Package: mstR (via r-universe)

August 23, 2024

Type Package

Title Procedures to Generate Patterns under Multistage Testing

Version 1.2

Date 2018-03-29

Author David Magis (U Liege, Belgium), Duanli Yan (ETS, USA), Alina von Davier (ACTNext, USA)

Maintainer David Magis <david.magis@uliege.be>

Depends R (>= 2.8.0)

Description Generation of response patterns under dichotomous and polytomous computerized multistage testing (MST) framework. It holds various item response theory (IRT) and score-based methods to select the next module and estimate ability levels (Magis, Yan and von Davier (2017, ISBN:978-3-319-69218-0)).

License GPL (>= 2)

LazyLoad yes

NeedsCompilation no

X-CRAN-Comment Archived on 2018-03-28 for policy violation. Leaves .pdf file in /tmp

Repository CRAN

Date/Publication 2018-03-29 20:31:49 UTC

Contents

`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.mstR` 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 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. URL http://www.jstatsoft.org/v48/i08/

See Also

thetaEst, genPolyMatrix, integrate.mstR

Examples

```
## Dichotomous models ##

 # Generation of an item bank under 3PL with 100 items
 m.3PL <- genDichoMatrix(100, model = "3PL")
 m.3PL <- as.matrix(m.3PL)

 # Creation of a response pattern (true ability level 0)
 set.seed(1)
 x <- genPattern(0, m.3PL)

 # EAP estimation, standard normal prior distribution
 eapEst(m.3PL, x)

 # EAP estimation, uniform prior distribution upon range [-2,2]
 eapEst(m.3PL, x, priorDist = "unif", priorPar = c(-2, 2))

 # EAP estimation, Jeffreys' prior distribution
 eapEst(m.3PL, x, priorDist = "Jeffreys")

 # Changing the integration settings
 eapEst(m.3PL, x, nqp = 100)

## 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
 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")


  # Generation of a item bank under PCM with 20 items and at most 3 categories
  m.PCM <- genPolyMatrix(20, 3, "PCM")
  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")

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

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

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

## End(Not run)
```

`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.mstR 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 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. URL http://www.jstatsoft.org/v48/i08/

See Also

thetaEst, genPolyMatrix

Examples

```
## Not run:

## Dichotomous models ##

 # Generation of an item bank under 3PL with 100 items
 m.3PL <- genDichoMatrix(100, model = "3PL")
 m.3PL <- as.matrix(m.3PL)

 # Creation of a response pattern (true ability level 0)
 set.seed(1)
 x <- genPattern(0, m.3PL)

 # EAP estimation, standard normal prior distribution
 th <- eapEst(m.3PL, x)
 c(th, eapSem(th, m.3PL, x))

 # EAP estimation, uniform prior distribution upon range [-2,2]
 th <- eapEst(m.3PL, x, priorDist = "unif", priorPar = c(-2, 2))
 c(th, eapSem(th, m.3PL, x, priorDist = "unif", priorPar=c(-2, 2)))

 # EAP estimation, Jeffreys' prior distribution
 th <- eapEst(m.3PL, x, priorDist = "Jeffreys")
```

```
 c(th, eapSem(th, m.3PL, x, priorDist = "Jeffreys"))


## 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"))


 # Generation of a item bank under PCM with 20 items and at most 3 categories
 m.PCM <- genPolyMatrix(20, 3, "PCM")
 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")

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

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

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

## End(Not run)
```

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, 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.
`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 in MST simulations.

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.

The random generation of item parameters si being controled by the seed argument.

The output is a data frame with 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.

Value

A data frame with four 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.

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 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. URL http://www.jstatsoft.org/v48/i08/

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))
```

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 ability value.

Usage

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

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`.
`seed`	either the random seed value or `NULL` (default). See **Details**.

Details

This function permits to randomly generate item responses for a given ability level, 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

A vector with the item responses in the order of appearance of the items in the it matrix.

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 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. URL http://www.jstatsoft.org/v48/i08/

See Also

rbinom and rmultinom for random draws; genPolyMatrix, Pi

Examples

```
## Dichotomous models ##

 # Generation of an item bank under 3PL with 100 items
 m.3PL <- genDichoMatrix(100, model = "3PL")
 m.3PL <- as.matrix(m.3PL)

 # Generation of a response pattern for ability level 0
 genPattern(th = 0, m.3PL)

 # Generation of a single response for the first item only
 genPattern(th = 0, m.3PL[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 a single response for the first item only
genPattern(0, m.GRM[1,], model = "GRM")

# Generation of a item bank under PCM with 20 items and at most 3 categories
m.PCM <- genPolyMatrix(20, 3, "PCM")
m.PCM <- as.matrix(m.PCM)

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

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

---

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)
```

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**.

Details

The `genPolyMatrix` permits to quickly generate a polytomous item bank in suitable format for further use in e.g. computing item response propbabilities 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):

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

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.

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"`.

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 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. URL http://www.jstatsoft.org/v48/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)
```

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 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. URL http://www.jstatsoft.org/v48/i08/

See Also

`Pi`, `thetaEst`, `genPolyMatrix`

Examples

```
## Dichotomous models ##

 # Generation of an item bank under 3PL with 100 items
 m.3PL <- genDichoMatrix(100, model = "3PL")
 m.3PL <- as.matrix(m.3PL)
```

```
 # Item information functions and derivatives
 # (various th and D values)
 Ii(th = 0, m.3PL)
 Ii(th = 0, m.3PL, D = 1.702)
 Ii(th = 1, m.3PL)

## 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")

 # Generation of a item bank under PCM with 20 items and at most 3 categories
 m.PCM <- genPolyMatrix(20, 3, "PCM")
 m.PCM <- as.matrix(m.PCM)

 # Computation of item information and derivatives for ability level 1
 Ii(1, m.PCM, model = "PCM")
```

---

integrate.mstR *Numerical integration by linear interpolation (for mstR 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.mstR(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 `mstR` 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

The `integrate.xy` function in package `sfsmisc`

Examples

```
  x <- seq(from = -4, to = 4, length = 33)
 y <- exp(x)
 integrate.mstR(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 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. URL http://www.jstatsoft.org/v48/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 ##

 # Generation of an item bank under 3PL with 100 items
 m.3PL <- genDichoMatrix(100, model = "3PL")
 m.3PL <- as.matrix(m.3PL)

 # Various J functions and derivatives
 # (various th and D values)
 Ji(th = 0, m.3PL)
 Ji(th = 0, m.3PL, D = 1.702)
 Ji(th = 1, m.3PL)

## 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")

 # Generation of a item bank under PCM with 20 items and at most 3 categories
```

```
m.PCM <- genPolyMatrix(20, 3, "PCM")
m.PCM <- as.matrix(m.PCM)

# Computation of J function and derivatives for ability level 1
Ji(1, m.PCM, model = "PCM")
```

MKL — *Module Kullback-Leibler (MKL) and posterior module Kullback-Leibler (MKLP)*

Description

This command returns the value of the Kullback-Leibler (MKL) or the posterior Kullback-Leibler (MKLP) weighted likelihood for a given target module and an item bank (both under dichotomous and polytomous IRT models).

Usage

```
MKL(itemBank, modules, target.mod, theta = NULL, it.given, x, model = NULL,
    lower = -4, upper = 4, nqp = 33, type = "MKL", priorDist = "norm",
    priorPar = c(0, 1), D = 1)
```

Arguments

`itemBank`	numeric: a suitable matrix of item parameters. See **Details**.
`modules`	a binary matrix that specifies the item membership to th emodules. See **Details**.
`target.mod`	numeric: the module (referred to as its column number in the `modules` matrix) for which the information must be computed.
`theta`	either the provisional ability level or `NULL` (default). See **Details**.
`it.given`	numeric: a vector of item indicators for all previously administered items.
`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). The length of `x` must be equal to the length of `it.given`.
`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 Kullback-Leibler information to be computed. Possible values are `"MKL"` (default) and `"MKLP"`. 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

This function extends the KL and the KLP methods to select the next item in CAT, to the MST framework. This command serves as a subroutine for the nextModule 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.

FROM HERE

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). Set M as the number of items in the target module of interest (not yet 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 of the target module ($j = 1, ..., M$) for a given ability level θ. Then, module Kullack-Leibler (MKL) information is defined as

$$MKL(\theta||\hat{\theta}_k) = \sum_{j=1}^{M} \sum_{t=0}^{g_j} P_{jt}(\hat{\theta}_k) \log \left(\frac{P_{jt}(\hat{\theta}_k)}{P_{jt}(\theta)} \right).$$

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 ($j = 1, ..., M$) for a given ability level θ. Then, MKL information reduces to

$$MKL(\theta||\hat{\theta}) = \sum_{j=1}^{M} \left\{ P_j(\hat{\theta}) \log \left(\frac{P_j(\hat{\theta}_k)}{P_j(\theta)} \right) + [1 - P_j(\hat{\theta}_k)] \log \left(\frac{1 - P_j(\hat{\theta}_k)}{1 - P_j(\theta)} \right) \right\}.$$

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

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

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

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

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

These integrals are approximated by the integrate.mstR 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, "MKL", computes the MKL value, while the MKLP value is obtained with type="MKLP". 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 "MKLP".

The provisional response pattern and the related administered items are provided by the vectors x and it.given respectively. The target module (for which the maximum information is computed) is given by its column number in the modules matrix, through the target.mod argument.

The provisioal (ad-interim) ability level can be provided through the theta argument. If not provided or set to NULL (default value), it is then internally computed as the ML estimate of ability for the given response pattern x and the previously administered items it.given.

Value

The required KL (or KLP) weighted module likelihood for the target module.

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.

See Also

Ii, nextModule, integrate.mstR, genPolyMatrix

Examples

```
## Dichotomous models ##

 # Generation of an item bank under 2PL, made of 7 successive modules that target
 # different average ability levels and with different lengths
 # (the first generated item parameters hold two modules of 8 items each)
 it <- rbind(genDichoMatrix(16, model = "2PL"),
             genDichoMatrix(6, model = "2PL", bPrior = c("norm", -1, 1)),
             genDichoMatrix(6, model = "2PL", bPrior = c("norm", 1, 1)),
             genDichoMatrix(9, model = "2PL", bPrior = c("norm", -2, 1)),
             genDichoMatrix(9, model = "2PL", bPrior = c("norm", 0, 1)),
             genDichoMatrix(9, model = "2PL", bPrior = c("norm", 2, 1)))
 it <- as.matrix(it)

 # Creation of the 'module' matrix to list item membership in each module
 modules <- matrix(0, 55, 7)
 modules[1:8, 1] <- modules[9:16, 2] <- modules[17:22, 3] <- 1
 modules[23:28, 4] <- modules[29:37, 5] <- modules[38:46, 6] <- 1
 modules[47:55, 7] <- 1

 # Creation of the response pattern for module 1 and true ability level 0
 x <- genPattern(th = 0, it = it[1:8,], seed = 1)

 # MKL for module 3
 MKL(it, modules, target.mod = 3, it.given = 1:8, x = x)

 # Same with pre-estimation of ability by ML (same result)
 th <- thetaEst(it[1:8,], x, method = "ML")
 MKL(it, modules, target.mod = 3, it.given = 1:8, x = x, theta = th)

 # Same with pre-estimation of ability by EAP (different result)
 th <- thetaEst(it[1:8,], x, method = "EAP")
 MKL(it, modules, target.mod = 3, it.given = 1:8, x = x, theta = th)

 # MKLP for module 3
 MKL(it, modules, target.mod = 3, it.given = 1:8, x = x, type = "MKLP")

 # MKL for for module 3, different integration range
 MKL(it, modules, target.mod = 3, it.given = 1:8, x = x, lower = -2, upper = 2, nqp = 20)

 # MKLP for module 3, uniform prior distribution on the range [-2,2]
 MKL(it, modules, target.mod = 3, it.given = 1:8, x = x, type = "MKLP",
     priorDist = "unif", priorPar = c(-2, 2))


## Polytomous models ##

 # Same structure as above but parameters are now generated from PCM with at most
```

```
# 4 categories
it.pol <- genPolyMatrix(55, model = "PCM", nrCat = 4)
it.pol <- as.matrix(it)

# Creation of the response pattern for module 1 and true ability level 0
x <- genPattern(th = 0, it = it.pol[1:8,], seed = 1)

# MKL for module 3
MKL(it.pol, modules, target.mod = 3, it.given = 1:8, x = x, model = "PCM")

# Same with pre-estimation of ability by ML (same result)
th <- thetaEst(it.pol[1:8,], x, method = "ML", model = "PCM")
MKL(it.pol, modules, target.mod = 3, it.given = 1:8, x = x, theta = th,
    model = "PCM")

# Same with pre-estimation of ability by EAP (different result)
th <- thetaEst(it.pol[1:8,], x, method = "EAP", model = "PCM")
MKL(it.pol, modules, target.mod = 3, it.given = 1:8, x = x, theta = th,
    model = "PCM")

# MKLP for module 3
MKL(it.pol, modules, target.mod = 3, it.given = 1:8, x = x, type = "MKLP",
    model = "PCM")

# MKL for for module 3, different integration range
MKL(it.pol, modules, target.mod = 3, it.given = 1:8, x = x, lower = -2,
    upper = 2, nqp = 20, model = "PCM")

# MKLP for module 3, uniform prior distribution on the range [-2,2]
MKL(it.pol, modules, target.mod = 3, it.given = 1:8, x = x, type = "MKLP",
    priorDist = "unif", priorPar = c(-2, 2), model = "PCM")
```

MWMI — *Maximum likelihood weighted module information (MLWMI) and maximum posterior weighted module information (MPWMI)*

Description

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

Usage

```
MWMI(itemBank, modules, target.mod, it.given, x, model = NULL, lower = -4,
    upper = 4, nqp = 33, type = "MLWMI", priorDist = "norm", priorPar = c(0, 1),
    D = 1)
```

Arguments

itemBank	numeric: a suitable matrix of item parameters. See **Details**.
modules	a binary matrix that specifies the item membership to th emodules. See **Details**.
target.mod	numeric: the module (referred to as its column number in the modules matrix) for which the information must be computed.
it.given	numeric: a vector of item indicators for all previously administered items.
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). The length of x must be equal to the length of it.given.
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 "MLWMI" (default) and "MPWMI". 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

This function extends the MLWI and the MPWI methods to select the next item in CAT, to the MST framework. This command serves as a subroutine for the nextModule 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 $I(\theta)$ as the information function of the module of interest (specified through `target.mod`), made by the sum of all item informations functions from this module, and evaluated at θ. Set finally $L(\theta|x_1, ..., x_k)$ as the likelihood function evaluated at θ, given the provisional response pattern. Then, the LWMI for the module is given by

$$LWMI = \int I(\theta) L(\theta|x_1, ..., x_k) d\theta$$

and the PWMI by

$$PWMI = \int I(\theta) \pi(\theta) L(\theta|x_1, ..., x_k) d\theta$$

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.mstR` 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, `"MLWMI"`, computes the MLWMI value, while the MPWMI value is obtained with `type="MPWMI"`. 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 `"MPWMI"`.

The provisional response pattern and the related administered items are provided by the vectors `x` and `it.given` respectively. The target module (for which the maximum information is computed) is given by its column number in the `modules` matrix, through the `target.mod` argument.

Value

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

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.

See Also

Ii, nextModule, integrate.mstR, genPolyMatrix

Examples

```
## Dichotomous models ##

 # Generation of an item bank under 2PL, made of 7 successive modules that target
 # different average ability levels and with different lengths
 # (the first generated item parameters hold two modules of 8 items each)
 it <- rbind(genDichoMatrix(16, model = "2PL"),
             genDichoMatrix(6, model = "2PL", bPrior = c("norm", -1, 1)),
             genDichoMatrix(6, model = "2PL", bPrior = c("norm", 1, 1)),
             genDichoMatrix(9, model = "2PL", bPrior = c("norm", -2, 1)),
             genDichoMatrix(9, model = "2PL", bPrior = c("norm", 0, 1)),
             genDichoMatrix(9, model = "2PL", bPrior = c("norm", 2, 1)))
 it <- as.matrix(it)

 # Creation of the 'module' matrix to list item membership in each module
 modules <- matrix(0, 55, 7)
 modules[1:8, 1] <- modules[9:16, 2] <- modules[17:22, 3] <- 1
 modules[23:28, 4] <- modules[29:37, 5] <- modules[38:46, 6] <- 1
 modules[47:55, 7] <- 1

 # Creation of the response pattern for module 1 and true ability level 0
 x <- genPattern(th = 0, it = it[1:8,], seed = 1)

 # MLWMI for module 3
 MWMI(it, modules, target.mod = 3, it.given = 1:8, x = x)

 # MPWMI for module 3
 MWMI(it, modules, target.mod = 3, it.given = 1:8, x = x, type = "MPWMI")

 # MLWMI for for module 3, different integration range
 MWMI(it, modules, target.mod = 3, it.given = 1:8, x = x, lower = -2, upper = 2, nqp = 20)

 # MPWI for module 3, uniform prior distribution on the range [-2,2]
 MWMI(it, modules, target.mod = 3, it.given = 1:8, x = x, type = "MPWMI",
     priorDist = "unif", priorPar = c(-2, 2))


## Polytomous models ##

 # Same structure as above but parameters are now generated from PCM with at most
 # 4 categories
 it.pol <- genPolyMatrix(55, model = "PCM", nrCat = 4)
 it.pol <- as.matrix(it)

 # Creation of the response pattern for module 1 and true ability level 0
 x <- genPattern(th = 0, it = it.pol[1:8,], seed = 1)

# MLWMI for module 3
```

```
MWMI(it.pol, modules, target.mod = 3, it.given = 1:8, x = x, model = "PCM")

# MPWMI for module 3
MWMI(it.pol, modules, target.mod = 3, it.given = 1:8, x = x, type = "MPWMI",
    model = "PCM")

# MLWMI for for module 3, different integration range
MWMI(it.pol, modules, target.mod = 3, it.given = 1:8, x = x, lower = -2,
    upper = 2, nqp = 20, model = "PCM")

# MPWI for module 3, uniform prior distribution on the range [-2,2]
MWMI(it.pol, modules, target.mod = 3, it.given = 1:8, x = x, type = "MPWMI",
    priorDist = "unif", priorPar = c(-2, 2), model = "PCM")
```

`nextModule` *Selection of the next module in MST*

Description

This command selects the next module to be administered in the multistage test, either bases on IRT scoring or on test score and by either providing thresholds or optimally selecting the next module.

Usage

```
nextModule(itemBank, modules, transMatrix, model = NULL, current.module,
   out, x = NULL, cutoff = NULL, theta = 0, criterion = "MFI",
   priorDist = "norm", priorPar = c(0, 1), D = 1, range = c(-4, 4),
   parInt = c(-4, 4, 33), randomesque = 1, random.seed = NULL)
```

Arguments

`itemBank`	a suitable matrix of item parameters. See **Details**.
`modules`	a binary matrix that specifies the item membership to the modules. See **Details**.
`transMatrix`	a binary squared matrix representing the structure of the MST and the transitions between the moduels and the stages. 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**.
`current.module`	integer: the module number (defined as the corresponding column number in the `modules` matrix) that indicates the last administered module.
`out`	numeric: the vector of item indicators (defined as the row numbers in the `itemBank` matrix) of previously administered items.
`x`	either a numeric vector of responses to previously administered items or `NULL` (default). Ignored if `criterion` is either `MFI` or `random`. See **Details**.

cutoff	either a suitable matrix of cut-off values or NULL (default). See Details.
theta	numeric: the current ability level for selecting the next module (default is 0). It can also hold the current test score made of the sum of responses to all administered items (when cutoff is provided). See **Details**.
criterion	character: the method for next item selection. Possible values are "MFI" (default), "MLWMI", "MPWMI", "MKL", "MKLP" and "random". Ignored if cutoff is not NULL. 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 criterion is neither "MPWMI" nor "KLP", or if cutoff is not NULL.
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 criterion is neither "MPWI" nor "KLP", or if cutoff is not NULL.
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 neither "MLWMI", "MPWMI", "KL", nor "KLP", or if cutoff is not NULL. See **Details**.
randomesque	numeric: a probability value to select the optimal module. Default is one so optimal module is always chosen. See **Details**.
random.seed	either NULL (default) or a numeric value to fix the random seed of randomesque selection of the module. Ignored if randomesque is equal to one.

Details

This function permits to select the next module of the MST. 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 modules argument must be a binary 0/1 matrix with as many rows as the item bank itemBank and as many columns as the number of modules. Values of 1 indicate to which module(s) the items belong to, i.e. a value of 1 on row *i* and column *j* means that the *i*-th item belongs to the *j*-th module.

The transMatrix argument must be a binary 0/1 square matrix with as many rows (and columns) as the number of modules. All values of 1 indicate the possible transitions from one module to another, i.e. a value of 1 on row *i* and column *j* means that the MST can move from *i*-th module to *j*-th module.

The two main approaches to select the next module are based on cut-off scores (to be provided) or by optimal module selection.

Optimal module selection is performed by providing an appropriate value to the criterion argument. Possible methods are:

1. "MFI" for maximum Fisher information(default);
2. "MLWMI" for maximum likelihood weighted module information;
3. "MPWMI" for maximum posterior weighted module information;
4. "MKL" for module Kullabck-Leibler selection;
5. "MKLP" for module posterior Kullabck-Leibler selection;
6. "random" for random selection.

See MWMI and MKL for further details.

In case of selection by predefined cut-off scores, the cutoff argument must be supplied by a matrix with as many rows as the number of thresholds between pairs of modules, and with three columns. Each row of the cutoff matrix holds first the two module indicators (i.e., their column number in the modules matrix) and then the threshold. For instance, the row c(3, 4, 1) indicates that the selection threshold between modules 3 and 4 is 1. Thus, if the next module must be chosen between modules 3 and 4, the module 3 is selected if the score is *strictly smaller* than 1, and module 4 is chosen if the score is gretar than or equal to 1.

This allows the selection among multiple modules within a stage as follows. Let modules 5 to 7 be the allowed moduels for selection, and set -1 and 1 as thresholds to distinguish between modules 5 and 6 and modules 6 and 7. By this way, module 5 is chosen if the score is strictly smaller than -1, module 7 if the score is larger than (or equal to) 1, and module 6 otherwise. This design is simply modeled through the cutoff matrix by including the rows c(5, 6, -1) nd c(6, 7, 1). Note that the order of the rows in the cutoff matrix is irrelevant. Moreover, integer cut-off scores (when theta is the test score) or numeric values (when theta is an ability estimate) are allowed in the cutoff matrix.

By default cutoff is NULL and optimal module selection is performed).

Whatever the method for next module selection (by optimal criterion or via cut-off scores), the randomesque argument allows for selecting a module that is not the optimal one. This argument takes a probability value (i.e., between zero and one) that sets the probability that the optimal module is eventually selected. All other elligible modules from the stage will be randomly chosen with a probability equal to (1-randomesque)/$K-1$ where K is he number of elligible modules in the stage (including the optimal one). This allows for module overexposure control. The random.seed argument permits ot fix the seed for random selection.

Value

A list with six arguments:

module the selected module (identified by its column number in the modules argument.

`items`	the items that belong to the selected module (identified by their number in the item bank).
`par`	the matrix of item parameters of the selected items (one row per item).
`info`	either the provisional ability level or score when `cutoff` is not `NULL`; or `NA` when random selection is performed.
`criterion`	the value of the `criterion` argument.
`best.module`	logical value indicating whether the optimal module was eventually returned or not.

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.

See Also

genPolyMatrix, MWMI, MKL

Examples

```
## Dichotomous models ##

 # Generation of an item bank under 2PL, made of 7 successive modules that target
 # different average ability levels and with different lengths
 # (the first generated item parameters hold two modules of 8 items each)
 it <- rbind(genDichoMatrix(16, model = "2PL"),
             genDichoMatrix(6, model = "2PL", bPrior = c("norm", -1, 1)),
             genDichoMatrix(6, model = "2PL", bPrior = c("norm", 1, 1)),
             genDichoMatrix(9, model = "2PL", bPrior = c("norm", -2, 1)),
             genDichoMatrix(9, model = "2PL", bPrior = c("norm", 0, 1)),
             genDichoMatrix(9, model = "2PL", bPrior = c("norm", 2, 1)))
 it <- as.matrix(it)

 # Creation of the 'modules' matrix to list item membership in each module
 modules <- matrix(0, 55, 7)
 modules[1:8, 1] <- modules[9:16, 2] <- modules[17:22, 3] <- 1
 modules[23:28, 4] <- modules[29:37, 5] <- modules[38:46, 6] <- 1
 modules[47:55, 7] <- 1

 # Creation of the transition matrix to define a 1-2-3 MST
trans <- matrix(0, 7, 7)
```

```
trans[1, 3:4] <- trans[2, 3:4] <- trans[3, 5:7] <- trans[4, 5:7] <- 1

# Module 1 previously administered, provisional ability 0, MFI criterion
nextModule(it, modules, trans, current.module = 1, out = 1:8)

# Generation of item responses for module 1
x <- genPattern(0, it[1:8,])

# MLWMI criterion
nextModule(it, modules, trans, current.module = 1, out = 1:8, x = x, criterion = "MLWMI")

# MPWMI criterion
nextModule(it, modules, trans, current.module = 1, out = 1:8, x = x, criterion = "MPWMI")

# MKL criterion
nextModule(it, modules, trans, current.module = 1, out = 1:8, x = x, criterion = "MKL")

# MKLP criterion
nextModule(it, modules, trans, current.module = 1, out = 1:8, x = x, criterion = "MKLP")

# Creation of cut-off scores for ability levels: cut score 0 between modules 3 and 4
# and cut scores -1 and 1 between modules 5, 6 and 7
cut <- rbind(c(3, 4, 0), c(5, 6, -1), c(6, 7, 1))

# Selection by cut-off score, module 1 previously administered, current ability level 0
# (=> module 4 is chosen)
nextModule(it, modules, trans, current.module = 1, out = 1:8, cutoff = cut, theta = 0)

# Same with current ability level -0.5 (=> module 3 is chosen)
nextModule(it, modules, trans, current.module = 1, out = 1:8, cutoff = cut, theta = -0.5)

# Modules 1 and 3 previously administered, current ability level 0 (=> module 6 is chosen)
nextModule(it, modules, trans, current.module = 3, out = c(1:8, 17:22), cutoff = cut,
          theta = 0)

# Same with current ability level 2 (=> module 7 is chosen)
nextModule(it, modules, trans, current.module = 3, out = c(1:8, 17:22), cutoff = cut,
          theta = 2)

# Ranomesque probability 0.5 and random.seed value 2 (=> module 6 is chosen)
nextModule(it, modules, trans, current.module = 3, out = c(1:8, 17:22), cutoff = cut,
          theta = 2, randomesque = 0.5, random.seed = 2)

# Creation of cut-off scores for test scores: cut score 4 between modules 3 and 4
# and cut scores 5 and 9 between modules 5, 6 and 7
cut.score <- rbind(c(3, 4, 4), c(5, 6, 5), c(6, 7, 9))

# Module 1 previously administered, current test score 1 (=> module 3 is chosen)
nextModule(it, modules, trans, current.module = 1, out = 1:8, cutoff = cut.score,
          theta = 1)

# Modules 1 and 3 previously administered, current tes score 6 (=> module 6 is chosen)
nextModule(it, modules, trans, current.module = 3, out = c(1:8, 17:22), cutoff = cut.score,
```

```
          theta = 6)


## Polytomous models ##

 # Same structure as above but parameters are now generated from PCM with at most
 # 4 categories
 it.pol <- genPolyMatrix(55, model = "PCM", nrCat = 4)
 it.pol <- as.matrix(it)

 # Module 1 previously administered, provisional ability 0, MFI criterion
 nextModule(it.pol, modules, trans, model = "PCM", current.module = 1, out = 1:8)

 # MLWMI criterion
 nextModule(it.pol, modules, trans, model = "PCM", current.module = 1, out = 1:8, x = x,
           criterion = "MLWMI")

# MKL criterion
 nextModule(it.pol, modules, trans, model = "PCM", current.module = 1, out = 1:8, x = x,
           criterion = "MKL")

# MKLP criterion
 nextModule(it.pol, modules, trans, model = "PCM", current.module = 1, out = 1:8, x = x,
           criterion = "MKLP")

# Selection by cut-off score, module 1 previously administered, current ability level 0
 # (=> module 4 is chosen)
 nextModule(it.pol, modules, trans, model = "PCM", current.module = 1, out = 1:8,
           cutoff = cut, theta = 0)
```

`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 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. URL http://www.jstatsoft.org/v48/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 ##

 # Generation of an item bank under 3PL with 100 items
```

```
 m.3PL <- genDichoMatrix(100, model = "3PL")
 m.3PL <- as.matrix(m.3PL)

 # Response probabilities and derivatives (various th and D values)
 Pi(th = 0, m.3PL)
 Pi(th = 0, m.3PL, D = 1.702)
 Pi(th = 1, m.3PL)

## 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")

 # Generation of a item bank under PCM with 20 items and at most 3 categories
 m.PCM <- genPolyMatrix(20, 3, "PCM")
 m.PCM <- as.matrix(m.PCM)

 # Computation of probabilities and derivatives for ability level 1
 Pi(1, m.PCM, model = "PCM")
```

---

randomMST *Random generation of multistage tests (dichotomous and polytomous models)*

Description

This command generates a response pattern to a multistage test, for a given item bank (with either dichotomous or polytomous models), an MST structure for modules and stages, a true ability level, and several lists of MST parameters.

Usage

```
randomMST(trueTheta, itemBank, modules, transMatrix, model = NULL,
  responses = NULL, genSeed = NULL, start = list(fixModule = NULL, seed = NULL,
  theta = 0, D = 1), test = list(method = "BM", priorDist = "norm",
  priorPar = c(0, 1), range = c(-4, 4), D = 1, parInt = c(-4, 4, 33),
  moduleSelect = "MFI", constantPatt = NULL, cutoff = NULL, randomesque = 1,
  random.seed = NULL, score.range = "all"), final = list(method = "BM",
  priorDist = "norm", priorPar = c(0, 1), 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 'mst'
print(x, ...)
## S3 method for class 'mst'
```

```
plot(x, show.path = TRUE, border.col = "red", arrow.col = "red",
 module.names = 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. See **Details**.
modules	a binary matrix that specifies the item membership to the modules. See **Details**.
transMatrix	a binary squared matrix representing the structure of the MST and the transitions between the moduels and the stages. 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**.
genSeed	either a numeric value to fix the random generation of responses pattern or NULL (default). Ignored if responses is not NULL. See **Details**.
start	a list with the options for starting the multistage test. See **Details**.
test	a list with the options for provisional ability estimation and next module selection. See **Details**.
final	a list with the options for final ability estimation or scoring. See **Details**.
allTheta	logical: should all provisional ability estimates and standard errors be computed and returned (even within each module)? Default is FALSE, meaning that provisional ability estimates and standard errors are computed only at the end of each module administration.
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	either an object of class "mst", typically an output of randomMST function, or a transition matrix (for plot.mst() function only).
show.path	logical: should the selected path (i.e. set of successive modules) be highlighted in the plot (default is TRUE)?
border.col	character: the color for the rectangle border of the path (i.e. selected modules). Default is "red". Ignored if show.path is FALSE.
arrow.col	character: the color for the connecting arrows in the path (i.e. between selected modules). Default is "red". Ignored if show.path is FALSE.
module.names	either NULL (default) or a vector of character names for the modules. See **Details**.
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 `randomMST` function generates a multistage test using an item bank specified by arguments `itemBank` and `model`, an MST structure provided by arguments `modules` and `transMatrix`, 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). See `genDichoMatrix` for further information.

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.

The `modules` argument must be a binary 0/1 matrix with as many rows as the item bank `itemBank` and as many columns as the number of modules. Values of 1 indicate to which module(s) the items belong to, i.e. a value of 1 on row *i* and column *j* means that the *i*-th item belongs to the *j*-th module.

The `transMatrix` argument must be a binary 0/1 square matrix with as many rows (and columns) as the number of modules. All values of 1 indicate the possible transitions from one module to another, i.e. a value of 1 on row *i* and column *j* means that the MST can move from *i*-th module to *j*-th module.

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 `randomMST` will usually lead to different response patterns).

It is possible, however, to provide a full response pattern of previously recorded responses to each item 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.

The test specification is made by means of three lists of options: one list for the selection of the starting module, one list with the options for provisional ability estimation and next module selection, and one list with the options for final ability estimation. These lists are specified respectively by the arguments `start`, `test` and `final`.

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

- `fixModule`: either an integer value, setting the module to be administered as first stage (as its row number in te `transMatrix` argument for instance), or `NULL` (default) to let the function select the module.
- `seed`: either a numeric value to fix the random seed for module selection, `NA` to randomly select the module withour fixing the random seed, or `NULL` (default) to make random module selection without fixing the random seed. Ignored if `fixModule` is not `NULL`.
- `theta`: the initial ability value, used to select the most informative module at this ability level (default is 0). Ignored if either `fixModule` or `seed` is not `NULL`. See `startModule` 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"`.

These arguments are passed to the function `startModule` to select the first module of the multistage test.

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

- `method`: a character string to specify the method for ability estimation or scoring. 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), and `"WL"` for weighted likelihood estimation (Warm, 1989). The `method` argument can also take the value `"score"`, meaning that module selection is based on the test score from the previously administered modules. The latter works only if `cutoff` argument is supplied appropriately, otherwise this leads to an error message.
- `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 comoutation of MLWI and MPWI values for next item selection (see `nextModule` for further details).
- `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 `"MLMWI"` nor `"MPMWI"`.

- `moduleSelect`: the rule for next module selecion, with possible values:
  1. `"MFI"` (default) for maximum Fisher information criterion;
  2. `"MLMWI"` for maximum likelihood weighted (module) information criterion;
  3. `"MPMWI"` for posterior weighted (module) information criterion;
  4. `"MKL"` for (module) Kullback-Leibler information methods;
  5. `"MKLP"` for posterior (module) Kullback-Leibler information methods;
  6. `"random"` for random selection.

  This argument is ignored if `cutoff` is supplied appropriately. See `nextModule` for further details.
- `constantPatt`: 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"` (for fixed stepsize adjustment with step 0.4), `"fixed7"` (for fixed stepsize adjustment with step 0.7), or `"var"` (for variable stepsize adjustment). This is currently implemented only for dichotomous IRT models and is sgnored if `method` is `"score"`. See `thetaEst` for further details.
- `cutoff`: either `NULL` (default) or a suitable matrix of thresholds to select the next module. Thresholds can reflect module selection based on ability estimation (and then `method` should hold one of the ability estimation methods) or on provisional test score (and then `method` must be set to `"score"`. See `nextModule` for further details about suitable definition of the `cutoff` matrix (and the examples below).
- `randomesque`: a probability value to select the optimal module. Default is one, so the optimal module is always chosen. With a value smaler than one, other elligible modules can be selected.
- `random.seed`: either `NULL` (default) or a numeric value to fix the random seed of randomesque selection of the module. Ignored if `randomesque` is equal to one.
- `score.range`: a character value that specifies on which set of modules the provisional test score should be computed. Possible values are `"all"` (default) to compute the score with all previously administered modules, or `"last"` to compute the score only with the last module. Ignored if `method` is not `"score"`.

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 `nextModule` to select the next module appropriately.

Finally, the `final` list can contain the arguments `method`, `priorDist`, `priorPar`, `range`, `D` and `parInt` 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.

If some arguments of these lists are missing, they are automatically set to their default value.

Usually the ability estimates and related standard errors are computed right after the full administration of each module (that is, if current module has *k* items, the *(k-1)* ability levels and standard errors from the first administered *(k-1)* are not computed). This can however be avoided by fixing the argument allTheta to TRUE (by default it is FALSE). In this case, all provisional ability estimates (or test scores) and standard errors (or NA's) are computed and returned.

The output of randomMST, as displayed by the print.mst 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.mst represents the whole MST structure with as many rectangles as there are available modules, arrows connecting all the modules according to the transMatrix structure. Each stage is displayed as one horizontal layout with stage 1 on the top and final stage at the bottom of the figure. The selected path (i.e. set of modules) is displayed on the plot when show.path is TRUE (which is the default value). Modules from the path and arrows between them are then highlighted in red (by default), and these colors can be modified by settong border.col and arrow.col arguments with appropriate color names. By default, modules are labelled as “module 1", “module 2" etc., the numbering starting from left module to right module and from stage 1 to last stage. These labels can be modified by providing a vector of character names to argument module.names. This vector must have as many components as the total number of modules and being ordered identically as described above.

Note that the MST structure can be graphically displayed by only providing (as x argument) the transition matrix of the MST. In this case, show.path argument is ignored. This is useful to represent the MST structure set by the transition matrix without running an MST simulation.

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 randomMST returns a list of class "mst" with the following arguments:

- trueTheta: the value of the trueTheta argument.
- selected.modules: a vector with the modules (identified by their position in the transition matrix) that were selected for the MST.
- items.per.module: a vector with the number of items per selected module (in the same order as in selected.modules).
- transMatrix: the value of the transMatrix 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.
- pattern: the generated (or selected) response pattern (as vector of 0 and 1 entries for dichotomous items or positive integer values for polytomous items).

thetaProv	a vector with the provisional ability estimates (or test scores if test$method is "score").
seProv	a vector with the standard errors of the provisional ability estimates (or vector of NA's if test$method is "score").
thFinal	the final ability estimate (or test score if test$method is "score").
seFinal	the standard error of the final ability estimate (or NA if test$method is "score").
ciFinal	the confidence interval of the final ability estimate (or c(NA, NA) if test$method is "score").
genSeed	the value of the genSeed argument.
startFixModule	the value of the start$fixModule argument (or its default value if missing).
startSeed	the value of the start$seed argument (or its default value if missing).
startTheta	the value of the start$theta argument (or its default value if missing).
startD	the value of the start$D argument (or its default value if missing).
startThStart	the starting ability value used for selecting the first module of the test.
startSelect	the value of the start$startSelect argument (or its default value if missing).
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$priorPar argument (or its default value if missing).
provRange	the value of the test$range argument (or its default value if missing).
provD	the value of the test$D argument (or its default value if missing)or NA if model is not NULL.
moduleSelect	the value of the test$moduleSelect argument (or its default value if missing).
constantPattern	the value of the test$constantPatt argument (or its default value if missing).
cutoff	the value of the test$cutoff argument (or its default value if missing).
randomesque	the value of the test$randomesque argument (or its default value if missing).
random.seed	the value of the test$random.seed argument (or its default value if missing).
score.range	the value of the test$score.range argument (or its default value if missing).
best.module	a vector of boolean values indicating whether the optimal modules were selected or not.
finalMethod	the value of the final$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).
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`.

`allTheta`
: either a table with all ad-interim ability estimates (even within module, in the CAT spirit) if `allTheta` is set to `TRUE`, or `NULL` if `allTheta` is set to `FALSE`.

`assigned.responses`
: the value of the `responses` argument (or its default value if missing).

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

Author(s)

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

Duanli Yan
Educational Testing Service, Princeton, USA
<dyan@ets.org>

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: 10.1016/0022-2496(69)90005-4

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.

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`, `eapEst`, `eapSem`, `genPattern`, `genDichoMatrix` , `genPolyMatrix` ,

`nextModule`

Examples

```
## Dichotomous models ##

  # Generation of an item bank under 2PL, made of 7 successive modules that target
 # different average ability levels and with different lengths
 # (the first generated item parameters hold two modules of 8 items each)
 it <- rbind(genDichoMatrix(16, model = "2PL"),
             genDichoMatrix(6, model = "2PL", bPrior = c("norm", -1, 1)),
             genDichoMatrix(6, model = "2PL", bPrior = c("norm", 1, 1)),
             genDichoMatrix(9, model = "2PL", bPrior = c("norm", -2, 1)),
             genDichoMatrix(9, model = "2PL", bPrior = c("norm", 0, 1)),
             genDichoMatrix(9, model = "2PL", bPrior = c("norm", 2, 1)))
 it <- as.matrix(it)

 # Creation of the 'modules' matrix to list item membership in each module
 modules <- matrix(0, 55, 7)
 modules[1:8, 1] <- modules[9:16, 2] <- modules[17:22, 3] <- 1
 modules[23:28, 4] <- modules[29:37, 5] <- modules[38:46, 6] <- 1
 modules[47:55, 7] <- 1

 # Creation of the transition matrix to define a 1-2-3 MST
 trans <- matrix(0, 7, 7)
 trans[1, 3:4] <- trans[2, 3:4] <- trans[3, 5:7] <- trans[4, 5:7] <- 1

 # Creation of the start list: selection by MFI with ability level 0
 start <- list(theta = 0)

 # Creation of the test list: module selection by MFI, ability estimation by WL,
 # stepsize .4 adjustment for constant pattern
 test <- list(method = "WL", moduleSelect = "MFI", constantPatt = "fixed4")

 # Creation of the final list: ability estimation by ML
 final <- list(method = "ML")

 # Random MST generation for true ability level 1 and all ad-interim ability estimates
 res <- randomMST(trueTheta = 1, itemBank = it, modules = modules, transMatrix = trans,
                  start = start, test = test, final = final, allTheta = TRUE)

 # Module selection by cut-scores for ability estimates
 # Creation of cut-off scores for ability levels: cut score 0 between modules 3 and 4
 # and cut scores -1 and 1 between modules 5, 6 and 7
 # randomesque selection with probability .8
 cut <- rbind(c(3, 4, 0), c(5, 6, -1), c(6, 7, 1))
 test <- list(method = "WL", constantPatt = "fixed4", cutoff = cut, randomesque = 0.8)
 res <- randomMST(trueTheta = 1, itemBank = it, modules = modules, transMatrix = trans,
                  start = start, test = test, final = final, allTheta = TRUE)

 # Module selection by cut-scores for test scores
 # Creation of cut-off scores for test scores: cut score 4 between modules 3 and 4
 # and cut scores 5 and 9 between modules 5, 6 and 7
 cut.score <- rbind(c(3, 4, 4), c(5, 6, 5), c(6, 7, 9))
 test <- list(method = "score", cutoff = cut.score)
```

```
final <- list(method = "score")
res <- randomMST(trueTheta = 1, itemBank = it, modules = modules, transMatrix = trans,
                 start = start, test = test, final = final, allTheta = TRUE)

# Modification of cut-scores of stage 3 to use only the last module from stage 2 (6 items):
# cut scores 2 and 4 between modules 5, 6 and 7
cut.score2 <- rbind(c(3, 4, 4), c(5, 6, 2), c(6, 7, 4))
test <- list(method = "score", cutoff = cut.score2, score.range = "last")
final <- list(method = "score")
res <- randomMST(trueTheta = 1, itemBank = it, modules = modules, transMatrix = trans,
                 start = start, test = test, final = final, allTheta = TRUE)

## Plot options
plot(trans)
plot(res)
plot(res, show.path = FALSE)
plot(res, border.col = "blue")
plot(res, arrow.col = "green")
```

---

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.

Usage

```
semTheta(thEst, it, x = NULL, model = NULL, D = 1, method = "BM",
   priorDist = "norm", priorPar = c(0, 1), parInt = c(-4, 4, 33),
   constantPatt = NULL)
```

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`). Ignored if `method` is not `"EAP"`.
`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" and "EAP". 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**.
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**.

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.

Four 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)and the weighted likelihood (WL) estimator (Warm, 1989). The selected estimator is specified by the method argument, with values "ML", "BM", "EAP" and "WL" 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.

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.

Value

The estimated standard error of the ability level.

Note

Currently the 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). Version of `catR` prior to 3.0 holds 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 currently in application until a corrected formula can be provided.

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.

Birnbaum, A. (1969). Statistical theory for logistic mental test models with a prior distribution of ability. *Journal of Mathematical Psychology, 6*, 258-276. doi: 10.1016/0022-2496(69)90005-4

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., 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. URL http://www.jstatsoft.org/v48/i08/

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

Examples

```
## Dichotomous models ##

 # Generation of an item bank under 3PL with 100 items
 m.3PL <- genDichoMatrix(100, model = "3PL")
 m.3PL <- as.matrix(m.3PL)

 # Creation of a response pattern (true ability level 0)
 set.seed(1)
 x <- genPattern(0, m.3PL)

 # ML estimation
 th <- thetaEst(m.3PL, x, method = "ML")
 c(th, semTheta(th, m.3PL, method = "ML"))

 # BM estimation, standard normal prior distribution
 th <- thetaEst(m.3PL, x)
 c(th, semTheta(th, m.3PL))

 # BM estimation, uniform prior distribution upon range [-2,2]
 th <- thetaEst(m.3PL, x, method = "BM", priorDist = "unif",
               priorPar = c(-2, 2))
 c(th, semTheta(th, m.3PL, method = "BM", priorDist = "unif",
      priorPar = c(-2, 2)))

 # BM estimation, Jeffreys' prior distribution
 th <- thetaEst(m.3PL, x, method = "BM", priorDist = "Jeffreys")
 c(th, semTheta(th, m.3PL, method = "BM", priorDist = "Jeffreys"))

 # EAP estimation, standard normal prior distribution
 th <- thetaEst(m.3PL, x, method = "EAP")
 c(th, semTheta(th, m.3PL, x, method = "EAP"))

 # EAP estimation, uniform prior distribution upon range [-2,2]
 th <- thetaEst(m.3PL, x, method = "EAP", priorDist = "unif",
               priorPar = c(-2, 2))
 c(th, semTheta(th, m.3PL, x, method = "EAP", priorDist = "unif",
      priorPar = c(-2, 2)))

 # EAP estimation, Jeffreys' prior distribution
 th <- thetaEst(m.3PL, x, method = "EAP", priorDist = "Jeffreys")
 c(th, semTheta(th, m.3PL, x, method = "EAP", priorDist = "Jeffreys"))

 # WL estimation
 th <- thetaEst(m.3PL, x, method = "WL")
 c(th, semTheta(th, m.3PL, method = "WL"))

 # 'fixed4' adjustment for constant pattern
```

```
 th <- thetaEst(m.3PL, rep(0, nrow(m.3PL)), constantPatt = "fixed4")
 c(th, semTheta(th, m.3PL, constantPatt = "fixed4"))

## 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") )

 # 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"))


 # 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
 th <- thetaEst(m.PCM, x, model = "PCM", method = "ML")
 c(th, semTheta(th, m.PCM, model = "PCM", method = "ML"))

 # BM estimation, standard normal prior distribution
 th <- thetaEst(m.PCM, x, model = "PCM")
 c(th, semTheta(th, m.PCM, model = "PCM"))

 # BM estimation, uniform prior distribution upon range [-2,2]
 th <- thetaEst(m.PCM, x, model = "PCM", method = "BM", priorDist = "unif",
    priorPar = c(-2, 2))
 c(th, semTheta(th, m.PCM, model = "PCM", method = "BM", priorDist = "unif",
  priorPar = c(-2, 2)))

 # BM estimation, Jeffreys' prior distribution
 th <- thetaEst(m.PCM, x, model = "PCM", method = "BM", priorDist = "Jeffreys")
 c(th, semTheta(th, m.PCM, model = "PCM", method = "BM", priorDist = "Jeffreys"))

 # EAP estimation, standard normal prior distribution
 th <- thetaEst(m.PCM, x, model = "PCM", method = "EAP")
 c(th, semTheta(th, m.PCM, x, model = "PCM", method = "EAP"))

 # EAP estimation, uniform prior distribution upon range [-2,2]
 th <- thetaEst(m.PCM, x, model = "PCM", method = "EAP", priorDist = "unif",
    priorPar = c(-2, 2))
 c(th, semTheta(th, m.PCM, x, model = "PCM", method = "EAP", priorDist = "unif",
  priorPar = c(-2, 2)))

 # EAP estimation, Jeffreys' prior distribution
 th <- thetaEst(m.PCM, x, model = "PCM", method = "EAP", priorDist = "Jeffreys")
 c(th, semTheta(th, m.PCM, x, model = "PCM", method = "EAP", priorDist = "Jeffreys"))

 # WL estimation
 th <- thetaEst(m.PCM, x, model = "PCM", method = "WL")
 c(th, semTheta(th, m.PCM, model = "PCM", method = "WL"))

## End(Not run)
```

---

startModule *Selection of the first module in MST*

Description

This command selects the first module of the multistage test, either randomly or on the basis of the module information function.

Usage

```
startModule(itemBank, modules, transMatrix, model = NULL, fixModule = NULL,
  seed = NULL, theta = 0, D = 1)
```

Arguments

itemBank	a suitable matrix of item parameters. See **Details**.
modules	a binary matrix that specifies the item membership to th emodules. See **Details**.
transMatrix	a binary squared matrix representing the structure of the MST and the transitions between the moduels and the stages. 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**.
fixModule	either a an integer value or NULL (default). See **Details**.
seed	either a numeric value, NA or NULL (default). Ignored if fixModule is not NULL. See **Details**.
theta	numeric: the initial ability level for selecting the first module (default is 0). Ignored if either fixModule 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.

Details

This function permits to select the first module of the MST. 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 modules argument must be a binary 0/1 matrix with as many rows as the item bank itemBank and as many columns as the number of modules. Values of 1 indicate to which module(s) the items belong to, i.e. a value of 1 on row *i* and column *j* means that the *i*-th item belongs to the *j*-th module.

The transMatrix argument must be a binary 0/1 square matrix with as many rows (and columns) as the number of modules. All values of 1 indicate the possible transitions from one module to another, i.e. a value of 1 on row *i* and column *j* means that the MST can move from *i*-th module to *j*-th module.

The first module of the MST can be selected by one of the following methods.

1. By specifying the module to be administered. The argument fixModule then holds the module number as listed in the modules or transMatrix. Setting fixModule to NULL (default value) disables this method.
2. By selecting it randomly into the modules matrix. The argument seed permits to fix the random selection by specifying the random seed number. Setting seed to NA disables the random seed (though items are still picked up randomly in the bank); in other words, successive runs of startModule with seed=NA may lead to different module selection. Setting seed to NULL (default value) disables this selection method.
3. By selecting the module according to an initial ability value. In this case, the selected module is such that the information fucntion (computed with the items of the module) is maximal for the given initial ability value.

The third method above will be used if and only if both fixModule and seed arguments are fixed to NULL. Otherwise, one of the first two methods will be used.

Value

A list with four arguments:

module	the selected module (identified by its column number in the modules argument.
items	the items that belong to the selected module (identified by their number in the item bank).
par	the matrix of item parameters of the selected items (one row per item).
thStart	the starting ability level used for selecting the module or NA (if not applicable).

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.

See Also

genPolyMatrix

Examples

```
## Dichotomous models ##

 # Generation of an item bank under 2PL, made of 7 successive modules that target
 # different average ability levels and with different lengths
 # (the first generated item parameters hold two modules of 8 items each)
 it <- rbind(genDichoMatrix(16, model = "2PL"),
             genDichoMatrix(6, model = "2PL", bPrior = c("norm", -1, 1)),
             genDichoMatrix(6, model = "2PL", bPrior = c("norm", 1, 1)),
             genDichoMatrix(9, model = "2PL", bPrior = c("norm", -2, 1)),
             genDichoMatrix(9, model = "2PL", bPrior = c("norm", 0, 1)),
             genDichoMatrix(9, model = "2PL", bPrior = c("norm", 2, 1)))
 it <- as.matrix(it)

 # Creation of the 'module' matrix to list item membership in each module
 modules <- matrix(0, 55, 7)
 modules[1:8, 1] <- modules[9:16, 2] <- modules[17:22, 3] <- 1
 modules[23:28, 4] <- modules[29:37, 5] <- modules[38:46, 6] <- 1
 modules[47:55, 7] <- 1

 # Creation of the transition matrix to define a 1-2-3 MST
trans <- matrix(0, 7, 7)
trans[1, 3:4] <- trans[2, 3:4] <- trans[3, 5:6] <- trans[4, 6:7] <- 1

 # Selection of module 2 as starting module
 startModule(it, modules, trans, fixModule = 2)

## Not run:

 # Selection of module 3 (not from stage 1 => mistake)
 startModule(it, modules, trans, fixModule = 3)

## End(Not run)

 # Random selection of starting module
 startModule(it, modules, trans, seed = 1)

 # Selection by maximizing information at ability level 0
 startModule(it, modules, trans, theta = 0)


## Polytomous models ##

 # Same structure as above but parameters are now generated from PCM with at most
 # 4 categories
 it <- genPolyMatrix(55, model = "PCM", nrCat = 4)
 it <- as.matrix(it)

 # Selection of module 2 as starting module
 startModule(it, modules, trans, fixModule = 2, model = "PCM")

## Not run:
```

```
 # Selection of module 3 (not from stage 1 => mistake)
 startModule(it, modules, trans, fixModule = 3, model = "PCM")

## End(Not run)

 # Random selection of starting module
 startModule(it, modules, trans, seed = 1, model = "PCM")

 # Selection by maximizing information at ability level 0
 startModule(it, modules, trans, theta = 0, model = "PCM")
```

---

testListMST *Testing the format of the MST input lists*

Description

This command tests whether format of the input lists for the random generation of multistage tests is convenient, and returns a warning message otherwise.

Usage

```
testListMST(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" and "final". See **Details**.

Details

The testListMST 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 randomMST function.

The three types of lists are: "start" with the parameters for selecting the first module; "test" with the options of the multistage test (i.e. method for next module selection, provisional ability estimator and related information); and "final" with the options for final ability estimation. See the help file of randomMST 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)*, 1-19. doi: 10.18637/jss.v076.c01

See Also

randomMST

Examples

```
 # Creation and test of a 'start' list
 start <- list(theta = 0)
 testListMST(start, type = "start")

 # Creation and test of a 'test' list
 test <- list(method = "WL", moduleSelect = "MFI", constantPatt = "fixed4")
 testListMST(test, type = "test")

 # Creation and test of a 'final' list (with mistake)
 final <- list(method = "MAP")
 testListMST(final, type = "final")

# Creation of cut-off scores for ability levels: cut score 0 between modules 3 and 4
 # and cut scores -1 and 1 between modules 5, 6 and 7
 cut <- matrix(NA, 7, 2)
 cut[3,] <- c(-Inf, 0)
 cut[4,] <- c(0, Inf)
 cut[5,] <- c(-Inf, -1)
 cut[6,] <- c(-1, 1)
```

```
cut[7,] <- c(1, Inf)
test <- list(method = "WL", constantPatt = "fixed4", cutoff = cut)
testListMST(test, "test")
```

---

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, Bayes modal (MAP), expected a posteriori (EAP) and weighted likelihood.

Usage

```
thetaEst(it, x, model = NULL, D = 1, method = "BM", priorDist = "norm",
  priorPar = c(0, 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.
`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"` and `"EAP"`. 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**.
`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.

Four 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) and the weighted likelihood (WL) estimator (Warm, 1989). The selected estimator is specified by the method argument, with values "ML", "BM", "EAP" and "WL" 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.

The range argument permits to limit the interval of investigation for the ML, BM and WL 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

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.

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.

Birnbaum, A. (1969). Statistical theory for logistic mental test models with a prior distribution of ability. *Journal of Mathematical Psychology, 6*, 258-276. doi: 10.1016/0022-2496(69)90005-4

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/S11336-013-9378-5

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. URL http://www.jstatsoft.org/v48/i08/

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 ##

 # Generation of an item bank under 3PL with 100 items
 m.3PL <- genDichoMatrix(100, model = "3PL")
 m.3PL <- as.matrix(m.3PL)


 # Creation of a response pattern (true ability level 0)
 set.seed(1)
 x <- genPattern(0, m.3PL)

 # ML estimation
 thetaEst(m.3PL, x, method = "ML")

 # BM estimation, standard normal prior distribution
 thetaEst(m.3PL, x)

 # BM estimation, uniform prior distribution upon range [-2,2]
 thetaEst(m.3PL, x, method = "BM", priorDist = "unif", priorPar = c(-2, 2))

 # BM estimation, Jeffreys' prior distribution
 thetaEst(m.3PL, x, method = "BM", priorDist = "Jeffreys")

 # EAP estimation, standard normal prior distribution
 thetaEst(m.3PL, x, method = "EAP")

 # EAP estimation, uniform prior distribution upon range [-2,2]
 thetaEst(m.3PL, x, method = "EAP", priorDist = "unif", priorPar = c(-2, 2))

 # EAP estimation, Jeffreys' prior distribution
 thetaEst(m.3PL, x, method = "EAP", priorDist = "Jeffreys")

 # WL estimation
 thetaEst(m.3PL, x, method = "WL")

 # Creation of two constant patterns and estimation with WL,
 # 'fixed4', 'fixed7' and 'var' stepsize adjustments
 x0 <- rep(0,nrow(m.3PL))
 x1 <- x0 + 1
 thetaEst(m.3PL, x0, constantPatt = "WL") # equivalent to thetaEst(m.3PL, x0, method = "WL")
 thetaEst(m.3PL, x1, constantPatt = "WL") # equivalent to thetaEst(m.3PL, x1, method = "WL")
 thetaEst(m.3PL, x0, constantPatt = "fixed4")
 thetaEst(m.3PL, x1, constantPatt = "fixed4")
 thetaEst(m.3PL, x0, constantPatt = "fixed7")
```

```
 thetaEst(m.3PL, x1, constantPatt = "fixed7")
 thetaEst(m.3PL, x0, constantPatt = "var")
 thetaEst(m.3PL, 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