCM", theta = 0, criterion = "MFI", out = 15)

## Progressive method
nextItem(cat_pav, model = "GPCM", theta = 0, criterion = "progressive", out = 15)

## KL method
nextItem(cat_pav, model = "GPCM", theta = 0, criterion = "KL", out = 15, x = 1)

## MFI with content balancing
cbList <- list(names = c("Audio1", "Audio2", "Written1", "Written2", "Written3"),
       props = c(0.1, 0.2, 0.2, 0.2, 0.3))
cat_pav<-genPolyMatrix(100, 4, model = "GPCM", cbControl = cbList)
bank<-breakBank(cat_pav)
nextItem(bank$itemPar, model = "GPCM", theta = 0, criterion = "MFI", out = 15,
         cbControl = cbList, cbGroup = bank$cbGroup)

## End(Not run)
```

OIi *Observed information function (dichotomous and polytomous models)*

Description

This command returns the observed information functions for a given ability value and a given matrix of item parameters, either under the 4PL model or any suitable polytomous IRT model.

Usage

```
OIi(th, it, x, model = NULL, D = 1)
```

Arguments

th	numeric: the ability value.
it	numeric: a suitable matrix of item parameters. See **Details**.
x	numeric: a vector of item responses.
model	either NULL (default) for dichotomous models, or any suitable acronym for polytomous models. Possible values are "GRM", "MGRM", "PCM", "GPCM", "RSM" and "NRM". See **Details**.
D	numeric: the metric constant. Default is D=1 (for logistic metric); D=1.702 yields approximately the normal metric (Haley, 1952). Ignored if model is not NULL.

Details

The observed information function for item *j* is given by

$$OI_j = -\frac{\partial^2}{\partial\theta^2} \log L(\theta|x_j)$$

where θ is the ability level, L is the likelihood function and x_j is the item response.

For dichotomous IRT models with success probability $P_j(\theta)$, it takes the following form:

$$OI_j = \frac{P_j\, Q_j\, {P'_j}^2 - (x_j - P_j)\,[P_j\, Q_j\, P''_j + {P_j}^2\,(P_j - Q_j]}{{P_j}^2\, {Q_j}^2}$$

where $P_j = P_j(\theta)$, $Q_j = 1 - P_j$ and P'_j and P''_j are the first and second derivatives of P_j respectively.

For polytomous IRT models, set X_j as the item response, taking values $k \in \{0, 1, ..., g_j\}$. Set $P_{jk}(\theta) = Pr(X_j = k|\theta)$ as the probability of answering response category k and set τ_{jk} as the boolean factor equal to 1 if $X_j = k$ and 0 otherwise. Then, the observed information function for item j takes the following form:

$$OI_j = \sum_{k=0}^{g_j} \tau_{jk} \left(\frac{{P'_{jk}(\theta)}^2}{{P_{jk}(\theta)}^2} - \frac{P''_{jk}(\theta)}{P_{jk}(\theta)} \right)$$

with the same notations for the first and second derivatives as above.

Under the 2PL model, the observed information function is exactly equal to Fisher's information function

$$I_j = -E\left[\frac{\partial^2}{\partial\theta^2} \log L(\theta|x_j)\right] = \frac{{P'_j}^2}{P_j Q_j}$$

(van der Linden, 1998; Veerkamp, 1996).

Dichotomous IRT models are considered whenever `model` is set to `NULL` (default value). In this case, `it` must be a matrix with one row per item and four columns, with the values of the discrimination, the difficulty, the pseudo-guessing and the inattention parameters (in this order). These are the parameters of the four-parameter logistic (4PL) model (Barton and Lord, 1981).

Polytomous IRT models are specified by their respective acronym: `"GRM"` for Graded Response Model, `"MGRM"` for Modified Graded Response Model, `"PCM"` for Partical Credit Model, `"GPCM"` for Generalized Partial Credit Model, `"RSM"` for Rating Scale Model and `"NRM"` for Nominal Response Model. The `it` still holds one row per item, end the number of columns and their content depends on the model. See `genPolyMatrix` for further information and illustrative examples of suitable polytomous item banks.

The observed information function is used to compute some item selection criteria, such as the Maximum Expected Information (MEI). See `nextItem` and `MEI` for further details.

Value

A vector with the observed item informations (one per item).

Author(s)

David Magis
Department of Psychology, University of Liege, Belgium
<david.magis@uliege.be>

References

Barton, M.A., and Lord, F.M. (1981). *An upper asymptote for the three-parameter logistic item-response model*. Research Bulletin 81-20. Princeton, NJ: Educational Testing Service.

Haley, D.C. (1952). *Estimation of the dosage mortality relationship when the dose is subject to error*. Technical report no 15. Palo Alto, CA: Applied Mathematics and Statistics Laboratory, Stanford University.

Magis, D. and Barrada, J. R. (2017). Computerized Adaptive Testing with R: Recent Updates of the Package *catR*. *Journal of Statistical Software*, *Code Snippets*, *76(1)*, 1-18. doi: 10.18637/jss.v076.c01

Magis, D., and Raiche, G. (2012). Random Generation of Response Patterns under Computerized Adaptive Testing with the R Package *catR*. *Journal of Statistical Software*, *48 (8)*, 1-31. doi: 10.18637/jss.v048.i08

van der Linden, W. (1998). Bayesian item selection criteria for adaptive testing. *Psychometrika, 63*, 201-216. doi: 10.1007/BF02294775

Veerkamp, W. J. J. (1996). *Statistical inference for adaptive testing*. Internal report. Enschede, The Netherlands: University of Twente.

See Also

nextItem, MEI

Examples

```
## Dichotomous models ##

 # Loading the 'tcals' parameters
 data(tcals)

 # Selecting item parameters only
 tcals <- as.matrix(tcals[,1:4])

 # Observed information functions
 # (various th, x and D values)
 OIi(th = 0, tcals, x = 0)
 OIi(th = 0, tcals, x = 0, D = 1.702)
 OIi(th = 0, tcals, x = 1)
 OIi(th = 1, tcals, x = 1)


## Polytomous models ##

 # Generation of an item bank under GRM with 100 items and at most 4 categories
```

```
m.GRM <- genPolyMatrix(100, 4, "GRM")
m.GRM <- as.matrix(m.GRM)

# Creation of a response pattern (true ability level 0)
set.seed(1)
x <- genPattern(0, m.GRM, model = "GRM")

# Observed information functions (various th values)
OIi(th = 0, m.GRM, x, model = "GRM")
OIi(th = 1, m.GRM, x, model = "GRM")

# Loading the cat_pav data
data(cat_pav)
cat_pav <- as.matrix(cat_pav)

# Creation of a response pattern (true ability level 0)
set.seed(1)
x <- genPattern(0, cat_pav, model = "GPCM")

# Observed information functions (various 'th' values)
OIi(th = 0, cat_pav, x, model = "GPCM")
OIi(th = 1, cat_pav, x, model = "GPCM")
```

`Pi` *Item response probabilities, first, second and third derivatives (dichotomous and polytomous models)*

Description

This command returns the item response probabilities for a given ability value and a given matrix of item parameters under either the 4PL model or any suitable polytomous model. Numerical values of the first, second and third derivatives of the response probabilities are also returned.

Usage

```
Pi(th, it, model = NULL, D = 1)
```

Arguments

`th`	numeric: the ability value.
`it`	numeric: a suitable matrix of item parameters. See **Details**.
`model`	either `NULL` (default) for dichotomous models, or any suitable acronym for polytomous models. Possible values are `"GRM"`, `"MGRM"`, `"PCM"`, `"GPCM"`, `"RSM"` and `"NRM"`. See **Details**.
`D`	numeric: the metric constant. Default is `D=1` (for logistic metric); `D=1.702` yields approximately the normal metric (Haley, 1952). Ignored if `model` is not `NULL`.

Details

Whatever the IRT model, the response probabilities and first, second, and third derivatives are computed algebraically. These derivatives are necessary for both the estimation of ability and the computation of related standard errors.

Dichotomous IRT models are considered whenever `model` is set to `NULL` (default value). In this case, `it` must be a matrix with one row per item and four columns, with the values of the discrimination, the difficulty, the pseudo-guessing and the inattention parameters (in this order). These are the parameters of the four-parameter logistic (4PL) model (Barton and Lord, 1981).

Polytomous IRT models are specified by their respective acronym: `"GRM"` for Graded Response Model (Samejima, 1969), `"MGRM"` for Modified Graded Response Model (Muraki, 1990), `"PCM"` for Partical Credit Model (Masters, 1982), `"GPCM"` for Generalized Partial Credit Model (Muraki, 1992), `"RSM"` for Rating Scale Model (Andrich, 1978) and `"NRM"` for Nominal Response Model (Bock, 1972). The `it` still holds one row per item, end the number of columns and their content depends on the model. See `genPolyMatrix` for further information and illustrative examples of suitable polytomous item banks.

The output list contains the response probabilities and the respective derivatives. In case of dichotomous models, only a vector of such values is returned, with one value per item. In case of polytomous models, matrices are returned instead, with one row per item and one column per response category. In case of unequal numbers of response categories (which may happen under GRM, PCM, GPCM and NRM), values for empty response categories are returned as `NA` values.

Value

Under dichotomous IRT models, a list with four arguments:

`Pi`	the vector with response probabilities (one value per item).
`dPi`	the vector with first derivatives of the response probabilities (one value per item).
`d2Pi`	the vector with second derivatives of the response probabilities (one value per item).
`d3Pi`	the vector with third derivatives of the response probabilities (one value per item).

Under polytomous IRT models, the aforementioned vectors are replaced by matrices with one row per item (labeled as `Item1`, `Item2` etc.) and one row per response category.

Note

For dichotomous IRT models, response probabilites exactly equal to zero are returned as `1e-10` values, as well as probabilities exactly equal to one which are returned as `1-1e-10` values. This is to permit the computation of ability estimates (with the `thetaEst` function) in such extreme cases.

Many thanks to Pan Tong (University of Texas MD Anderson Cancer Center, USA) who noticed this problem.

Author(s)

David Magis
Department of Psychology, University of Liege, Belgium
<david.magis@uliege.be>

References

Andrich, D. (1978). A rating formulation for ordered response categories. *Psychometrika, 43*, 561-573. doi: 10.1007/BF02293814

Barton, M.A., and Lord, F.M. (1981). *An upper asymptote for the three-parameter logistic item-response model*. Research Bulletin 81-20. Princeton, NJ: Educational Testing Service.

Bock, R. D. (1972). Estimating item parameters and latent ability when responses are scored in two or more nominal categories. *Psychometrika, 37*, 29-51. doi: 10.1007/BF02291411

Haley, D.C. (1952). *Estimation of the dosage mortality relationship when the dose is subject to error*. Technical report no 15. Palo Alto, CA: Applied Mathematics and Statistics Laboratory, Stanford University.

Magis, D. and Barrada, J. R. (2017). Computerized Adaptive Testing with R: Recent Updates of the Package *catR*. *Journal of Statistical Software*, *Code Snippets*, *76(1)*, 1-18. doi: 10.18637/jss.v076.c01

Magis, D., and Raiche, G. (2012). Random Generation of Response Patterns under Computerized Adaptive Testing with the R Package *catR*. *Journal of Statistical Software*, *48 (8)*, 1-31. doi: 10.18637/jss.v048.i08

Masters, G. N. (1982). A Rasch model for partial credit scoring. *Psychometrika, 47*, 149-174. doi: 10.1007/BF02296272

Muraki, E. (1990). Fitting a polytomous item response model to Likert-type data. *Applied Psychological Measurement, 14*, 59-71. doi: 10.1177/014662169001400106

Muraki, E. (1992). A generalized partial credit model: Application of an EM algorithm. *Applied Psychological Measurement, 16*, 19-176. doi: 10.1177/014662169201600206

Samejima, F. (1969). *Estimation of latent ability using a response pattern of graded scores*. Psychometrika Monograph (vol. 17).

See Also

Ii, thetaEst

Examples

```
## Dichotomous models ##

 # Loading the 'tcals' parameters
 data(tcals)

 # Selecting item parameters only
 tcals <- as.matrix(tcals[,1:4])

 # Response probabilities and derivatives (various th and D values)
 Pi(th = 0, tcals)
 Pi(th = 0, tcals, D = 1.702)
 Pi(th = 1, tcals)

## Polytomous models ##
```

```
# Generation of an item bank under GRM with 100 items and at most 4 categories
m.GRM <- genPolyMatrix(100, 4, "GRM")
m.GRM <- as.matrix(m.GRM)

# Computation of probabilities and derivatives for ability level 0
Pi(0, m.GRM, model = "GRM")

# Loading the cat_pav data
data(cat_pav)
cat_pav <- as.matrix(cat_pav)

# Computation of probabilities and derivatives for ability level 1
Pi(1, cat_pav, model = "GPCM")
```

---

randomCAT *Random generation of adaptive tests (dichotomous and polytomous models)*

Description

This command generates a response pattern to an adaptive test, for a given item bank (with either dichotomous or polytomous models), a true ability level, and several lists of CAT parameters (starting items, stopping rule, provisional and final ability estimators).

Usage

```
randomCAT(trueTheta, itemBank, model = NULL, responses = NULL, min.length = 0,
  a.stratified = NULL, genSeed = NULL, cbControl = NULL, nAvailable = NULL,
  start = list(fixItems = NULL, seed = NULL, nrItems = 1, theta = 0, D = 1,
  randomesque = 1, random.seed = NULL, startSelect = "MFI", cb.control = FALSE,
  random.cb = NULL), test = list(method = "BM", priorDist = "norm",
  priorPar = c(0, 1), weight = "Huber", tuCo = 1, sem.type = "classic",
  sem.exact = FALSE, se.ase = 10, range = c(-4, 4), D = 1, parInt = c(-4, 4, 33),
  itemSelect = "MFI", infoType = "observed", randomesque = 1,
  random.seed = NULL, AP = 1, proRule = "length", proThr = 20,
  constantPatt = NULL), stop = list(rule = "length", thr = 20, alpha = 0.05),
  final = list(method = "BM", priorDist = "norm", priorPar = c(0, 1),
  weight = "Huber", tuCo = 1, sem.type = "classic", sem.exact = FALSE,
  range = c(-4, 4), D = 1, parInt = c(-4, 4, 33), alpha = 0.05),
  allTheta = FALSE, save.output = FALSE, output = c("path", "name", "csv"))
## S3 method for class 'cat'
print(x, ...)
## S3 method for class 'cat'
plot(x, ci = FALSE, alpha = 0.05, trueTh = TRUE, classThr = NULL,
  save.plot = FALSE, save.options = c("path", "name", "pdf"),...)
```

Arguments

trueTheta	numeric: the value of the true ability level.
itemBank	numeric: a suitable matrix of item parameters (possibly augmented by group membership for content balancing). See **Details**.
model	either NULL (default) for dichotomous models, or any suitable acronym for polytomous models. Possible values are "GRM", "MGRM", "PCM", "GPCM", "RSM" and "NRM". See **Details**.
responses	either NULL (default) or a vector of pre-specified item responses with as many components as the rows of itemBank. See **Details**.
min.length	integer: the minimum number of items to administer default is zero so no minimum test length). The CAT will not stop before the number of administered items equals min.length, even if the stopping rule is satisfied.
a.stratified	either NULL (default), an integer value or a vector of integer values to set up a-stratified sampling. See **Details**.
genSeed	either a numeric value to fix the random generation of responses pattern or NULL (default). Ignored if responses is not NULL. See **Details**.
cbControl	either a list of accurate format to control for content balancing, or NULL. See **Details**.
nAvailable	either a boolean vector indicating which items (denoted by 1's) are available at the start of the test and which (denoted by 0's) are not, or NULL (default). See **Details**.
start	a list with the options for starting the adaptive test. See **Details**.
test	a list with the options for provisional ability estimation and next item selection. See **Details**.
stop	a list with the options of the stopping rule. See **Details**.
final	a list with the options for final ability estimation. See **Details**.
allTheta	logical: should all provisional ability estimates and standard errors be computed and returned (including among the starting items)? Default is FALSE, meaning that provisional ability estimates and standard errors are computed only after the selection of the starting items. Ignored if the $nrItems of the start list is equal to zero.
save.output	logical: should the output be saved in an external text file? (default is FALSE).
output	character: a vector of three components. The first component is either the file path to save the output of "path" (default), the second component is the name of the output file, and the third component is the file type, either "txt" or "csv" (default). See **Details**.
x	an object of class "cat", typically an output of randomCAT function.
ci	logical: should the confidence intervals be plotted for each provisional ability estimate? (default is TRUE).
alpha	numeric: the significance level for provisional confidence intervals (default is 0.05). Ignored if ci is FALSE.
trueTh	logical: should the true ability level be drawn by a horizontal line? (default is TRUE).

classThr	either a numeric value giving the classification threshold to be displayed, or NULL.
save.plot	logical: should the plot be saved in an external figure? (default is FALSE).
save.options	character: a vector of three components. The first component is either the file path or "path" (default), the second component is the name of the output file or ,"name" (default), and the third component is the file extension, either "pdf" (default) or "jpeg". Ignored if save.plot is FALSE. See **Details**.
...	other generic arguments to be passed to print and plot functions.

Details

The randomCAT function generates an adaptive test using an item bank specified by arguments itemBank and model, and for a given true ability level specified by argument trueTheta.

Dichotomous IRT models are considered whenever model is set to NULL (default value). In this case, itemBank must be a matrix with one row per item and four columns, with the values of the discrimination, the difficulty, the pseudo-guessing and the inattention parameters (in this order). These are the parameters of the four-parameter logistic (4PL) model (Barton and Lord, 1981).

Polytomous IRT models are specified by their respective acronym: "GRM" for Graded Response Model, "MGRM" for Modified Graded Response Model, "PCM" for Partical Credit Model, "GPCM" for Generalized Partial Credit Model, "RSM" for Rating Scale Model and "NRM" for Nominal Response Model. The itemBank still holds one row per item, end the number of columns and their content depends on the model. See genPolyMatrix for further information and illustrative examples of suitable polytomous item banks.

By default all item responses will be randomly drawn from parent distribution set by the item bank parameters of the itemBank matrix (using the genPattern function for instance). Moreover, the random generation of the item responses can be fixed (for e.g., replication purposes) by assigning some numeric value to the genSeed argument. By default this argument is equal to NULL so the random seed is not fixed (and two successive runs of randomCAT will usually lead to different response patterns).

It is possible, however, to provide a full response pattern of previously recorded responses to each of the item bank, for instance for post-hoc simulations. This is done by providing to the responses argument a vector of binary entries (without missing values). By default responses is set to NULL and item responses will be drawn from the item bank parameters.

With the aforementioned item bank structures, content balancing cannot be controled and cbControl must be set to NULL (default value). Otherwise this will most often lead to an error. In order to allow for content balancing control:

1. the itemBank must be updated with an additional column holding the group names;
2. the cbControl argument must be set properly as a list with group names and theoretical proportions for content balancing.

See the nextItem function for further details on how to specify cbControl properly and under which conditions it is operational (see Kingsbury and Zara, 1989, for further details). Separation of the item parameters and the vector of group membership is performed internally through the breakBank function (and thus should not be performed prior to CAT generation).

An alternative method to control for content balancing is to performed a-stratified sampling (Chang and Ying, 1999). This is specified by providing the a.stratified argument either an integer value

(i.e., the number of strata) or a vector of integer values (i.e., the number of items per strata). The value to be provided corresponds to the K argument of the aStratified function. Note that this is allowed only with dichotomous IRT models and polytomous GRM, MGRM and GPCM models. By default, a.stratified is NULL and a-stratified sampling is not performed.

The test specification is made by means of four lists of options: one list for the selection of the starting items, one list with the options for provisional ability estimation, one list to define the stopping rule, and one list with the options for final ability estimation. These lists are specified respectively by the arguments start, test, stop and final.

The start list can contain one or several of the following arguments:

- fixItems: either a vector of integer values, setting the items to be administered as first items, or NULL (default) to let the function select the items.
- seed: either a numeric value to fix the random seed for item selection, NA to randomly select the items withour fixing the random seed, or NULL (default) to select the items on the basis of their difficulty level. Ignored if fixItems is not NULL.
- nrItems: numeric, the number of starting items to be randomly selected (default is 1). Can be equal to zero to avoid initial selection of items (see **Details**). Used only if fixItems is NULL and seed is not NULL.
- theta: numeric, a vector of the initial ability levels for selecting the first items (default is the single value 0). Ignored if either fixItems or seed is not NULL. See startItems for further details.
- D: numeric, the metric constant. Default is D=1 (for logistic metric); D=1.702 yields approximately the normal metric (Haley, 1952). Ignored if model is not NULL and if startSelect is not "MFI".
- randomesque: integer, the number of 'randomesque' items to be picked up optimally for each value of theta vector, before random selection of a single one. Ignored if either fixItems or seed is not NULL. See startItems for further details.
- random.seed: either NULL (default) or a numeric value to fix the random seed of randomesque selection of the items. Ignored if either fixItems or seed is not NULL.
- startSelect: the method for selecting the first items of the test, with possible values "bOpt" and "MFI" (default). Ignored if either fixItems or seed is not NULL. See startItems for further details.
- cb.control: logical value indicating whether control for content balancing should also be done when selecting the starting items. Default is FALSE. Ignored if argument cbControl is NULL.
- random.cb: either NULL (default) or a numeric value to fix the selection of subgroups of items in case of content balancing control with starting items. Ignored if either cbControl is NULL or if start$cb.control is FALSE.

These arguments are passed to the function startItems to select the first items of the test.

If the argument nrItems is set to zero, then no starting item is selected and the adaptive process starts with a provisional ability level equal to the value of argument theta (or its default). Moreover, the likelihood function is then set as a flat, uniform function on the whole ability range. See the nextItem function for further details.

The test list can contain one or several of the following arguments:

- method: a character string to specify the method for ability estimation. Possible values are: "BM" (default) for Bayesian modal estimation (Birnbaum, 1969), "ML" for maximum likelihood estimation (Lord, 1980), "EAP" for expected a posteriori (EAP) estimation (Bock and Mislevy, 1982), "WL" for weighted likelihood estimation (Warm, 1989) and "ROB" for robust estimation (Schuester and Yuan, 2011).
- priorDist: a character string which sets the prior distribution. Possible values are: "norm" (default) for normal distribution, "unif" for uniform distribution, and "Jeffreys" for Jeffreys' noninformative prior distribution (Jeffreys, 1939, 1946). Ignored if method is neither "BM" nor "EAP".
- priorPar: a vector of two numeric components, which sets the parameters of the prior distribution. If (method="BM" or method=="EAP") and priorDist="norm", the components of priorPar are respectively the mean and the standard deviation of the prior normal density. If (method="BM" or method="EAP") and priorDist="unif", the components of priorPar are respectively the lower and upper bound of the prior uniform density. Ignored in all other cases. By default, priorPar takes the parameters of the prior standard normal distribution (i.e., priorPar=c(0,1)). In addition, priorPar also provides the prior parameters for the computation of MLWI and MPWI values for next item selection (see nextItem for further details).
- weight: the type of weight function for the robust estimator. Possible values are "Huber" (default) and "Tukey". Ignored if method is not "ROB" or if model is not NULL.
- tuCo: the value of the tuning constant for the weight function (default is 1, suitable with "Huber" weight). Ignored if method is not "ROB" or if model is not NULL.
- sem.type: the type of ASE formula to be used, either "classic" (default) or "new". Ignored if method is neither "BM" nor "WL", or if model is not NULL.
- sem.exact: a logical value indicating whether *exact* standard error should be computed instead of asymptotic standard error. (default is FALSE). Ignored if model is not NULL.
- se.ase: an integer value specifying the maximum number of item responses that are considered for computing the exact SE values, before switching to ASE values (due to computational time constraints). Default is 10; that is, ASE values are computed from the 11-th administered item on. Ignored if model is not NULL or if sem.exact id FALSE.
- range: the maximal range of ability levels, set as a vector of two numeric components. The ability estimate will always lie to this interval (set by default to [-4, 4]). Ignored if method=="EAP".
- D: the value of the metric constant. Default is D=1 for logistic metric. Setting D=1.702 yields approximately the normal metric (Haley, 1952). Ignored if model is not NULL.
- parInt: a numeric vector of three components, holding respectively the values of the arguments lower, upper and nqp of the eapEst, eapSem and MWI commands. It specifies the range of quadrature points for numerical integration, and is used for computing the EAP estimate, its standard error, and the MLWI and MPWI values for next item selection. Default vector is (-4, 4, 33), thus setting the range from -4 to 4 by steps of 0.25. Ignored if method is not "EAP" and if itemSelect is neither "MLWI" nor "MPWI".
- itemSelect: the rule for next item selecion, with possible values "MFI" (default) for maximum Fisher information criterion; "bOpt" for optimal ability-difficulty match (or Urry's procedure) (not available if model is not NULL); "thOpt" for optimal theta selection (not available if model is not NULL); "MLWI" and "MPWI" for respectively maximum likelihood and

posterior weighted information criterion; "MEPV" for minimum expected posterior variance; "MEI" for maximum expected information; "KL" and "KLP" for Kullback-Leibler and posterior Kullback-Leibler information methods; "progressive" and "proportional" for progressive and proportional methods; ; and "random" for random selection. For further details, see nextItem.

- infoType: character: the type of information function to be used for next item selection. Possible values are "observed" (default) for observed information function, and "Fisher" for Fisher information function. Ignored if itemselect is not "MEI".
- randomesque: integer: the number of items to be chosen from the next item selection rule, among those the next item to be administered will be randomly picked up. Default value is 1 and leads to usual selection of the optimal item (Kingsbury and Zara, 1989).
- random.seed: either NULL (default) or a numeric value to fix the random seed of randomesque selection of the items. Ignored if either fixItems or seed is not NULL.
- AP: the acceleration parameter required for progressive and proportional methods, with default value 1. Ignored with all other selection methods.
- proRule: the stopping rule considered for progressive and proportional methods, with possible values "length" (default), "precision" or both. Ignored with all other selection methods.
- proThr: the stopping rule threshold considered for progressive and proportional methods. Default value is 20. Ignored with all other selection methods.
- constantPatt: character: the method to estimate ability in case of constant pattern (i.e. only correct or only incorrect responses). Can be either NULL (default), "BM", "EAP", "WL", "fixed4", "fixed7" or "var". *Currently only implemented for dichotomous IRT models*.

These arguments are passed to the functions thetaEst and semTheta to estimate the ability level and the standard error of this estimate. In addition, some arguments are passed to nextItem to select the next item appropriately.

The stop list can contain one or several of the following arguments:

- rule: a vector of character strings specifying the stopping rules. Possible values are: "length" (default), to stop the test after a pre-specified number of items administered; "precision", to stop the test when the provisional standard error of ability becomes less than or equal to the pre-specified value; "classification", for which the test ends whenever the provisional confidence interval (set by the alpha argument) does not hold the classification threshold anymore (this is also called the ACI rule; see e.g. Thomson, 2009); and "minInfo" to stop the test if the maximum item information of the available items at current ability estimate is smaller than the prespecified threshold. Can take a single value.
- thr: a vector of numeric values fixing the threshold(s) of the stopping rule(s). If rule="length", thr is the maximal number of items to be administered. If rule="precision", thr is the precision level (i.e. the standard error) to be reached before stopping. If rule="classification", thr corresponds to the ability level which serves as a classification rule (i.e. which must not be covered by the provisional confidence interval). Finally, If rule="minInfo", thr corresponds to the minimum item information that can be observed in the bank of remaining available items. The "classification" rule is not available for the progressive and proportional item selection rules.

- `alpha`: the significance (or α) level for computing the priovisional confidence interval of ability. Ignored if `rule` is not `"classification"`. *Important:* the `thr` value smust be sorted in the same order of appearance as the `rule` methods.

Eventually, the `final` list can contain one or several arguments of the `test` list (with possiblly different values), as well as the additional `alpha` argument. The latter specifies the α level of the final confidence interval of ability, which is computed as

$$[\hat{\theta} - z_{1-\alpha/2}\ se(\hat{\theta}); \hat{\theta} + z_{1-\alpha/2}\ se(\hat{\theta})]$$

where $\hat{\theta}$ and $se(\hat{\theta})$ are respectively the ability estimate and its standard error. Note that the argument `itemSelect` of the `test` list is not used for final estimation of the ability level, and is therefore not allowed into the `final` list.

If some arguments of these lists are missing, they are automatically set to their default value. The contents of the lists is checked with the `testList` function, and the adaptive test is generated only if the lists are adequately defined. Otherwise, a message error is printed. Note that the `testList` function works for both dichotomous and polytomous models.

Usually the ability estimates and related standard errors are computed right after the administration of the starting items (that is, if *k* starting items are administered, the first *(k-1)* ability levels and standard errors are missing). This can however be avoided by fixing the argument `allTheta` to `TRUE` (by default it is `FALSE`). In this case, all provisional ability estimates and standard errors are computed and returned, but in the display of th output file, the first *(k-1)* abilities and standard errors are printed in parentheses (otherwise they are returned as `NA` values). Note that `allTheta` is ignored if no starting item was selected (that is, if argument `nrItems` of the `start` list is set to zero).

The output of `randomCAT`, as displayed by the `print.cat` function, can be stored in a text file provided that `save.output` is set to `TRUE` (the default value `FALSE` does not execute the storage). In this case, the `(output` argument mus hold three character values: the path to where the output file must be stored, the name of the output file, and the type of output file. If the path is not provided (i.e. left to its default value `"path"`), it will be saved in the default working directory. The default name is `"name"`, and the default type is `"csv"`. Any other value yields a text file. See the **Examples** section for an illustration.

The function `plot.cat` represents the set of provisional and final ability estimates throughout the test. Corresponding confidence intervals (with confidence level defined by the argument `alpha`) are also drawn if `ci=TRUE` (which is *not* the default value), except when stepsize adjustment was made for constant patterns (as the standard error cannot be estimated at this stage). The true ability level can be drawn by a horizontal solid line by specifying `trueTh=TRUE` (which is the default value); setting it to `FALSE` will undo the drawing. Finally, any classification threshold can be additionally displayed by specifying a numeric value to the argument `classThr`. The default value `NULL` does not display any threshold.

Finally, the plot can be saved in an external file, either as PDF or JPEG format. First, the argument `save.plot` must be set to `TRUE` (default is `FALSE`). Then, the file path for figure storage, the name of the figure and its format are specified through the argument `save.options`, all as character strings. See the `Examples` section for further information and a practical example.

Value

The function `randomCAT` returns a list of class "cat" with the following arguments:

trueTheta	the value of the trueTheta argument.
model	the value of the model argument.
testItems	a vector with the items that were administered during the test.
itemPar	a matrix with the parameters of the items administered during the test.
itemNames	either a vector wit the names of the selected items during the CAT, or NULL.
pattern	the generated response pattern (as vector of 0 and 1 entries).
thetaProv	a vector with the provisional ability estimates.
seProv	a vector with the standard errors of the provisional ability estimates.
ruleFinal	either the stopping rule(s) that was (were) satisfied to make the CAT stop, or NULL.
thFinal	the final ability estimate.
seFinal	the standard error of the final ability estimate.
ciFinal	the confidence interval of the final ability estimate.
min.length	the value of the min.length argument.
a.stratified	the value of the a.stratified argument.
genSeed	the value of the genSeed argument.
startFixItems	the value of the start$fixItems argument (or its default value if missing).
startSeed	the value of the start$seed argument (or its default value if missing).
startNrItems	the value of the start$nrItems argument (or its default value if missing).
startTheta	the value of the start$theta argument (or its default value if missing).
startD	the value of the start$D argument (or its default value if missing).
startRandomesque	the value of the start$randomesque argument (or its default value if missing).
startThStart	the starting ability values used for selecting the first items of the test.
startSelect	the value of the start$startSelect argument (or its default value if missing).
startCB	logical value, being TRUE if both cbControl is not NULL and start$cb.control is TRUE, and FALSE otherwise.
provMethod	the value of the test$method argument (or its default value if missing).
provDist	the value of the test$priorDist argument (or its default value if missing).
provPar	the value of the test$priorPar argument (or its default value if missing).
provWeight	the value of the test$weight argument (or its default value if missing).
provTuCo	the value of the test$tuCo argument (or its default value if missing).
provSemType	the value of the test$sem.type argument (or its default value if missing).
provSemExact	the value of the test$sem.exact argument (or its default value if missing).
se.ase	the value of the test$se.ase argument (or its default value if missing).
provRange	the value of the test$range argument (or its default value if missing).
provD	the value of the test$D argument (or its default value if missing)or NA if model is not NULL.

itemSelect	the value of the test$itemSelect argument (or its default value if missing).
infoType	the value of the test$infoType argument (or its default value if missing). Not returned if model is not NULL.
randomesque	the value of the test$randomesque argument (or its default value if missing).
AP	the value of the test$AP argument (or its default value if missing).
constantPattern	the value of the test$constantPatt argument (or its default value if missing).
cbControl	the value of the cbControl argument (or its default value if missing).
cbGroup	the value of the itemBank$cbGroup element of the item bank itemBank (for dichotomous IRT models), or the cbGroup element returned by the breakBank function (for polytomous IRT models), or NULL.
stopRule	the value of the stop$rule argument (or its default value if missing).
stopThr	the value of the stop$thr argument (or its default value if missing).
stopAlpha	the value of the stop$alpha argument (or its default value if missing).
endWarning	a logical indactor indicating whether the adaptive test stopped because the stopping rule(s) was (were) satisfied, or becasue all items in the bank were administered.
finalMethod	the value of the final$method argument (or its default value if missing).
finalDist	the value of the final$priorDist argument (or its default value if missing).
finalPar	the value of the final$priorPar argument (or its default value if missing).
finalWeight	the value of the final$weight argument (or its default value if missing).
finalTuCo	the value of the final$tuCo argument (or its default value if missing).
finalSemType	the value of the final$sem.type argument (or its default value if missing).
finalSemExact	the value of the final$sem.exact argument (or its default value if missing).
finalRange	the value of the final$range argument (or its default value if missing).
finalD	the value of the final$D argument (or its default value if missing), or NA if model is not NULL.
finalAlpha	the value of the final$alpha argument (or its default value if missing).
save.output	the value of the save.output argument.
output	the value of the output argument.
assigned.responses	a logical value, being TRUE if responses was provided or FALSE responses was set to NULL.

The function print.cat returns similar (but differently organized) results.

Note

When computing the exact standard errors, the number of test items should not be larger than 10 under the 2PL model or any more complex model. This constraint does not hold under the 1PL model.

Author(s)

David Magis
Department of Psychology, University of Liege, Belgium
<david.magis@uliege.be>

Juan Ramon Barrada
Department of Psychology and Sociology, Universidad Zaragoza, Spain
<barrada@unizar.es>

References

Barrada, J. R., Olea, J., Ponsoda, V., and Abad, F. J. (2010). A method for the comparison of item selection rules in computerized adaptive testing. *Applied Psychological Measurement*, *20*, 213-229. doi: 10.1177/0146621610370152

Barton, M.A., and Lord, F.M. (1981). *An upper asymptote for the three-parameter logistic item-response model*. Research Bulletin 81-20. Princeton, NJ: Educational Testing Service.

Birnbaum, A. (1969). Statistical theory for logistic mental test models with a prior distribution of ability. *Journal of Mathematical Psychology, 6*, 258-276. doi: 10.1016/00222496(69)900054

Bock, R. D., and Mislevy, R. J. (1982). Adaptive EAP estimation of ability in a microcomputer environment. *Applied Psychological Measurement, 6*, 431-444. doi: 10.1177/014662168200600405

Chang, H.-H., and Ying, Z. (1999). A-stratified multistage computerized adaptive testing. *Applied Psychological Measurement, 23*, 211-222. doi: 10.1177/01466219922031338

Haley, D.C. (1952). *Estimation of the dosage mortality relationship when the dose is subject to error*. Technical report no 15. Palo Alto, CA: Applied Mathematics and Statistics Laboratory, Stanford University.

Jeffreys, H. (1939). *Theory of probability*. Oxford, UK: Oxford University Press.

Jeffreys, H. (1946). An invariant form for the prior probability in estimation problems. *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences, 186*, 453-461.

Kingsbury, G. G., and Zara, A. R. (1989). Procedures for selecting items for computerized adaptive tests. *Applied Measurement in Education, 2*, 359-375. doi: 10.1207/s15324818ame0204_6

Lord, F. M. (1980). *Applications of item response theory to practical testing problems*. Hillsdale, NJ: Lawrence Erlbaum.

Magis, D. and Barrada, J. R. (2017). Computerized Adaptive Testing with R: Recent Updates of the Package *catR*. *Journal of Statistical Software*, *Code Snippets*, *76(1)*, 1-18. doi: 10.18637/jss.v076.c01

Magis, D., and Raiche, G. (2012). Random Generation of Response Patterns under Computerized Adaptive Testing with the R Package *catR*. *Journal of Statistical Software*, *48 (8)*, 1-31. doi: 10.18637/jss.v048.i08

Schuester, C., and Yuan, K.-H. (2011). Robust estimation of latent ability in item response models. *Journal of Educational and Behavioral Statistics, 36*, 720)735.doi: 10.3102/1076998610396890

Thompson, N. A. (2009). Item selection in computerized classification testing. *Educational and Psychological Measurement, 69*, 778-793. doi: 10.1177/0013164408324460

Urry, V. W. (1970). *A Monte Carlo investigation of logistic test models.* Unpublished doctoral dissertation. West Lafayette, IN: Purdue University.

van der Linden, W. J. (1998). Bayesian item selection criteria for adaptive testing. *Psychometrika, 63*, 201-216. doi: 10.1007/BF02294775

Veerkamp, W. J. J., and Berger, M. P. F. (1997). Some new item selection criteria for adaptive testing. *Journal of Educational and Behavioral Statistics, 22*, 203-226. doi: 10.3102/10769986022002203

Warm, T.A. (1989). Weighted likelihood estimation of ability in item response models. *Psychometrika, 54*, 427-450. doi: 10.1007/BF02294627

See Also

testList, startItems, nextItem, thetaEst, semTheta, eapEst, eapSem, MWI, MEI, KL, breakBank, genPolyMatrix, genPattern, aStratified

Examples

```
## Dichotomous models ##

 # Loading the 'tcals' parameters
 data(tcals)

 # Matrix of item parameters (with no content balancing)
 bank <- as.matrix(tcals[,1:4])

 # Creation of a starting list: 5 items with evenly spread theta values
 start <- list(theta = -2:2)

 # Creation of 'test' list: weighted likelihood estimation of provisional ability,
 # and MFI criterion for next item selection
 test <- list(method = "WL", itemSelect = "MFI")

 # Creation of 'final' list: EAP estimation of final ability
 final <- list(method = "EAP")

 # Creation of a stopping rule: precision criterion, standard error to be reached 0.3
 stop <- list(rule = "precision", thr = 0.3)

 # CAT test
 res <- randomCAT(0, bank, start = start, test = test, stop = stop, final = final)

## Not run:
 # Update of the stopping rule: by adding a length criterion, with threshold of 10 items
 stop <- list(rule = c("precision", "length"), thr = c(0.3, 10))

 # CAT test
 res <- randomCAT(0, bank, start = start, test = test, stop = stop, final = final)

 # Modifying 'start', 'test' and 'final' lists to compute exact SEs
 start.exact <- list(theta = c(-1, 1))
 test.exact <- list(method = "WL", itemSelect = "MFI", sem.exact = TRUE)
 final.exact <- list(method = "WL", sem.exact = TRUE)
 res.exact <- randomCAT(0, bank, start = start.exact, test = test.exact,
```

```
                    stop = stop, final = final.exact)

# Creation of a (purely artificial) response pattern for post-hoc simulation
resp <- rbinom(nrow(bank), 1, 0.5)
res.ph <- randomCAT(0, bank, start = start, test = test, stop = stop, final = final,
       responses = resp)

# New 'test' and 'final' rules (BM and EAP estimation with Jeffreys' prior,
# randomesque value 5)
test2 <- list(method = "BM", priorDist = "Jeffreys", randomesque = 5)
final2 <- list(method = "EAP", priorDist = "Jeffreys")

# New stopping rule: classification criterion, with classification threshold 0 and
# alpha level 0.05
stop2 <- list(rule = "classification", thr = 0, alpha = 0.05)

# CAT test with new 'test', 'stop' and 'final' rules
res2 <- randomCAT(0, bank, start = start, test = test2, stop = stop2, final = final2)

# New stopping rule: classification criterion, with classification threshold 0.5
# and alpha level 0.05
stop3 <- list(rule = "classification", thr = 0.5, alpha = 0.05)

# CAT test with new 'stop' rule
res3 <- randomCAT(0, bank, start = start, test = test2, stop = stop3, final=final2)

# new 'test' and 'stop' rule for next item selection
test3 <- list(method = "WL", itemSelect = "MLWI")
stop4 <- list(rule = "length",thr = 10)
res4 <- randomCAT(0, bank, start = start, test = test3, stop = stop4, final = final2)

# Creation of the 'cbList' list with arbitrary proportions
cbList <- list(names = c("Audio1", "Audio2", "Written1", "Written2",
       "Written3"), props = c(0.1, 0.2, 0.2, 0.2, 0.3))

# CAT test with 'start', 'test', 'stop4' and 'final2' lists and content balancing
# using 'cbList' ('tcals' must be used now for content balancing)
stop4 <- list(rule = "length",thr = 10)
res5 <- randomCAT(0, tcals, start = start, test = test, stop = stop4, final = final2,
     cbControl = cbList)

# new 'start' list to force content balancing control at the starting step
start2 <- list(seed = 1, nrItems = 3, cb.control = TRUE)
res6 <- randomCAT(0, tcals, start = start2, test = test, stop = stop4, final = final2,
     cbControl = cbList)

# Using progressive item selection rule and requiring all ability estimates and SEs
test4 <- list(itemSelect = "progressive")
res6 <- randomCAT(0, tcals, start = start, test = test4, stop = stop4, final = final,
     cbControl = cbList, allTheta = TRUE)

# Saving the output in the external 'out' text file within folder 'Program Files'
# of hard drive 'C'
```

```
res5 <- randomCAT(0, tcals, start = start, test = test, stop = stop4, final = final2,
     cbControl = cbList, save.output = TRUE,
     output = c("c:/Program Files/", "out", "txt"))

# Plotting results
plot(res)
plot(res, ci = TRUE)
plot(res, ci = TRUE, trueTh = FALSE)
plot(res, ci = TRUE, classThr = 1)

# Saving last figure into PDF file 'figure' within folder 'C:/Program Files/'
plot(res, ci = TRUE, classThr = 1, save.plot = TRUE,
     save.options = c("c:/Program Files/", "figure", "pdf"))

# With mistake
plot(res, ci = 0.05)
plot(res, classThr = TRUE)


## Polytomous models ##

# Generation of an item bank under GRM with 100 items and at most 4 categories
m.GRM <- genPolyMatrix(100, 4, "GRM")
m.GRM <- as.matrix(m.GRM)

# CAT options
start <- list(theta = c(-1, 0),startSelect = "MFI")
test <- list(method = "BM", itemSelect = "KL")
stop <- list(rule = "precision", thr = 0.4)
final <- list(method = "EAP")

# CAT test
res <- randomCAT(0, m.GRM,model = "GRM", start = start, test = test, stop = stop,
 final = final)
res


# Creation of an appropriate list for content balancing
# Equal proprotions across subgroups of items
cbList <- list(names = c("Group1", "Group2", "Group3", "Group4"), props = rep(1,4))

# With content balancing, all ability estimates and progressive method
m.GRM <- genPolyMatrix(100, 4, "GRM", cbControl = cbList)
test <- list(method = "BM", itemSelect = "progressive")
res <- randomCAT(0, m.GRM, model = "GRM", start = start, test = test, stop = stop,
 final = final, cbControl = cbList, allTheta = TRUE)
res

# Loading the cat_pav data
data(cat_pav)
cat_pav <- as.matrix(cat_pav)

stop <- list(rule = "length", thr = 10)
```

```
res <- randomCAT(0, cat_pav, model = "GPCM", start = start, test = test, stop = stop,
 final = final, allTheta = TRUE)
res

## End(Not run)
```

---

semTheta *Standard error of ability estimation (dichotomous and polytomous models)*

Description

This command returns the estimated standard error of the ability estimate, for a given response pattern and a given matrix of item parameters, either under the 4PL model or any suitable polytomous IRT model. Exact standard errors are available for dichotomous models.

Usage

```
semTheta(thEst, it, x = NULL, model = NULL, D = 1, method = "BM",
   priorDist = "norm", priorPar = c(0, 1), weight = "Huber", tuCo = 1,
   sem.type = "classic", parInt = c(-4, 4, 33), constantPatt = NULL,
   sem.exact = FALSE, trueTh = NULL, range = c(-4, 4))
```

Arguments

thEst	numeric: the ability estimate.
it	numeric: a suitable matrix of item parameters. See **Details**.
x	numeric: a vector of item responses (default is NULL). Can also hold missing data (coded as NAs). Ignored if method is not "EAP" but must be provided in presence of missing responses. See **Details**.
model	either NULL (default) for dichotomous models, or any suitable acronym for polytomous models. Possible values are "GRM", "MGRM", "PCM", "GPCM", "RSM" and "NRM". See **Details**.
D	numeric: the metric constant. Default is D=1 (for logistic metric); D=1.702 yields approximately the normal metric (Haley, 1952).
method	character: the ability estimator. Possible values are "BM" (default), "ML", "WL", "EAP" and "ROB". See **Details**.
priorDist	character: specifies the prior distribution. Possible values are "norm" (default), "unif" and "Jeffreys". Ignored if method is neither "BM" nor "EAP". See **Details**.
priorPar	numeric: vector of two components specifying the prior parameters (default is c(0,1)) of the prior ability distribution. Ignored if method is neither "BM" nor "EAP", or if priorDist="Jeffreys". See **Details**.

weight	character: the type of weight function for the robust estimator. Possible values are "Huber" (default) and "Tukey". Ignored if method is not "ROB" or if model is not NULL. See **Details**.
tuCo	numeric: the value of the tuning constant for the weight function (default is 1, suitable with "Huber" weight). Ignored if method is not "ROB" or if model is not NULL. See **Details**.
sem.type	character: the type of ASE formula to be used, either "classic" (default) or "new". Ignored if method is neither "BM" nor "WL", or if model is not NULL. See **Details**.
parInt	numeric: vector of three components, holding respectively the values of the arguments lower, upper and nqp of the eapEst command. Default vector is (-4, 4, 33). Ignored if method is not "EAP".
constantPatt	character: the method to estimate ability in case of constant pattern (i.e. only correct or only incorrect responses). Can be eitehr NULL (default), "BM", "EAP", "WL", "fixed4", "fixed7" or "var". *Currently only implemented for dichotomous IRT models*. See **Details**.
sem.exact	logical: should *exact* standard error be computed instead of asymptotic standard error? (default is FALSE). Ignored if model is not NULL. See **Details**.
trueTh	(For simulation study purposes only) either NULL (default) or the true ability level of interest. Ignored if sem.exact is FALSE or if model is not NULL. See **Details**.
range	numeric: vector of two components specifying the range wherein the ability estimate must be looked for (default is c(-4,4)). Used only to compute exact standard errors. Ignored if sem.exact is FALSE or if model is not NULL.

Details

Dichotomous IRT models are considered whenever model is set to NULL (default value). In this case, it must be a matrix with one row per item and four columns, with the values of the discrimination, the difficulty, the pseudo-guessing and the inattention parameters (in this order). These are the parameters of the four-parameter logistic (4PL) model (Barton and Lord, 1981).

Polytomous IRT models are specified by their respective acronym: "GRM" for Graded Response Model, "MGRM" for Modified Graded Response Model, "PCM" for Partical Credit Model, "GPCM" for Generalized Partial Credit Model, "RSM" for Rating Scale Model and "NRM" for Nominal Response Model. The it still holds one row per item, end the number of columns and their content depends on the model. See genPolyMatrix for further information and illustrative examples of suitable polytomous item banks.

The vector of response patterns x can also hold missing responses (more useful in linear testing, not in CAT). In this case the missing responses must be coded as NA values. They are discarded from the ability estimation process. Note that in presence of missing responses, the pattern x must be provided indepedently of the estimation method (to discard items with missing responses from the computation).

Five ability estimators are available: the maximum likelihood (ML) estimator (Lord, 1980), the Bayes modal (BM) estimator (Birnbaum, 1969), the expected a posteriori (EAP) estimator (Bock and Mislevy, 1982), the weighted likelihood (WL) estimator (Warm, 1989) and the robust estimator

(Schuster & Yuan, 2011). The selected estimator is specified by the `method` argument, with values `"ML"`, `"BM"`, `"EAP"`, `"WL"` and `"ROB"` respectively.

For the BM and EAP estimators, three prior distributions are available: the normal distribution, the uniform distribution and the Jeffreys' prior distribution (Jeffreys, 1939, 1946). The prior distribution is specified by the argument `priorPar`, with values `"norm"`, `"unif"` and `"Jeffreys"`, respectively. The `priorPar` argument is ignored if `method="ML"` or `method="WL"`.

The argument `priorPar` determines either: the prior mean and standard deviation of the normal prior distribution (if `priorDist="norm"`), or the range for defining the prior uniform distribution (if `priorDist="unif"`). This argument is ignored if `priorDist="Jeffreys"`.

The `eapPar` argument sets the range and the number of quadrature points for numerical integration in the EAP process. By default, it takes the vector value (-4, 4, 33), that is, 33 quadrature points on the range [-4; 4] (or, by steps of 0.25). See `eapEst` for further details.

Robust estimation requires an appropriate weight function that depends on an accurate tuning constant. Suggested functions are the Huber weight (Schuester and Yuan, 2011) and the Tukey weight (Mosteller and Tukey, 1977). Both can be set by the `weight` argument, with respective values `"Huber"` and `"Tukey"`. Default function is Huber. Moreover, the `tuCo` argument specifies the tuning constant for the weight function. Default value is 1 and suggested for Huber weight (also by default), and value 4 is suggested for Tukey weight (Schuester and Yuan, 2011).

New ASE formulas, proposed by Magis (2016), are now available. They can be supplied by the `sem.type` argument, which takes the default value `"classic"` (so usual ASEs are computed) or `"new"` (for the newly suggested formulas). Note that new ASEs are available only for BM and WL estimators, as for other estimators the classic and new versions are identical. Note also that these new ASE formulas are available for dichotomous IRT models only.

Note that in the current version, the ability estimate must be specified through the `thEst` argument. Moreover, the response pattern must be specified through the `x` argument to compute the standard error of the EAP estimate. For the other estimation methods, this is not necessary, and `x` is set to `NULL` by default for this purpose.

Note also that if specific stepsize adjustment was required for constant patterns with the `constantPatt` argument (that is, if it takes value `"fixed4"`, `"fixed7"` or `"var"`) then an infinite value `Inf` is being returned.

Finally, exact standard errors can be computed with dichotomous IRT models only (Magis, 2014). This is requested by setting argument `sem.exact` to `TRUE` (default is `FALSE` so regular, asymptotic formula is considered). The exact standard error is computed using the full ability distribution provided by the `link{fullDist}` function. Two additional arguments can be set up: `range` to define the range of estimated ability levels during the derivation of the full distribution, and `trueTh` that can be used to specify the true ability level in addition to the estimated one (through `thEst` argument). The latter argument is for simulation study purposes, as the set of all observable ability estimates in `fullDist` must then be computed only once. If `trueTh` is provided some real value, then two SEs are returned: the *exact* SE obtained with the estimated `thEst` level, and the *true* SE derived as the exact SE with true `trueTh` level.

Important: The computation of the exact standard error is very efficient under the Rasch (1PL) model due to the ue of the Lord-Wingersky algorithm (Lord and Wingersky, 1984). Therefore, any test length can be considered for computing the exact SE under this model. But for other dichotomous IRT models, the computatinal effort becomes very demanding even with a limited set of n items, since 2^n patterns must be generated and ability estimated with each such pattern. For

this reason, it is recommended not to consider exact SE with models other than the Rasch (1PL) with more than 10 items.

Value

The estimated standard error of the ability level (or `Inf` if the response pattern is constant and `constantPatt` is not `NULL`).

If exact standard error is computed (`sem.exact = TRUE`) and `trueTh` argument takes some numeric value, then output is a vector of two components with the two exact standard errors, the first SE corresponding to the `thEst` value and the second SE to the `trueTh` value.

Note

The classic asymptotic standard error of the WL estimator is computed with the same formula as that of the ML estimator (up to the plug-in of the WL estimate instead of the ML estimate). Note however that versions of `catR` prior to 3.0 hold a different formula mentioned in Magis and raiche (2012), but it appeared that this formula can lead to negative values of the square of the standard error. So the usual suggestion by Warm (1989) of using the same asymptotic formulas for ML and WL is the currently used formula for classic asymptotic computation of the standard error.

Author(s)

David Magis
Department of Psychology, University of Liege, Belgium
<david.magis@uliege.be>

Lianne Ippel
Department of Psychology, University of Liege, Belgium
<g.j.e.ippel@gmail.com>

References

Barton, M.A., and Lord, F.M. (1981). *An upper asymptote for the three-parameter logistic item-response model*. Research Bulletin 81-20. Princeton, NJ: Educational Testing Service.

Birnbaum, A. (1969). Statistical theory for logistic mental test models with a prior distribution of ability. *Journal of Mathematical Psychology, 6*, 258-276. doi: 10.1016/00222496(69)900054

Bock, R. D., and Mislevy, R. J. (1982). Adaptive EAP estimation of ability in a microcomputer environment. *Applied Psychological Measurement, 6*, 431-444. doi: 10.1177/014662168200600405

Dodd, B. G., De Ayala, R. J., and Koch, W. R. (1995). Computerized adaptive testing with polytomous items. *Applied Psychological Measurement, 19*, 5-22. doi: 10.1177/014662169501900103

Haley, D.C. (1952). *Estimation of the dosage mortality relationship when the dose is subject to error*. Technical report no 15. Palo Alto, CA: Applied Mathematics and Statistics Laboratory, Stanford University.

Jeffreys, H. (1939). *Theory of probability*. Oxford, UK: Oxford University Press.

Jeffreys, H. (1946). An invariant form for the prior probability in estimation problems. *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences, 186*, 453-461.

Lord, F.M. (1980). *Applications of item response theory to practical testing problems*. Hillsdale, NJ: Lawrence Erlbaum.

Lord, F. M., and Wingersky, M. S. (1984). Comparison of IRT true-score and equipercentile observed-score equatings. *Applied Psychological Measurement, 8*, 453-461. doi: 10.1177/014662168400800409

Magis, D. (2014). Accuracy of asymptotic standard errors of the maximum and weighted likelihood estimators of proficiency levels with short tests. *Applied Psychological Measurement, 38*, 105-121. doi: 10.1177/0146621613496890

Magis, D. (2016). Efficient standard errors formulas of ability estimators with dichotomous item response models. *Psychometrika, 81*, 184-200. doi: 10.1007/s1133601594433

Magis, D. and Barrada, J. R. (2017). Computerized Adaptive Testing with R: Recent Updates of the Package *catR*. *Journal of Statistical Software*, *Code Snippets*, *76(1)*, 1-18. doi: 10.18637/jss.v076.c01

Magis, D., and Raiche, G. (2012). Random Generation of Response Patterns under Computerized Adaptive Testing with the R Package *catR*. *Journal of Statistical Software*, *48 (8)*, 1-31. doi: 10.18637/jss.v048.i08

Mosteller, F., and Tukey, J. (1977). *Exploratory data analysis and regression*. Reading, MA: Addison-Wesley.

Schuester, C., and Yuan, K.-H. (2011). Robust estimation of latent ability in item response models. *Journal of Educational and Behavioral Statistics, 36*, 720)735.doi: 10.3102/1076998610396890

Warm, T.A. (1989). Weighted likelihood estimation of ability in item response models. *Psychometrika, 54*, 427-450. doi: 10.1007/BF02294627

See Also

eapSem, thetaEst, genPolyMatrix, fullDist

Examples

```
## Dichotomous models ##

 # Loading the 'tcals' parameters
 data(tcals)

 # Selecting item parameters only
 tcals <- as.matrix(tcals[,1:4])

 # Creation of a response pattern (tcals item parameters, true ability level 0)
 x <- genPattern(0, tcals, seed = 1)

 # ML estimation
 th <- thetaEst(tcals, x, method = "ML")
 c(th, semTheta(th, tcals, method = "ML"))

 # ML estimation, new ASE formula (=> yields the same result)
 c(th, semTheta(th, tcals, method = "ML", sem.type = "new"))

 # With first two responses missing
 x.mis <- x
 x.mis[1:2] <- NA
 th <- thetaEst(tcals, x.mis, method = "ML")
```

```
c(th, semTheta(th, tcals, x.mis, method = "ML"))

# BM estimation, standard normal prior distribution
th <- thetaEst(tcals, x)
c(th, semTheta(th, tcals))

# BM estimation and new ASE formula
c(th, semTheta(th, tcals, sem.type = "new"))

# BM estimation, uniform prior distribution upon range [-2,2]
th <- thetaEst(tcals, x, method = "BM", priorDist = "unif",
               priorPar = c(-2, 2))
c(th, semTheta(th, tcals, method = "BM", priorDist = "unif",
   priorPar = c(-2, 2)))

# BM estimation, Jeffreys' prior distribution
th <- thetaEst(tcals, x, method = "BM", priorDist = "Jeffreys")
c(th, semTheta(th, tcals, method = "BM", priorDist = "Jeffreys"))

# EAP estimation, standard normal prior distribution
th <- thetaEst(tcals, x, method = "EAP")
c(th, semTheta(th, tcals, x, method = "EAP"))

## Not run:

# EAP estimation, uniform prior distribution upon range [-2,2]
th <- thetaEst(tcals, x, method = "EAP", priorDist = "unif",
               priorPar = c(-2, 2))
c(th, semTheta(th, tcals, x, method = "EAP", priorDist = "unif",
   priorPar = c(-2, 2)))

# EAP estimation, Jeffreys' prior distribution
th <- thetaEst(tcals, x, method = "EAP", priorDist = "Jeffreys")
c(th, semTheta(th, tcals, x, method = "EAP", priorDist = "Jeffreys"))

# WL estimation
th <- thetaEst(tcals, x, method = "WL")
c(th, semTheta(th, tcals, method = "WL"))

# WL estimation, new ASE formula
c(th, semTheta(th, tcals, method = "WL", sem.type = "new"))

# 'fixed4' adjustment for constant pattern
th <- thetaEst(tcals, rep(0, nrow(tcals)), constantPatt = "fixed4")
c(th, semTheta(th, tcals, constantPatt = "fixed4"))

# Robust estimation
th <- thetaEst(tcals, x, method = "ROB")
c(th, semTheta(th, tcals, method = "ROB"))

# Robust estimation, Huber weight and tuning constant 2
th <- thetaEst(tcals, x, method = "ROB", tuCo = 2)
c(th, semTheta(th, tcals, method = "ROB", tuCo = 2))
```

```
 # Robust estimation, Tukey weight and tuning constant 4
 th <- thetaEst(tcals, x, method = "ROB", weight = "Tukey", tuCo = 4)
 c(th, semTheta(th, tcals, method = "ROB", weight = "Tukey", tuCo = 4))

 ## Exact SE computation under 1PL model:
 # Creation of a 1PL item bank with difficulties from 'tcals' (85 items)
 tcals2 <- cbind(1, tcals[, 2],  0, 1)

 # Pattern generation for true ability level 1
 x2 <- genPattern(1, tcals2, seed = 1)

 # ML estimation
 th2 <- thetaEst(tcals2, x2, method = "ML")
 c(th2, semTheta(th2, tcals2, x2, method = "ML", sem.exact = TRUE))

 # ML estimation, true SE in addition
 c(th2, semTheta(th2, tcals2, x2, method = "ML", sem.exact = TRUE, trueTh = 1))

 ## Exact SE computation under 2PL model:
 # Creation of a 2PL item bank with ten items
 it <- genDichoMatrix(10, model = "2PL", seed = 1)

 # Pattern generation for true ability level 1
 x3 <- genPattern(1, it, seed = 1)

 # ML estimation
 th3 <- thetaEst(it, x3, method = "ML")
 c(th3, semTheta(th3, it, x3, method = "ML", sem.exact = TRUE))

 # ML estimation, true SE in addition
 c(th3, semTheta(th3, it, x3, method = "ML", sem.exact = TRUE, trueTh = 1))

## End(Not run)

## Polytomous models ##

 # Generation of an item bank under GRM with 100 items and at most 4 categories
 m.GRM <- genPolyMatrix(100, 4, "GRM")
 m.GRM <- as.matrix(m.GRM)

 # Creation of a response pattern (true ability level 0)
 set.seed(1)
 x <- genPattern(0, m.GRM, model = "GRM")

# ML estimation
 th <- thetaEst(m.GRM, x, model = "GRM", method = "ML")
 c(th, semTheta(th, m.GRM, model = "GRM", method = "ML"))

 # BM estimation, standard normal prior distribution
 th <- thetaEst(m.GRM, x, model = "GRM")
 c(th, semTheta(th, m.GRM, model = "GRM"))
```

```
# BM estimation, uniform prior distribution upon range [-2,2]
th <- thetaEst(m.GRM, x, model = "GRM", method = "BM", priorDist = "unif",
   priorPar = c(-2, 2))
c(th, semTheta(th, m.GRM, model = "GRM", method = "BM", priorDist = "unif",
 priorPar = c(-2, 2)))

# BM estimation, Jeffreys' prior distribution
th <- thetaEst(m.GRM, x, model = "GRM", method = "BM", priorDist = "Jeffreys")
c(th, semTheta(th, m.GRM, model = "GRM", method = "BM", priorDist = "Jeffreys"))

# EAP estimation, standard normal prior distribution
th <- thetaEst(m.GRM, x, model = "GRM", method = "EAP")
c(th, semTheta(th, m.GRM, x, model = "GRM", method = "EAP") )

## Not run:

# EAP estimation, uniform prior distribution upon range [-2,2]
th <- thetaEst(m.GRM, x, model = "GRM", method = "EAP", priorDist = "unif",
   priorPar = c(-2, 2))
c(th, semTheta(th, m.GRM, x, model = "GRM", method = "EAP", priorDist = "unif",
 priorPar = c(-2, 2)))

# EAP estimation, Jeffreys' prior distribution
th <- thetaEst(m.GRM, x, model = "GRM", method = "EAP", priorDist = "Jeffreys")
c(th, semTheta(th, m.GRM, x, model = "GRM", method = "EAP", priorDist = "Jeffreys"))

# WL estimation
th <- thetaEst(m.GRM, x, model = "GRM", method = "WL")
c(th, semTheta(th, m.GRM, model = "GRM", method = "WL"))


# Loading the cat_pav data
data(cat_pav)
cat_pav <- as.matrix(cat_pav)

# Creation of a response pattern (true ability level 0)
set.seed(1)
x <- genPattern(0, cat_pav, model = "GPCM")

# ML estimation
th <- thetaEst(cat_pav, x, model = "GPCM", method = "ML")
c(th, semTheta(th, cat_pav, model = "GPCM", method = "ML"))

# BM estimation, standard normal prior distribution
th <- thetaEst(cat_pav, x, model = "GPCM")
c(th, semTheta(th, cat_pav, model = "GPCM"))

# BM estimation, uniform prior distribution upon range [-2,2]
th <- thetaEst(cat_pav, x, model = "GPCM", method = "BM", priorDist = "unif",
   priorPar = c(-2, 2))
c(th, semTheta(th, cat_pav, model = "GPCM", method = "BM", priorDist = "unif",
 priorPar = c(-2, 2)))
```

```
# BM estimation, Jeffreys' prior distribution
th <- thetaEst(cat_pav, x, model = "GPCM", method = "BM", priorDist = "Jeffreys")
c(th, semTheta(th, cat_pav, model = "GPCM", method = "BM", priorDist = "Jeffreys"))

# EAP estimation, standard normal prior distribution
th <- thetaEst(cat_pav, x, model = "GPCM", method = "EAP")
c(th, semTheta(th, cat_pav, x, model = "GPCM", method = "EAP"))

# EAP estimation, uniform prior distribution upon range [-2,2]
th <- thetaEst(cat_pav, x, model = "GPCM", method = "EAP", priorDist = "unif",
   priorPar = c(-2, 2))
c(th, semTheta(th, cat_pav, x, model = "GPCM", method = "EAP", priorDist = "unif",
 priorPar = c(-2, 2)))

# EAP estimation, Jeffreys' prior distribution
th <- thetaEst(cat_pav, x, model = "GPCM", method = "EAP", priorDist = "Jeffreys")
c(th, semTheta(th, cat_pav, x, model = "GPCM", method = "EAP", priorDist = "Jeffreys"))

# WL estimation
th <- thetaEst(cat_pav, x, model = "GPCM", method = "WL")
c(th, semTheta(th, cat_pav, model = "GPCM", method = "WL"))

## End(Not run)
```

`simulateRespondents` *Simulation of multiple examinees of adaptive tests*

Description

This command runs a set of adaptive tests, for a given item bank, a set of ability levels, a possible matrix of item responses, and several lists of CAT parameters (starting items, stopping rule, provisional and final ability estimators).

Usage

```
simulateRespondents(thetas, itemBank, responsesMatrix = NULL, model = NULL,
   genSeed = NULL, cbControl = NULL, rmax = 1, Mrmax = "restricted",
   start = list(fixItems = NULL, seed = NULL, nrItems = 1, theta = 0,
   D = 1, randomesque = 1, random.seed = NULL, startSelect = "MFI",
   cb.control = FALSE, random.cb = NULL), test = list(method = "BM",
   priorDist = "norm", priorPar = c(0,1), weight = "Huber", tuCo = 1,
   sem.type = "classic", sem.exact = FALSE, se.ase = 10, range = c(-4, 4),
   D = 1, parInt = c(-4, 4, 33), itemSelect = "MFI", infoType = "observed",
   randomesque = 1, random.seed = NULL, AP = 1, proRule = "length",
   proThr = 20, constantPatt = NULL), stop = list(rule = "length",
   thr = 20, alpha = 0.05), final = list(method = "BM", priorDist = "norm",
   priorPar = c(0,1), weight = "Huber", tuCo = 1, sem.type = "classic",
   sem.exact = FALSE, range = c(-4, 4), D = 1, parInt = c(-4, 4, 33),
```

```
   alpha = 0.05), save.output = FALSE, output = c("", "catR", "csv"))
## S3 method for class 'catResult'
print(x, ...)
## S3 method for class 'catResult'
plot(x, type = "all", deciles = "theta", save.plot = FALSE,
   save.options = c("", "plot", "pdf"), res = 300, ...)
```

Arguments

thetas	numeric: a vector of true ability values for which a CAT must be generated for each component.
itemBank	numeric: a suitable matrix of item parameters (possibly augmented by group membership for content balancing). See **Details**.
responsesMatrix	numeric: either NULL (default) or a suitable matrix of item responses. See **Details**.
model	either NULL (default) for dichotomous models, or any suitable acronym for polytomous models. Possible values are "GRM", "MGRM", "PCM", "GPCM", "RSM" and "NRM". See **Details**.
genSeed	either a vector of numeric values to fix the random seed of each generated pattern, or NULL (default). Ignored if responsesMatrix is not NULL. See **Details**.
cbControl	either a list of accurate format to control for content balancing, or NULL. See **Details**.
rmax	numeric: the maximum exposure rate (default is 1).
Mrmax	character: the method for controlling maximum exposure rate. Possible values are "restricted"(default) or "IE". See **Details**.
start	a list with the options for starting the adaptive test. See **Details**.
test	a list with the options for provisional ability estimation and next item selection. See **Details**.
stop	a list with the options of the stopping rule. See **Details**.
final	a list with the options for final ability estimation. See **Details**.
save.output	logical: should the output be saved in an external text file? (default is FALSE).
output	character: a vector of three components. The first component is either the file path to save the output or "" (default), the second component is either the initial part of the name of the output file or "catR", and the third component is the file type, either "txt" or "csv" (default). See **Details**.
x	an object of class "cat", typically an output of simulateRespondents function.
type	character: the type of plot to display. Possible values are "all" (default), "trueEst", "expRate", "cumExpRate", "cumNumberItems", "expRatePara", "condBias", "condRMSE", "numberItems", "sError" and ="condThr". See **Details**.
deciles	whether the deciles number ("deciles") or the mean ability level per decile ("theta") will be used in the axis of the plots.

save.plot	logical: should the plot be saved in an external figure? (default is FALSE).
save.options	character: a vector of three components. The first component is either the file path or "" (default), the second component is the name of the output file or ,"plot" (default), and the third component is the file extension, either "pdf" (default) or "jpeg". Ignored if save.plot is FALSE. See **Details**.
res	numeric: the resolution for JPEG figures (default value is 300).
...	other generic arguments to be passed to print and plot functions.

Details

The simulateRespondents function permits to generate several adaptive tests to a set of respondents defined by their ability levels. It makes a repeated call to an adaptive test using an item bank specified by arguments itemBank model, and with the same start, test, stop and final lists. Content balancing can also be controlled for each respondent with the cbControl argument. All arguments of simulateRespondents are used in exactly the same manner as in randomCAT (so refer to this function for further information), except the following four.

First, thetas is now a vector of ability levels, and a CAT will be generated for each component of thetas. If responsesMatrix is NULL, item responses are generated from the IRT model, item bank parameters, and the ability levels. In this case, thetas can be considered as real ability levels. Otherwise, responsesMatrix must be provided as a matrix with as many rows as the length of thetas and as many columns as the number of items in itemBank. Each row contains the response pattern of one examinee whose ability level is given by the value of the corresponding component of thetas. Note that only allowable item responses can be included in responsesMatrix and missing values are not accepted. Fixing the random seed can also be done with the genSeed argument. The latter must hold as many components as the vector thetas, otherwise an error message is returned. Each component of genSeed is used to fix the seed for ach pattern generation.

The option of providing a response matrix through responsesMatrix is considered for two possible uses:

1. *post-hoc simulations*: examinees provided responses to the full item bank and one wants to test the performance of a CAT with those responses,
2. simulations considering *item parameter estimation errors*: responses to the full bank are generated with the real parameters and the CAT is run is the estimated parameters and the responses from the correct model.

Note that if thetas holds a single value, then the function simply calls randomCAT and returns its output instead.

Second, rmax fixes the desired maximum exposure rate for all items. Default value is 1, allowing thus items to be administered to all respondents without restrictions.

Third, the Mrmax argument fixes the method to constraint exposure rates to be smaller than the maximum allowed rate. Possible methods are the restricted method ("restricted"; Revuelta and Ponsoda, 1998) and the item-eligibility method ("IE"; van der Linden and Veldkamp, 2004). A description of both methods can be found in Barrada, Abad and Veldkamp (2009)

Fourth, if the length of thetas is greater than 1, save.output for randomCAT is fixed to "FALSE". Otherwise, the same file would be overwritten for each new respondent.

The output of simulateRespondents, as displayed by the print.catResult function, proposes summary statistics related to overall accuracy (bias, RMSE, etc), conditional accuracy per decile,

and item exposure control (minimum and maximum exposure rates, test overlap rate, etc) among all generated CATs. This output can be saved when `save.output` is set to `TRUE`, and if so, three outpout files are returned: (1) one with the main summary statistics, with overall results and conditional on decile results; (2) the file with respondents' patterns, items administered and provisional ability estimates; and (3) a table with true and estimated ability levels, final standard errors and numbers of items administered per respondent. Specific information provided and saved depends on the `rule` used in `stop`.

This output can also be graphically displayed with the `plot.catResult` function. In addition to the output of the function and the item bank, it takes the argument `type` to determine which plot should be returned. Ten different single plots can be displayed:

1. `"trueEst"`: the scatterplot of true vs. estimated ability levels.
2. `"expRate"`: the exposure rates of the items, having ranked them according to these exposure rates.
3. `"cumExpRate"`: the cumulative exposure rates of the items, having ranked them according to these exposure rates.
4. `"cumNumberItems"`: the test length as a function of cumulative percent of examinees. This plot is not available when `rule` is 'length'.
5. `"expRatePara"`: the scatterplot of item exposure rates vs. item discrimination parameters. This plot is not available when `model` is 'PCM' or 'NRM', as in those IRT models there are no discrimination parameters.
6. `"condBias"`: the conditional bias of ability estimation as a function of the deciles of the true ability levels.
7. `"condRMSE"`: the conditional RMSE of ability estimation as a function of the deciles of the true ability levels.
8. `"numberItems"`: the conditional test length as a function of the deciles of the true ability levels. This plot is not available when `rule` is 'length'.
9. `"sError"`: the conditional standard error of ability estimation as a function of the deciles of the true ability levels.
10. `"condThr"`: the conditional proportions of CATs satisfying the 'precision' or 'classification' stopping rule, as a function of the deciles of the true ability levels. This plot is not available when `rule` is 'length'.

In addition, the value `"all"` (default value) displays several available plots in a single panel. Displayed plots depend on the `rule` used in `stop`: (a) with 'length', `"trueEst"`, `"condBias"`, `"condRMSE"`, `"expRate"`, `"cumExpRate"`, `"expRatePara"`; (b) with 'precision' and 'classification', all the plots but `"cumExpRate"`.

These plots can be saved as external PDF or JPEG files, by setting `save.plot` to `TRUE` and defining the arguments of `save.options` accurately (see `randomCAT` for further explanations and the **Examples** section below).

Value

The function `simulateRespondents` returns a list of class "catResult" with the following arguments:

`thetas` the value of the `thetas` argument.

itemBank	the value of the itemBank argument.
responsesMatrix	the value of the responsesMatrix argument.
model	the value of the model argument.
genSeed	the value of the genSeed argument.
cbControl	the value of the cbControl argument.
rmax	the value of the rmax argument.
Mrmax	the value of the Mrmax argument.
start	the value of the start argument.
test	the value of the test argument.
stop	the value of the stop argument.
final	the value of the final argument.
save.output	the value of the save.output argument.
output	the value of the output argument.
estimatedThetas	a vector with (final) estimated ability levels.
correlation	the correlation between the thetas vector and estimated ability levels.
bias	the value of the bias between true and estimated ability levels.
RMSE	the value of the RMSE between true and estimated ability levels.
thrOK	a vector indicating whether the respondents finished the test satisfying (1) or not (0) the stop criteria.
exposureRates	a vector with empirical exposure rates of all item in the bank.
testLength	the mean test length.
overlap	the item overlap rate.
numberItems	a vector with the lengths of each adaptive test (i.e. the number of items administered).
condTheta	a vector with the mean ability level per decile.
condBias	a vector with conditional mean bias per decile.
condRMSE	a vector with conditional RMSE per decile.
condnItems	a vector with conditional mean test length per decile.
condSE	a vector with conditional mean standard error per decile.
condthrOK	a vector with conditional proportion of respondents that finish the test satisfying the stop criteria per decile.
ndecile	a vector with the number of respondents per decile.
final.values.df	a data frame with true ability levels, final ability estimates and standard errors, and test lengths.
responses.df	a data frame with all items administered, all item responses and all provisional ability estimates. -99 is displayed when the actual test length is smaller than the maximum test length for those item positions where no item was administered as the stop criterion was already reached.
start.time	the CPU time at the start of the CAT generation.
finish.time	the CPU time at the end of all CAT generations.

Author(s)

Juan Ramon Barrada
Department of Psychology and Sociology, Universidad Zaragoza, Spain
<barrada@unizar.es>

David Magis
Department of Psychology, University of Liege, Belgium
<david.magis@uliege.be>

Guido Corradi
Department of Psychology and Sociology, Universidad Zaragoza, Spain
<guidocor@gmail.com>

References

Barrada, J. R., Abad, F. J., and Veldkamp, B. P. (2009). Comparison of methods for controlling maximum exposure rates in computerized adaptive testing. *Psicothema*, *21*, 313-320.

Magis, D. and Barrada, J. R. (2017). Computerized Adaptive Testing with R: Recent Updates of the Package *catR*. *Journal of Statistical Software*, *Code Snippets*, *76(1)*, 1-18. doi: 10.18637/jss.v076.c01

Magis, D., and Raiche, G. (2012). Random Generation of Response Patterns under Computerized Adaptive Testing with the R Package *catR*. *Journal of Statistical Software*, *48 (8)*, 1-31. doi: 10.18637/jss.v048.i08

Revuelta, J., and Ponsoda, V. (1998). A comparison of item exposure control methods in computerized adaptive testing. *Journal of Educational Measurement, 35*, 311-327. doi: 10.1111/j.1745-3984.1998.tb00541.x

van der Linden, W.J., and Veldkamp, B.P. (2004). Constraining item exposure in computerized adaptive testing with shadow tests. *Journal of Educational and Behavioral Statistics, 29*, 273-291. doi: 10.3102/10769986029003273

See Also

randomCAT

Examples

```
## Dichotomous IRT model ##

 # Loading the 'tcals' parameters
 data(tcals)
 bank <- as.matrix(tcals[,1:4])

 # Creation of a starting list with three theta values
 start <- list(theta = -1:1, randomesque = 5)

 # Creation of 'test' list: maximum likelihood estimation and
 # progressive method
 test <- list(method = "ML", itemSelect = "progressive")
```

```
# Creation of a stopping rule: precision criterion, standard
# error to be reached 0.3
stop <- list(rule = "precision", thr = 0.3)

# Creation of 'final' list: ML estimation of final ability
final <- list(method = "ML")

# Generation of ten respondents
set.seed(1)
thetas <- rnorm(10)

# Default CAT generations, output not saved
res <- simulateRespondents(thetas, bank, start = start, test = test, stop = stop,
 final = final)

# Maximum exposure restricted to 0.8
res2 <- simulateRespondents(thetas, bank, start = start, test = test, stop = stop,
 final = final, rmax = 0.8)

## Not run:

# Output saved
res3 <- simulateRespondents(thetas, bank, start = start, test = test, stop = stop,
 final = final, save.output = TRUE, output = c("C:/Program Files/", "out", "txt"))

## End(Not run)

# With content balancing #

# Creation of an appropriate list for content balancing
# Equal proportions across subgroups of items
cbList <- list(names = c("Audio1", "Audio2", "Written1", "Written2", "Written3"),
       props = c(0.1, 0.2, 0.2, 0.2, 0.3))

# CAT test (same options as above)
res4 <- simulateRespondents(thetas, tcals, start = start, test = test, stop = stop,
 final = final, cbControl = cbList)

# Plotting and saving output #

# Plotting all possible panels
plot(res)
plot(res, deciles = "deciles")

## Not run:

# Saving the plot in the "fig" pdf file in "c:/Program Files/"
plot(res, save.plot = TRUE, save.options = c("c:/Program Files/", "fig", "pdf"))

## End(Not run)

# Plotting the 'trueEst' type of plot
plot(res, type = "trueEst")
```

`startItems` *Selection of the first items*

Description

This command selects the first items of the adaptive test, either randomly or on the basis of their information function.

Usage

```
startItems(itemBank, model = NULL, fixItems = NULL, seed = NULL, nrItems = 1,
   theta = 0, D = 1, randomesque = 1, random.seed = NULL, startSelect = "MFI",
   nAvailable = NULL, cbControl = NULL, cbGroup = NULL, random.cb=NULL)
```

Arguments

`itemBank`	numeric: a suitable matrix of item parameters. See **Details**.
`model`	either `NULL` (default) for dichotomous models, or any suitable acronym for polytomous models. Possible values are `"GRM"`, `"MGRM"`, `"PCM"`, `"GPCM"`, `"RSM"` and `"NRM"`. See **Details**.
`fixItems`	either a vector of integer values or `NULL` (default). See **Details**.
`seed`	either a numeric value, `NA` or `NULL` (default). Ignored if `fixItems` is not `NULL`. See **Details**.
`nrItems`	numeric: the number of starting items to be randomly selected (default is 1). Can be equal to zero to avoid initial selection of items (see **Details**). Used only if `fixItems` is `NULL` and `seed` is not `NULL`.
`theta`	numeric: a vector of the initial ability levels for selecting the first items (default is the single value 0). Ignored if either `fixItems` or `seed` is not `NULL`. See **Details**.
`D`	numeric: the metric constant. Default is `D=1` (for logistic metric); `D=1.702` yields approximately the normal metric (Haley, 1952). Ignored if `model` is not `NULL` and if `startSelect` is not `"MFI"`.
`randomesque`	integer: the number of 'randomesque' items to be picked up optimally for each value of `theta` vector, before random selection of a single one. Ignored if either `fixItems` or `seed` is not `NULL`. See **Details**.
`random.seed`	either `NULL` (default) or a numeric value to fix the random seed of randomesque selection of the items. Ignored if either `fixItems` or `seed` is not `NULL`.
`startSelect`	character: the criterion for selecting the first items. Possible values are `"bOpt"`, `"thOpt"`, `"progressive"`, `"proportional"`, and `"MFI"` (default). See **Details**.
`nAvailable`	either a boolean vector indicating which items (denoted by 1's) are available at the start of the test and which (denoted by 0's) are not, or `NULL` (default). See **Details**.

`cbControl` either a list of accurate format to control for content balancing, or `NULL`. See **Details**.

`cbGroup` either a factor vector of accurate format to control for content balancing, or `NULL`. See **Details**.

`random.cb` either `NULL` (default) or a numeric value to fix the selection of subgroups of items for random sampling the starting items. See **Details**.

Details

This function permits to select the first item(s) of the test. It works with both dichotomous and polytomous item banks.

Dichotomous IRT models are considered whenever `model` is set to `NULL` (default value). In this case, `it` must be a matrix with one row per item and four columns, with the values of the discrimination, the difficulty, the pseudo-guessing and the inattention parameters (in this order). These are the parameters of the four-parameter logistic (4PL) model (Barton and Lord, 1981).

Polytomous IRT models are specified by their respective acronym: `"GRM"` for Graded Response Model, `"MGRM"` for Modified Graded Response Model, `"PCM"` for Partical Credit Model, `"GPCM"` for Generalized Partial Credit Model, `"RSM"` for Rating Scale Model and `"NRM"` for Nominal Response Model. The `it` still holds one row per item, end the number of columns and their content depends on the model. See `genPolyMatrix` for further information and illustrative examples of suitable polytomous item banks.

The number of starting items is given by the length of `fixItems` argument, the `nrItems` argument (in case of random selection) or by the length of `theta` argument (in case of optimal selection), with default value 1 in all cases. It can be set to zero; in this case, only `NULL` values are returned in the output list and the CAT process will start without starting items.

The first item(s) of the adaptive test can be selected by one of the following methods.

1. By specifying the item(s) to be administered. The argument `fixItems` then holds the item number(s) as listed in the item bank. Setting `fixItems` to `NULL` (default value) disables this method.
2. By selecting it (them) randomly into the item bank. The argument `seed` permits to fix the random selection by specifying the random seed number and the number of selected items is fixed by the `nrItems` argument. Setting `seed` to `NA` disables the random seed (though items are still picked up randomly in the bank); in other words, successive runs of `startItems` with `seed=NA` may lead to different item(s) selection. Setting `seed` to `NULL` (default value) disables this selection method.
3. By selecting the item(s) according to an initial sequence of ability values set by the `theta` argument. In this case, five criteria can be used, specified through the `startSelect` argument:
   (a) `"MFI"` (default): one selects the most informative item(s) for the given initial ability value(s);
   (b) `"bOpt"`: one selects the item(s) whose difficulty level is as close as possible to the inital ability value(s);
   (c) `"thOpt"`: one selects the item(s) with the ability value where they get their maximum Fisher information is as close as possible to the inital ability value(s) (see Magis, 2013, for further details);
   (d) `"progressive"` for the *progressive* method (see `nextItem`);

(e) "proportional" for the *proportional* method (see nextItem).

If the "progressive" or "proportional" methods are selected, this will force the values of fixItems to NULL, seed to NULL, and nrItems to 1. Thus, a single item will be selected randomly.

The third method above will be used if and only if both fixItems and seed arguments are fixed to NULL. Otherwise, one of the first two methods will be used (see also testList for details about debugging misspecifications of the starting arguments).

The sequence of initial ability estimates is specified by the argument theta. For each component of theta one item will be picked up optimally according to the chosen startSelect argument. however, it is possible to perform randomesque selection at this stage by setting the randomesque argument to an integer value larger than one. In this case, the randomesque most optimal items are chosen per value of theta, and the final starting item is chosen randomly among the randomesque items. By default, only one item is picked up per ability level (and is therefore *the* most optimal one).

Only part of the full item bank can be made available for the selection of the first item(s), while others can be dropped out from this first step. This is fixed by the nAvailable argument, which is a vector with as many components as items in the bank and with zeros and ones only. Values 1 code for available items, values 0 for non-available items. By default, nAvailable is NULL and all items are available. Note that nrItems should never be larger than the number of available items (i.e. sum(nAvailable)). Otherwise an error message is returned.

Finally, in case of random selection of the first item(s), it is possible to force this selection to approximately match the content balancing control options set by arguments cbControl and cbGroup. Detailed description of these arguments (and their accurate format) are described in nextItem function. Note that cbControl should be tested with the test.cbList function prior to using startItems.

In practice, when fixing both the number of items to select (through nrItems) and the proportions of items per subgroups (through cbControl$props), the selection is made in two steps. First, the number of items per each subgroup is determined in order to reach closest distribution to the theoretical one. Second, within each subgroup the required number of items are randomly drawn (this draw can be fixed with the seed argument; fixing it to NA withdraws this option).

In the first step, it is possible that several subgroups are equally proable for selection; in this case the selection of the subgroups can be fixed by the argument random.cb. Otherwise the subgroups are sampled without any random seed control.

Value

A list with five arguments:

items	the selected items (identified by their number in the item bank) or NULL (if nrItems is 0).
par	the matrix of item parameters of the selected items (one row per item) or NULL (if nrItems is 0).
thStart	the sequence of starting ability values used for selecting the items or NA (if not applicable) or NULL (if nrItems is 0).
startSelect	the value of the startSelect argument or NA (if not applicable) or NULL (if nrItems is 0).

`names` either a vector with the names of the selected item(s) or `NULL` (if the item bank has no item names).

Note

Currently only the `"MFI"` value for `startSelect` is allowed for polytomous IRT models (i.e., when `model` is not `NULL`). Otherwise an error message is returned.

Author(s)

David Magis
Department of Psychology, University of Liege, Belgium
<david.magis@uliege.be>

Juan Ramon Barrada
Department of Psychology and Sociology, Universidad Zaragoza, Spain
<barrada@unizar.es>

References

Barton, M.A., and Lord, F.M. (1981). *An upper asymptote for the three-parameter logistic item-response model*. Research Bulletin 81-20. Princeton, NJ: Educational Testing Service.

Haley, D.C. (1952). *Estimation of the dosage mortality relationship when the dose is subject to error*. Technical report no 15. Palo Alto, CA: Applied Mathematics and Statistics Laboratory, Stanford University.

Magis, D. (2013). A note on the item information function of the four-parameter logistic model. *Applied Psychological Measurement*, *37*, 304-315. doi: 10.1177/0146621613475471

Magis, D. and Barrada, J. R. (2017). Computerized Adaptive Testing with R: Recent Updates of the Package *catR*. *Journal of Statistical Software*, *Code Snippets*, *76(1)*, 1-18. doi: 10.18637/jss.v076.c01

Magis, D., and Raiche, G. (2012). Random Generation of Response Patterns under Computerized Adaptive Testing with the R Package *catR*. *Journal of Statistical Software*, *48 (8)*, 1-31. doi: 10.18637/jss.v048.i08

See Also

`testList`, `genPolyMatrix`

Examples

```
## Dichotomous models ##

 # Loading the 'tcals' parameters
 data(tcals)

 # Item bank creation with 'tcals' item parameters
 bank <- as.matrix(tcals[,1:4])

 # Random selection of 4 starting items
```

```
startItems(bank, seed = 1, nrItems = 4)

# Random selection of 4 starting items without fixing the seed
startItems(bank, seed = NA, nrItems = 4)
startItems(bank, seed = NA, nrItems = 4) # may provide a different result!

## With content balancing control
prov <- breakBank(tcals)
cbGroup <- prov$cbGroup

# Creation of the 'cbList' list with arbitrary proportions
cbList <- list(names = c("Audio1", "Audio2", "Written1", "Written2", "Written3"),
       props = c(0.1, 0.2, 0.2, 0.2, 0.3))

startItems(bank, seed = 1, nrItems = 3, cbControl = cbList, cbGroup = cbGroup)
startItems(bank, seed = NA, nrItems = 3, cbControl = cbList, cbGroup = cbGroup,
           random.cb = 1)

# Selection of the first 5 starting items
startItems(bank, fixItems = 1:5)

# Selecting 1 starting item, initial ability estimate is 0
startItems(bank)

# Selecting 3 starting items for ability levels -1, 0 and 2
startItems(bank, theta = c(-1, 0, 2))

# Same with 5 randomesque items per theta value
startItems(bank, theta = c(-1, 0, 2), randomesque = 5)

# 5 randomesque items per theta value, with fixed random seed number
startItems(bank, theta = c(-1, 0, 2), randomesque = 5, random.seed = 1)

# Idem but with 'bOpt' criterion
startItems(bank, theta = c(-1, 0, 2), startSelect = "bOpt")

# Selecting only the first 10 items as available items
avail <- c(rep(1, 10), rep(0, nrow(bank)-10))
startItems(bank, theta = c(-1, 0, 2), nAvailable = avail)

## Not run:
# Selecting too many items among available ones
startItems(bank, nrItems = 11, theta = 2, halfRange = 3, nAvailable = avail)

## End(Not run)

## Polytomous models ##

# Generation of an item bank under GRM with 100 items and at most 4 categories
m.GRM <- genPolyMatrix(100, 4, "GRM")
m.GRM <- as.matrix(m.GRM)

# Random selection of 4 starting items
```

```
 startItems(m.GRM, model = "GRM", seed = 1, nrItems = 4)

 # Selection of the first 5 starting items
 startItems(m.GRM, model = "GRM", fixItems = 1:5)

 # Selecting 3 starting items for theta values -1, 0 and 2
 startItems(m.GRM, model = "GRM", theta = c(-1, 0, 2))

## Not run:

 # Idem but with 'bOpt' criterion
 startItems(m.GRM, model = "GRM", nrItems = 3, theta = 1, halfRange = 2,
            startSelect = "bOpt")

## End(Not run)

 # Selecting only the first 10 items as available items
 avail <- c(rep(1, 10), rep(0, nrow(m.GRM)-10))
 startItems(m.GRM, model = "GRM", theta = c(-1, 0, 2),
            nAvailable = avail)

## Not run:

 # Selecting too many items among available ones
 startItems(m.GRM, model = "GRM", theta = seq(from = -2, to = 2, length = 11),
            nAvailable = avail)

## End(Not run)

 # Loading the cat_pav data
 data(cat_pav)
 cat_pav <- as.matrix(cat_pav)

 # Random selection of 4 starting items
 startItems(cat_pav, model = "GPCM", seed = 1, nrItems = 4)

 # Selection of the first 5 starting items
 startItems(cat_pav, model = "GPCM", fixItems = 1:5)

 # Selecting 3 starting items for theta values -1, 0 and 2
 startItems(cat_pav, model = "GPCM", theta = c(-1, 0, 2))

## Not run:

 # Idem but with 'bOpt' criterion
 startItems(cat_pav, model = "GPCM", theta = c(-1, 0, 2), startSelect = "bOpt")

## End(Not run)

 # Selecting only the first 10 items as available items
 avail <- c(rep(1, 10), rep(0, nrow(cat_pav)-10))
 startItems(cat_pav, model = "GPCM", theta = c(-1, 0, 2), nAvailable = avail)
```

```
## Not run:

 # Selecting too many items among available ones
 startItems(cat_pav, model = "GPCM", theta = seq(from = -2, to = 2, length = 11),
            nAvailable = avail)

## End(Not run)
```

`tcals` *Items parameters of the TCALS 1998 data set and subgroups of items*

Description

The TCALS (*Test de Connaissance en Anglais Langue Seconde*) is an aptitude test of English language as a second language in the French speaking college of Outaouais (Gatineau, QC, Canada). The test consists of 85 items and is administered every year to newly incoming students. The item parameters of the year 1998 have been estimated under the 3PL model. Inattention parameters are therefore fixed to one. Subgroups of items are also included for content balancing purposes.

Format

A matrix with 85 rows and five columns, respectively holding the discrimination, difficulty, pseudo-guessing and inattention parameters as calibrated on the results of the 1998 application of the TCALS questionnaire. The fifth column holds the name of the subgroups of items:

- *Audio1*: listening comprehension of sentences (items **1** to **12**).
- *Audio2*: listening comprehension of dialogs and short texts (items **13** to **33**).
- *Written1*: written vocabulary exercises (items **34** to **46**).
- *Written2*: written grammar exercises (items **47** to **63**).
- *Written3*: written exercises of other types: reading and mistake detection (items **64** to **85**).

Source

The TCALS test was originally developed by Laurier, Froio, Pearo and Fournier (1998) and item parameters were obtained from Raiche (2002).

References

Laurier, M., Froio, L., Pearo C., and Fournier, M. (1998). *Test de classement d'anglais langue seconde au collegial*. Montreal, Canada: College de Maisonneuve.

Magis, D. and Barrada, J. R. (2017). Computerized Adaptive Testing with R: Recent Updates of the Package *catR*. *Journal of Statistical Software*, *Code Snippets*, *76(1)*, 1-18. doi: 10.18637/jss.v076.c01

Magis, D., and Raiche, G. (2012). Random Generation of Response Patterns under Computerized Adaptive Testing with the R Package *catR*. *Journal of Statistical Software*, *48 (8)*, 1-31. doi: 10.18637/jss.v048.i08

Raiche, G. (2002). *Le depistage du sous-classement aux tests de classement en anglais, langue seconde, au collegial [The detection of under classification to the collegial English as a second language placement tests]*. Gatineau, QC: College de l'Outaouais.

`test.cbList` *Testing the format of the list for content balancing under dichotomous or polytomous IRT models*

Description

This command tests whether format of the list to be assigned to `cbControl` argument of function `nextItem` is appropriate for content balancing, and returns a warning message otherwise.

Usage

```
test.cbList(list, cbGroup)
```

Arguments

`list`	a list of arguments to be tested. See **Details**.
`cbGroup`	a vector of character or factor names of the subgroups of items in the bank, or `NULL`.

Details

The `test.cbList` function checks whether the list provided in the `cbControl` argument of the `nextItem` and `randomCAT` functions, is accurate for controlling for content balancing. It mainly serves as an initial check for the `randomCAT` function.

The function returns an `"ok"` message if the arguments of `list` match the requirement of the list `cbControl` for content balancing. Otherwise, a message is returned with information about list - type mismatch. This will be the case if:

- `list` is not a list or has not exactly two elements,
- at least one of the argument names is incorrect,
- the lengths of the arguments are different, or different from the number of subgroups of items,
- the 'names' element does not match with the subgroups' names,
- the 'props' element is not numeric or holds negative values.

Each mismatch yields a different output message to help in debugging the problem.

Value

A list with two arguments:

`test`	a logical value indicating whether the format of the list is accurate (`TRUE`) or not (`FALSE`).
`message`	either a message to indicate the type of misspecification, or `"ok"` if the format is accurate.

Author(s)

David Magis
Department of Psychology, University of Liege, Belgium
<david.magis@uliege.be>

References

Magis, D. and Barrada, J. R. (2017). Computerized Adaptive Testing with R: Recent Updates of the Package *catR*. *Journal of Statistical Software*, *Code Snippets*, *76(1)*, 1-18. doi: 10.18637/jss.v076.c01

Magis, D., and Raiche, G. (2012). Random Generation of Response Patterns under Computerized Adaptive Testing with the R Package *catR*. *Journal of Statistical Software*, *48 (8)*, 1-31. doi: 10.18637/jss.v048.i08

See Also

nextItem, randomCAT

Examples

```
## Dichotomous models ##

 # Loading the 'tcals' parameters
 data(tcals)

 # Vector of group membership for 'tcals' data set
 group <- tcals$Group

 # Creation of a correct list with arbitrary proportions
 cbList <- list(names = c("Audio1", "Audio2", "Written1", "Written2", "Written3"),
        props = c(0.1, 0.2, 0.2, 0.2, 0.3))

 # Testing 'cbList'
 test.cbList(cbList, group)

 # Creation of an incorrect list (mismatch in first name)
 cbList <- list(names = c("audio1", "Audio2", "Written1", "Written2", "Written3"),
        props=c(0.1, 0.2, 0.2, 0.2, 0.3))
 test.cbList(cbList, group)

 # Creation of an incorrect list (mismatch in name of second
 # element)
 cbList <- list(names = c("Audio1", "Audio2", "Written1", "Written2", "Written3"),
        prop = c(0.1, 0.2, 0.2, 0.2, 0.3))
 test.cbList(cbList, group)

 # Creation of an incorrect list (second element shorter than
 # first element)
 cbList <- list(names = c("Audio1", "Audio2", "Written1", "Written2", "Written3"),
         props=c(0.1, 0.2, 0.2, 0.2))
```

```
test.cbList(cbList, group)

# Creation of an incorrect list (adding a third element)
cbList <- list(names = c("Audio1", "Audio2", "Written1", "Written2", "Written3"),
       props = c(0.1, 0.2, 0.2, 0.2), third = "hi")
test.cbList(cbList, group)


## Polytomous models ##

# Creation of an appropriate list for content balancing
# Equal proprotions across subgroups of items
cbList <- list(names = c("Group1", "Group2", "Group3", "Group4"), props = rep(1, 4))

# Creation of a "wrong" list
cbList2 <- list(names=c("group1","group2"),props = c(1, 1))

# Generation of an item bank under GRM with 100 items and at most 4 categories
m.GRM <- genPolyMatrix(100, 4, "GRM", cbControl = cbList)

# Breaking the 'm.GRM' bank
bank <- breakBank(m.GRM)

# Testing 'cbList' and 'cbList2'
test.cbList(cbList, bank$cbGroup)
test.cbList(cbList2, bank$cbGroup)

# Generation of an item bank under PCM with 100 items, 4 categories and groups
m.PCM <- genPolyMatrix(100, 4, "PCM", same.nrCat = TRUE, cbControl = cbList2)

# Breaking the 'm.PCM' bank
bank2 <- breakBank(m.PCM)

# Testing 'cbList' and 'cbList2'
test.cbList(cbList, bank2$cbGroup)
test.cbList(cbList2, bank2$cbGroup)
```

`testList` *Testing the format of the input lists*

Description

This command tests whether format of the input lists for the random generation of adaptive tests is convenient, and returns a warning message otherwise.

Usage

```
testList(list, type = "start")
```

Arguments

`list`	a list of arguments to be tested. See **Details**.
`type`	character: the type of list for checking. Possible values are `"start"` (default), `"test"`, `"stop"` and `"final"`. See **Details**.

Details

The `testList` function checks whether the list provided in the `list` argument is accurate for the selected `type`. It mainly serves as an initial check for the `randomCAT` function.

The four types of lists are: `"start"` with the parameters for selecting the first items; `"test"` with the options of the adaptive test (i.e. method for next item selection, provisional ability estimator and related information); `"stop"` with the options setting the stopping rule; and `"final"` with the options for final ability estimation. See the help file of `randomCAT` for further details about the different lists, their allowed arguments and their contents.

The function returns an `"ok"` message if the arguments of `list` match the requirement of the corresponding `type`. Otherwise, a message is returned with information about list - type mismatch. This will be the case if:

- `list` is not a list, or has no argument names,
- `list` has too many arguments for the `type` specified,
- at least one of the argument names is incorrect,
- the content of at least one argument is not adequate (e.g. character instead of numeric).

Each mismatch yields a different output message to help in debugging the problem.

Value

A list with two arguments:

`test`	a logical value indicating whether the format of the list is accurate (`TRUE`) or not (`FALSE`).
`message`	either a message to indicate the type of misspecification, or `"ok"` if the format is accurate.

Author(s)

David Magis
Department of Psychology, University of Liege, Belgium
<david.magis@uliege.be>

References

Magis, D. and Barrada, J. R. (2017). Computerized Adaptive Testing with R: Recent Updates of the Package *catR*. *Journal of Statistical Software*, *Code Snippets*, *76(1)*, 1-18. doi: 10.18637/jss.v076.c01

Magis, D., and Raiche, G. (2012). Random Generation of Response Patterns under Computerized Adaptive Testing with the R Package *catR*. *Journal of Statistical Software*, *48 (8)*, 1-31. doi: 10.18637/jss.v048.i08

See Also

randomCAT

Examples

```
# Creation and test of a 'start' list
start <- list(nrItems = 3, theta = c(-1, 0, 2, -2), randomesque = 2)
testList(start, type = "start")

# Using former 'halfRange' argument yields an error now:
start <- list(nrItems = 3, theta = 0, halfRange = 2)
testList(start, type = "start")

# Modification of the list to introduce a mistake
names(start)[1] <- "nrItem"
testList(start, type = "start")

# Creation and test of a 'test' list
test <- list(method = "WL", itemSelect = "bOpt")
testList(test, type = "test")

# Creation and test of a 'stop' list
stop <- list(method = "WL")
testList(stop, type = "test")

# Creation and test of a 'final' list (with mistake)
final <- list(method = "MAP")
testList(final, type = "final")
```

---

thetaEst *Ability estimation (dichotomous and polytomous models)*

Description

This command returns the ability estimate for a given response pattern and a given matrix of item parameters, either under the 4PL model or any suitable polytomous IRT model. Available estimators are maximum likelihood (ML), Bayes modal (BM), expected a posteriori (EAP), weighted likelihood (WL) and robust estimator (ROB).

Usage

```
thetaEst(it, x, model = NULL, D = 1, method = "BM", priorDist = "norm",
  priorPar = c(0, 1), weight = "Huber", tuCo = 1, range = c(-4, 4),
  parInt = c(-4, 4, 33), constantPatt = NULL, current.th = 0, bRange = c(-2, 2))
```

Arguments

it	numeric: a suitable matrix of item parameters. See **Details**.
x	numeric: a vector of item responses. Can also hold missing data (coded as NAs). See **Details**.
model	either NULL (default) for dichotomous models, or any suitable acronym for polytomous models. Possible values are "GRM", "MGRM", "PCM", "GPCM", "RSM" and "NRM". See **Details**.
D	numeric: the metric constant. Default is D=1 (for logistic metric); D=1.702 yields approximately the normal metric (Haley, 1952). Ignored if model is not NULL.
method	character: the ability estimator. Possible values are "BM" (default), "ML", "WL", "EAP" and "ROB". See **Details**.
priorDist	character: specifies the prior distribution. Possible values are "norm" (default), "unif" and "Jeffreys". Ignored if method is neither "BM" nor "EAP". See **Details**.
priorPar	numeric: vector of two components specifying the prior parameters (default is c(0,1)) of the prior ability distribution. Ignored if method is neither "BM" nor "EAP", or if priorDist="Jeffreys". See **Details**.
weight	character: the type of weight function for the robust estimator. Possible values are "Huber" (default) and "Tukey". Ignored if method is not "ROB" or if model is not NULL. See **Details**.
tuCo	numeric: the value of the tuning constant for the weight function (default is 1, suitable with "Huber" weight). Ignored if method is not "ROB" or if model is not NULL. See **Details**.
range	numeric: vector of two components specifying the range wherein the ability estimate must be looked for (default is c(-4,4)). Ignored if method=="EAP".
parInt	numeric: vector of three components, holding respectively the values of the arguments lower, upper and nqp of the eapEst command. Default vector is (-4, 4, 33). Ignored if method is not "EAP".
constantPatt	character: the method to estimate ability in case of constant pattern (i.e. only correct or only incorrect responses). Can be either NULL (default), "BM", "EAP", "WL", "fixed4", "fixed7" or "var". *Currently only implemented for dichotomous IRT models*. See **Details**.
current.th	numeric: the current ability estimate (default is zero). Required for ability estimation in presence of constant pattern. Ignored if constantPatt is neither "fixed4", "fixed7" nor "var". See **Details**.
bRange	numeric: vector of two componentns with the range of difficulty parameters in the parent item bank (default is c(-2,2)). *Currently only implemented for dichotomous IRT models*. See **Details**.

Details

Dichotomous IRT models are considered whenever model is set to NULL (default value). In this case, it must be a matrix with one row per item and four columns, with the values of the discrimination,

the difficulty, the pseudo-guessing and the inattention parameters (in this order). These are the parameters of the four-parameter logistic (4PL) model (Barton and Lord, 1981).

Polytomous IRT models are specified by their respective acronym: "GRM" for Graded Response Model, "MGRM" for Modified Graded Response Model, "PCM" for Partical Credit Model, "GPCM" for Generalized Partial Credit Model, "RSM" for Rating Scale Model and "NRM" for Nominal Response Model. The it still holds one row per item, end the number of columns and their content depends on the model. See genPolyMatrix for further information and illustrative examples of suitable polytomous item banks.

The vector of response patterns x can also hold missing responses (more useful in linear testing, not in CAT). In this case the missing responses must be coded as NA values. They are discarded from the ability estimation process.

Five ability estimators are available: the maximum likelihood (ML) estimator (Lord, 1980), the Bayes modal (BM) estimator (Birnbaum, 1969), the expected a posteriori (EAP) estimator (Bock and Mislevy, 1982), the weighted likelihood (WL) estimator (Warm, 1989) and the robust estimator (Schuster & Yuan, 2011). The selected estimator is specified by the method argument, with values "ML", "BM", "EAP", "WL" and "ROB" respectively.

For the BM and EAP estimators, three prior distributions are available: the normal distribution, the uniform distribution and Jeffreys' prior distribution (Jeffreys, 1939, 1946). The prior distribution is specified by the argument priorPar, with values "norm", "unif" and "Jeffreys", respectively. The priorPar argument is ignored if method="ML" or method="WL".

The argument priorPar determines either the prior mean and standard deviation of the normal prior distribution (if priorDist="norm"), or the range for defining the prior uniform distribution (if priorDist="unif"). This argument is ignored if priorDist="Jeffreys".

The parInt argument sets the range and the number of quadrature points for numerical integration in the EAP process. By default, it takes the vector value (-4, 4, 33), that is, 33 quadrature points on the range [-4; 4] (or, by steps of 0.25). See eapEst for further details.

Robust estimation requires an appropriate weight function that depends on an accurate tuning constant. Suggested functions are the Huber weight (Schuester and Yuan, 2011) and the Tukey weight (Mosteller and Tukey, 1977). Both can be set by the weight argument, with respective values "Huber" and "Tukey". Default function is Huber. Moreover, the tuCo argument specifies the tuning constant for the weight function. Default value is 1 and suggested for Huber weight (also by default), and value 4 is suggested for Tukey weight (Schuester and Yuan, 2011).

The range argument permits to limit the interval of investigation for the ML, BM, WL and ROB ability estimates (in particular, to avoid infinite ability estimates). The default range is [-4, 4].

Specific ability estimation methods are available in presence of constant patterns (that is with only correct or only incorrect responses) under dichotomous IRT models. These methods are specified by the argument constantPatt. By default it is set to NULL and hence ability is estimated with the specified method (even in presence of constant pattern). Six methods are currently available for constant patterns: "BM", "EAP" and "WL" that call for Bayes modal, expected a posteriori and weighted likelihood estimation respectively; "fixed4" and "fixed7" that perform fixed stepsize adjustment (i.e. increase or decrease of constant magnitude) with step 0.4 and 0.7 respectively; and "var" for variable stepsize adjustment. Note that in case odf stepsize adjustment, the range of difficultyu parameters must be provided through the bRange argument, as vector of two components (default value being c(-2,2)). See Dodd, De Ayala, and Koch (1995) for further details.

Value

The estimated ability level.

Note

1) It has been shown that in some cases the weighted likelihood estimator and the Bayes modal estimator with Jeffreys prior return exactly the same ability estimates. This is the case under the 2PL model, and subsequently the 1PL model (Warm, 1989) as well as under all polytomous models currently available (Magis, 2015). Nevertheless, both estimators remain available since (a) Jeffreys prior can also be considered with the EAP estimator, and (b) the 3PL and 4PL models are also available.

2) So far the robust estimator is available for dichotomous IRT models only.

Author(s)

David Magis
Department of Psychology, University of Liege, Belgium
<david.magis@uliege.be>

Lianne Ippel
Department of Psychology, University of Liege, Belgium
<g.j.e.ippel@gmail.com>

References

Barton, M.A., and Lord, F.M. (1981). *An upper asymptote for the three-parameter logistic item-response model*. Research Bulletin 81-20. Princeton, NJ: Educational Testing Service.

Birnbaum, A. (1969). Statistical theory for logistic mental test models with a prior distribution of ability. *Journal of Mathematical Psychology, 6*, 258-276. doi: 10.1016/00222496(69)900054

Bock, R. D., and Mislevy, R. J. (1982). Adaptive EAP estimation of ability in a microcomputer environment. *Applied Psychological Measurement, 6*, 431-444. doi: 10.1177/014662168200600405

Dodd, B. G., De Ayala, R. J., and Koch, W. R. (1995) . Computerized adaptive testing with polytomous items. *Applied Psychological Measurement, 19*, 5-22. doi: 10.1177/014662169501900103

Haley, D.C. (1952). *Estimation of the dosage mortality relationship when the dose is subject to error.* Technical report no 15. Palo Alto, CA: Applied Mathematics and Statistics Laboratory, Stanford University.

Jeffreys, H. (1939). *Theory of probability*. Oxford, UK: Oxford University Press.

Jeffreys, H. (1946). An invariant form for the prior probability in estimation problems. *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences, 186*, 453-461.

Lord, F.M. (1980). *Applications of item response theory to practical testing problems*. Hillsdale, NJ: Lawrence Erlbaum.

Magis, D. (2015). A note on weighted likelihood and Jeffreys modal estimation of proficiency levels in polytomous item response models. *Psychometrika, 80*, 200-204. doi: 10.1007/S1133601393785

Magis, D. and Barrada, J. R. (2017). Computerized Adaptive Testing with R: Recent Updates of the Package *catR*. *Journal of Statistical Software*, *Code Snippets*, *76(1)*, 1-18. doi: 10.18637/jss.v076.c01

Magis, D., and Raiche, G. (2012). Random Generation of Response Patterns under Computerized Adaptive Testing with the R Package *catR*. *Journal of Statistical Software*, *48 (8)*, 1-31. doi: 10.18637/jss.v048.i08

Mosteller, F., and Tukey, J. (1977). *Exploratory data analysis and regression*. Reading, MA: Addison-Wesley.

Schuester, C., and Yuan, K.-H. (2011). Robust estimation of latent ability in item response models. *Journal of Educational and Behavioral Statistics, 36*, 720)735.doi: 10.3102/1076998610396890

Warm, T.A. (1989). Weighted likelihood estimation of ability in item response models. *Psychometrika, 54*, 427-450. doi: 10.1007/BF02294627

See Also

eapEst, semTheta, genPolyMatrix

Examples

```
## Dichotomous models ##

 # Loading the 'tcals' parameters
 data(tcals)

 # Selecting item parameters only
 tcals <- as.matrix(tcals[,1:4])

 # Creation of a response pattern (tcals item parameters, true ability level 0)
 set.seed(1)
 x <- genPattern(0, tcals)

 # ML estimation
 thetaEst(tcals, x, method = "ML")

 # With first two responses missing
 x.mis <- x
 x.mis[1:2] <- NA
 thetaEst(tcals, x.mis, method = "ML")

 # BM estimation, standard normal prior distribution
 thetaEst(tcals, x)

 # BM estimation, uniform prior distribution upon range [-2,2]
 thetaEst(tcals, x, method = "BM", priorDist = "unif", priorPar = c(-2, 2))

 # BM estimation, Jeffreys' prior distribution
 thetaEst(tcals, x, method = "BM", priorDist = "Jeffreys")

 # EAP estimation, standard normal prior distribution
 thetaEst(tcals, x, method = "EAP")

 # EAP estimation, uniform prior distribution upon range [-2,2]
 thetaEst(tcals, x, method = "EAP", priorDist = "unif", priorPar = c(-2, 2))
```

```
# EAP estimation, Jeffreys' prior distribution
thetaEst(tcals, x, method = "EAP", priorDist = "Jeffreys")

# WL estimation
thetaEst(tcals, x, method = "WL")

# Robust estimation, Huber weight with tuning constant 1 (default)
thetaEst(tcals, x, method = "ROB")

# Robust estimation, Huber weight with tuning constant 2
thetaEst(tcals, x, method = "ROB", tuCo = 2)

# Robust estimation, Tukey weight with tuning constant 4
thetaEst(tcals, x, method = "ROB", weight = "Tukey", tuCo = 4)

# Creation of two constant patterns and estimation with WL,
# 'fixed4', 'fixed7' and 'var' stepsize adjustments
x0 <- rep(0,nrow(tcals))
x1 <- x0 + 1
thetaEst(tcals, x0, constantPatt = "WL") # equivalent to thetaEst(tcals, x0, method = "WL")
thetaEst(tcals, x1, constantPatt = "WL") # equivalent to thetaEst(tcals, x1, method = "WL")
thetaEst(tcals, x0, constantPatt = "fixed4")
thetaEst(tcals, x1, constantPatt = "fixed4")
thetaEst(tcals, x0, constantPatt = "fixed7")
thetaEst(tcals, x1, constantPatt = "fixed7")
thetaEst(tcals, x0, constantPatt = "var")
thetaEst(tcals, x1, constantPatt = "var")

## Not run:

## Polytomous models ##

# Generation of an item bank under GRM with 100 items and at most 4 categories
m.GRM <- genPolyMatrix(100, 4, "GRM")
m.GRM <- as.matrix(m.GRM)

# Creation of a response pattern (true ability level 0)
set.seed(1)
x <- genPattern(0, m.GRM, model = "GRM")

# ML estimation
thetaEst(m.GRM, x, model = "GRM", method = "ML")

# BM estimation, standard normal prior distribution
thetaEst(m.GRM, x, model = "GRM")

# BM estimation, uniform prior distribution upon range [-2,2]
thetaEst(m.GRM, x, model = "GRM", method = "BM", priorDist = "unif",
         priorPar = c(-2, 2))

# BM estimation, Jeffreys' prior distribution
thetaEst(m.GRM, x, model = "GRM", method = "BM", priorDist = "Jeffreys")
```

```
# EAP estimation, standard normal prior distribution
thetaEst(m.GRM, x, model = "GRM", method = "EAP")

# EAP estimation, uniform prior distribution upon range [-2,2]
thetaEst(m.GRM, x, model = "GRM", method = "EAP", priorDist = "unif",
         priorPar = c(-2, 2))

# EAP estimation, Jeffreys' prior distribution
thetaEst(m.GRM, x, model = "GRM", method = "EAP", priorDist = "Jeffreys")

# WL estimation
thetaEst(m.GRM, x, model = "GRM", method = "WL")


# Generation of an item bank under PCM with 20 items and 4 categories
m.PCM <- genPolyMatrix(20, 4, "PCM", same.nrCat = TRUE)
m.PCM <- as.matrix(m.PCM)

# Creation of a response pattern (true ability level 0)
set.seed(1)
x <- genPattern(0, m.PCM, model = "PCM")

# ML estimation
thetaEst(m.PCM, x, model = "PCM", method = "ML")

# BM estimation, standard normal prior distribution
thetaEst(m.PCM, x, model = "PCM")

# BM estimation, uniform prior distribution upon range [-2,2]
thetaEst(m.PCM, x, model = "PCM", method = "BM", priorDist = "unif",
         priorPar = c(-2, 2))

# BM estimation, Jeffreys' prior distribution
thetaEst(m.PCM, x, model = "PCM", method = "BM", priorDist = "Jeffreys")

# EAP estimation, standard normal prior distribution
thetaEst(m.PCM, x, model = "PCM", method = "EAP")

# EAP estimation, uniform prior distribution upon range [-2,2]
thetaEst(m.PCM, x, model = "PCM", method = "EAP", priorDist = "unif",
         priorPar = c(-2, 2))

# EAP estimation, Jeffreys' prior distribution
thetaEst(m.PCM, x, model = "PCM", method = "EAP", priorDist = "Jeffreys")

# WL estimation
thetaEst(m.PCM, x, model = "PCM", method = "WL")

## End(Not run)
```

Index

∗ **datasets**
    cat_pav, 6
    tcals, 108

aStratified, 2, *77*, *84*

breakBank, 4, *76*, *82*, *84*

cat_pav, 6
checkStopRule, 7

eapEst, 9, *18, 19*, *21*, *49*, *78*, *84*, *88, 89*, *114, 115*, *117*
eapSem, 12, *18, 19*, *78*, *84*, *91*
EPV, 16, *60*

fullDist, 20, *89*, *91*

GDI, 23, *60*
genDichoMatrix, 27
genPattern, 30, *76*, *84*
genPolyMatrix, *3, 4*, *10, 11*, *13, 14*, *17*, *19*, *26*, *30, 31*, 32, *37, 38*, *41, 42*, *44*, *46*, *50, 51*, *54*, *56*, *61*, *64*, *69*, *72*, *76*, *84*, *88*, *91*, *103*, *105*, *115*, *117*

Ii, 37, *51*, *56*, *73*
integrate.catR, *10, 11*, *14*, *25, 26*, 39, *45, 46*, *54*, *56*

Ji, 40

KL, *40*, 43, *60*, *64*, *84*

MEI, 48, *60, 61*, *64*, *69, 70*, *84*
MWI, 53, *60, 61*, *64*, *84*

nextItem, *17*, *19*, *25, 26*, *29*, *34*, *44*, *46*, *50, 51*, *54*, *56*, 57, *62*, *69, 70*, *76–79*, *84*, *103, 104*, *109, 110*

OIi, *50, 51*, 68

Pi, *8*, *30, 31*, *33*, *35*, *38*, 71
plot.cat *(*randomCAT*)*, 74
plot.catResult *(*simulateRespondents*)*, 95
print.cat *(*randomCAT*)*, 74
print.catResult *(*simulateRespondents*)*, 95

randomCAT, *5*, *7, 8*, *27*, *29*, *34*, *64*, 74, *97, 98*, *100*, *109, 110*, *112, 113*
rbinom, *30, 31*
rmultinom, *30, 31*

semTheta, *21*, *23*, *42*, *79*, *84*, 87, *117*
set.seed, *27*
simulateRespondents, *8*, 95
startItems, *61*, *77*, *84*, 102, *104*

tcals, 108
test.cbList, *62*, *64*, *104*, 109
testList, *80*, *84*, *104, 105*, 111
thetaEst, *5*, *11*, *14*, *23*, *38*, *42*, *45*, *51*, *61*, *64*, *72, 73*, *79*, *84*, *91*, 113