

Package ‘irtoys’

October 12, 2009

Type Package

Title Simple interface to the estimation and plotting of IRT models

Version 0.1.2

Date 2009-10-11

Author Ivailo Partchev

Maintainer Ivailo Partchev <Ivailo.Partchev@uni-jena.de>

Description Provides a simple common interface to the estimation of item parameters in IRT models for binary responses with three different programs (ICL, BILOG-MG, and ltm, and a variety of functions useful with IRT models.

License GPL (>= 2)

Depends sm, ltm

Suggests MASS

Repository CRAN

Date/Publication 2009-10-12 06:48:48

R topics documented:

irtoys-package	2
api	3
dpv	4
eap	5
est	6
grp	8
iif	9
internal	10
irf	10
itf	11
l2m	14

mlebm	14
normal.qu	15
npp	16
plot.iif	17
plot.irf	18
plot.tif	19
plot.trf	20
qrs	21
read.ip.bilog	22
read.ip.icl	22
read.qu.icl	23
read.resp	24
sca	25
sco	26
Scored	27
sim	27
tgp	28
tif	29
trf	30
tsc	31
Unscored	32

Index	33
--------------	-----------

irtoys-package *Estimate and plot IRT models for binary responses*

Description

Provides a common interface to the estimation of item parameters in IRT models for binary responses with three different programs (ICL, BILOG, and `ltm`, and a variety of functions useful with IRT models.

Details

The `irtoys` package contains a bunch of functions potentially useful to those teaching or learning Item Response Theory (IRT). Although there is no shortage of good IRT programs, those tend to have wildly different and often unwieldy user interfaces. Besides, no single program does everything one needs. Item parameters can be estimated with a program like ICL or BILOG, non-parametric approaches are implemented in TestGraf, transformation to a common scale needs ST, and so on. Some programs, such as ICL, have no graphical capabilities at all, while others offer stunning interactive graphics but refuse to output a Postscript file.

Package `irtoys` provides a common interface to some of the most basic functions in ICL, BILOG, and R's own `ltm`, some of the functionality of TestGraf and ST, and a variety of other functions. Those who want to take advantage of the full functionality of ICL, BILOG & Co. must still master their syntax.

To take full advantage of `irttoys`, some IRT software is needed. Package `ltm` is automatically loaded. ICL by Brad Hanson can be downloaded from his site, www.b-a-h.com: executables are provided for Windows, Linux, and Macintosh. BILOG is commercial software sold by SSI — see www.ssicentral.com for further detail.

On Windows, make sure that the executable files (`icl.exe` for ICL, `BLM1.EXE`, `BLM2.EXE`, and `BLM3.EXE` for BILOG) are located in a directory that is included in the `PATH` variable. On Linux, the default directory for the BILOG files is now `~/ .wine/drive_c/`; because BILOG is a Windows program, `wine` must be installed. On Macintosh, at least `ltm` should work.

Author(s)

Ivailo Partchev <Ivailo.Partchev@uni-jena.de>

References

S. E. Embretson and S. P. Reise (2000), *Item Response Theory for Psychologists*, Lawrence Erlbaum Associates, Mahwah, NJ

api

The Z3 appropriateness index

Description

Computes the Z3 appropriateness index, a measure of person fit in IRT models

Usage

```
api(resp, ip)
```

Arguments

<code>resp</code>	A matrix of responses: persons as rows, items as columns, entries are either 0 or 1, no missing data
<code>ip</code>	Item parameters: a matrix with one row per item, and three columns: [,1] item discrimination a , [,2] item difficulty b , and [,3] asymptote c .

Value

A vector of length equal to the number of rows in `resp`, containing the appropriateness indices

Author(s)

Ivailo Partchev

References

Drasgow, F., Levine, M. V., & Williams, E. A. (1985). Appropriateness measurement with polychotomous item response models and standardized indices. *British Journal of Mathematical and Statistical Psychology*, 38, 67–80

Examples

```
data(Scored)
p.2pl <- est(Scored, model="2PL", engine="ltm")
api(Scored, p.2pl)
```

dpv

Draw plausible values

Description

Draws (by rejection sampling) plausible values from the posterior distribution of ability

Usage

```
dpv(resp, ip, mu=0, sigma=1, n=5)
```

Arguments

resp	A matrix of responses: persons as rows, items as columns, entries are either 0 or 1, no missing data
ip	Item parameters: a matrix with one row per item, and three columns: [,1] item discrimination a , [,2] item difficulty b , and [,3] asymptote c .
mu	Mean of the apriori distribution. Ignored when <code>method="ML"</code> . Default is 0.
sigma	Standard deviation of the apriori distribution. Ignored when <code>method="ML"</code> . Default is 1.
n	The number of plausible values to draw for each person (default is 5).

Value

A matrix with `n` columns

Author(s)

Ivailo Partchev

See Also

[mlebbe](#), [eap](#)

Examples

```
data(Scored)
p.2pl <- est(Scored, model="2PL", engine="ltm")
plval <- dpv(resp=Scored, ip=p.2pl)
```

eap *EAP estimation of ability*

Description

Estimates the expectation of the posterior distribution of the latent variable ("ability") for each person.

Usage

```
eap(resp, ip, qu)
```

Arguments

<code>resp</code>	A matrix of responses: persons as rows, items as columns, entries are either 0 or 1, no missing data
<code>ip</code>	Item parameters: a matrix with one row per item, and three columns: [,1] item discrimination a , [,2] item difficulty b , and [,3] asymptote c .
<code>qu</code>	A quadrature object produced with <code>normal.qu</code> or read in with <code>read.qu.icl</code>

Value

A matrix with the ability estimates in column 1, and their standard errors of measurement (SEM) in column 2, and the number of non-missing responses in column 3

Author(s)

Ivailo Partchev

See Also

[mlebme](#), [normal.qu](#), [read.qu.icl](#)

Examples

```
data(Scored)
p.2pl <- est(Scored, model="2PL", engine="ltm")
th.eap <- eap(resp=Scored, ip=p.2pl, qu=normal.qu())
```

est

*Estimate item parameters***Description**

Estimate IRT item parameters using either ICL, BILOG, or ltm. Provides access to the most widely used options in these programs.

Usage

```
est(resp, model = "2PL", engine = "icl", nqp = 20, est.distr = FALSE,
    logistic = TRUE, nch = 5, a.prior = TRUE, b.prior = FALSE, c.prior = TRUE,
    bilog.defaults = TRUE, rasch = FALSE, run.name = "mymodel")
```

Arguments

resp	A matrix of responses: persons as rows, items as columns, entries are either 0 or 1, no missing data
model	The IRT model: "1PL", "2PL", or "3PL". Default is "2PL".
engine	One of "icl", "bilog", or "ltm". Default is "icl".
nqp	Number of quadrature points. Default is 20.
est.distr	T if the probabilities of the latent distribution are to be estimated, F if a normal distribution is assumed. Default is F. Ignored when engine="ltm".
logistic	logistic=T sets the constant D in the IRT model to 1 ("logistic metric"), logistic=F sets it to 1.7. Default is T. When engine="ltm" and logistic=T, the estimated item discriminations are simply divided by 1.7.
nch	Number of choices in the original item formulation. Used to determine the prior for the asymptote when engine="bilog", model="3PL", and c.prior=T. Default is 5.
a.prior	Whether a prior for the item discriminations is used. Ignored when model="1PL" or engine="ltm". Default is T.
b.prior	Whether a prior for the item difficulties is used. Ignored when engine="ltm". Default is F.
c.prior	Whether a prior for the asymptotes is used. Ignored when model="1PL" or model="2PL" or engine="ltm". Default is T.
bilog.defaults	When engine="icl" and a prior is used, use the default priors in BILOG rather than the default priors in ICL. Ignored when engine="ltm". Default is T.
rasch	When engine="bilog" and model="1PL" and "rasch"=T, the common value for discriminations is forced to 1, and the sum of the difficulties is 0. When engine="ltm" and model="1PL" and "rasch"=T, the common value for discriminations is forced to 1. Ignored in all other cases. Default is F.
run.name	A (short) string used in the names of all files read or written by ICL or BILOG. Default is "mymodel". Change to something else to keep the outputs of ICL or BILOG for further use. Ignored when engine="ltm"

Details

Estimate the parameters of an IRT model defined in the most general case ("3PL") as

$$P(U_{ij} = 1|\theta_i, a_j, b_j, c_j) = c_j + (1 - c_j) \frac{\exp(Da_j(\theta_i - b_j))}{1 + \exp(Da_j(\theta_i - b_j))}$$

where U_{ij} is a binary response given by person i to item j , θ_i is the value of the latent variable ("ability") for person i , a_j is the discrimination parameter for item j , b_j is the difficulty parameter for item j , c_j is the asymptote for item j , and D is a constant usually set to either 1.7 or 1.

In the 2PL model (`model="2PL"`), all asymptotes c_j are 0. In the 1PL model (`model="1PL"`), all asymptotes c_j are 0 and the discriminations a_j are equal for all items (and sometimes to 1).

Package `irtoys` provides a simple common interface to the estimation of item parameters with three different programs. It only accesses the most basic and widely used options in these programs. Each of the three programs has a much wider choice of options and capabilities, and serious users must still learn the corresponding syntax in order to access the advanced features. Even when models are fit "by hand", `irtoys` may be useful in plotting results, doing comparisons across programs etc.

Estimation of the more complex IRT models (2PL and 3PL) for some "difficult" data sets often has to use prior distributions for the item parameters. `irtoys` adopts the default behaviour of BILOG: no priors for b in any model, priors for a in the 2PL and 3PL models, priors for c in the 3PL model. This can be overridden by changing the values of `a.prior`, `b.prior`, and `c.prior`.

If priors are used at all, they will be the same for all items. Note that both ICL and BILOG can, at some additional effort, set different priors for any individual item. At default, the common priors are the BILOG defaults: `normal(0, 2)` for b , `lognormal(0, 0.5)` for a , and `beta(20*p+1, 20(1-p)+1)` for c ; p is 1 over the number of choices in the original item formulations, which can be set with the parameter `nch`, and is again assumed the same for all items.

When `engine="icl"` and `bilog.defaults=F`, any priors used will be the ICL default ones, and based on the 4-parameter beta distribution: `beta(1.01, 1.01, -6, 6)` for b , `beta(1.75, 3, 0, 3)` for a , and `beta(3.5, 4, 0, 0.5)` for c . When `engine="ltm"`, all commands involving priors are ignored.

`est` only works when some IRT software is installed. Package `ltm` is automatically loaded. ICL can be downloaded from www.b-a-h.com. BILOG is commercial software sold by SSI — see www.ssicentral.com for further detail. On Windows, make sure that the executable files (`icl.exe` for ICL, `blm1.exe`, `blm2.exe`, and `blm3.exe`, for BILOG) are located in directories that are included in the `PATH` variable.

Value

A matrix with one row per item, and three columns: [,1] item discrimination a , [,2] item difficulty b , and [,3] asymptote c . For the 1PL and 2PL models, all asymptotes are equal to 0; for the 1PL, the discriminations are all equal but not necessarily equal to 1.

Author(s)

Ivailo Partchev

References

- Bradley A. Hanson (2002), ICL: IRT Command Language. www.b-a-h.com
- Dimitris Rizopoulos (2006). ltm: Latent Trait Models under IRT. cran.r-project.org
- M. F. Zimowski, E. Muraki, R. J. Mislevy and R. D. Bock (1996), BILOG–MG. Multiple-Group IRT Analysis and Test Maintenance for Binary Items, SSI Scientific Software International, Chicago, IL. www.ssicentral.com

Examples

```
data(Scored)
p.1pl <- est(Scored, model="1PL", engine="ltm")
p.2pl <- est(Scored, model="2PL", engine="ltm")
```

 grp

Bin the latent variable estimates

Description

Groups a vector of estimates of the latent variable into a histogram-like object. Typically invoked by function `itf`, unless the user wishes to access some of the non-default settings.

Usage

```
grp(theta, bins=9, breaks=NULL, equal="count", type="meds")
```

Arguments

<code>theta</code>	The latent variable estimates to be binned
<code>bins</code>	Desired number of bins
<code>breaks</code>	A vector of cutpoints. Overrides <code>bins</code> if present.
<code>equal</code>	Either "width" for bins of equal width, or "count" for bins with roughly counts of observations. Default is "quant"
<code>type</code>	The points at which <code>itf</code> will evaluate the IRF. One of "mids" (the mid-point of each bin), "meds" (the median of the values in the bin), or "means" (the mean of the values in the bin). Default is "meds".

Value

A list of:

<code>breaks</code>	The breaks between adjacent bins
<code>counts</code>	The number of values in each bin
<code>ref</code>	Depending on <code>ref.type</code> , the mids of the bin, or the mean or median of the values in the bin

Author(s)

Ivailo Partchev

See Also[itf](#)**Examples**

```
data(Scored)
p.2pl <- est(Scored, model="2PL", engine="ltm")
th.mle <- mlebme(resp=Scored, ip=p.2pl)
gr <- grp(th.mle, bins=7)
```

iif

*Item information function***Description**

The item information function (IIF) for the 3PL model can be computed as

$$I(\theta) = a^2 \frac{Q(\theta)}{P(\theta)} \left[\frac{P(\theta) - c}{1 - c} \right]^2,$$

where θ is the value of the latent variable for a person, a is the discrimination parameter for the item, P is the IRF for the person and item, and $Q = 1 - P$. For the 1PL and 2PL models, the expression reduces to $a^2 PQ$.

A common use of this function would be to obtain a plot of the IIF.

Usage

```
iif(ip, x = NULL)
```

Arguments

ip	Item parameters: a matrix with one row per item, and three columns: [,1] item discrimination a , [,2] item difficulty b , and [,3] asymptote c .
x	The values of the latent variable (θ in the equation above), at which the IIF will be evaluated. If not given, 99 values spaced evenly between -4 and +4 will be used, handy for plotting.

Value

A list of:

x	A copy of the argument x
f	A matrix containing the IIF values: persons (values of (x) as rows and items as columns

Author(s)

Ivailo Partchev

See Also`plot.iif, irf`**Examples**

```
data(Scored)
p.2pl <- est(Scored, model="2PL", engine="ltm")
plot(iif(p.2pl[1:3,]))
```

`internal`*Internal functions*

Description

These functions are for internal use.

Author(s)

Ivailo Partchev

`irf`*Item response function*

Description

Returns the item response function of the 3PL (1PL, 2PL) model, the i.e. the probabilities defined by

$$P(U_{ij} = 1|\theta_i, a_j, b_j, c_j) = c_j + (1 - c_j) \frac{\exp(Da_j(\theta_i - b_j))}{1 + \exp(Da_j(\theta_i - b_j))}$$

where U_{ij} is a binary response given by person i to item j , θ_i is the value of the latent variable ("ability") for person i , a_j is the discrimination parameter for item j , b_j is the difficulty parameter for item j , c_j is the asymptote for item j , and D is a constant usually set to either 1.7 or 1. Some authors call the IRF "the item characteristic curve".

In the 2PL model (`model="2PL"`), all asymptotes c_j are 0. In the 1PL model (`model="1PL"`), all asymptotes c_j are 0 and the discriminations a_j are equal for all items (and sometimes to 1).

A common use of this function would be to obtain a plot of the IRF.

Usage

```
irf(ip, x = NULL)
```

Arguments

<code>ip</code>	Item parameters: a matrix with one row per item, and three columns: [,1] item discrimination a , [,2] item difficulty b , and [,3] asymptote c .
<code>x</code>	The values of the latent variable (θ in the equation above), at which the IRF will be evaluated. If not given, 99 values spaced evenly between -4 and +4 will be used, handy for plotting.

Value

A list of:

<code>x</code>	A copy of the argument <code>x</code>
<code>f</code>	A matrix containing the IRF values: persons (values of (<code>x</code>)) as rows and items as columns

Author(s)

Ivailo Partchev

See Also

[plot.irf](#)

Examples

```
data(Scored)
p.2pl <- est(Scored, model="2PL", engine="ltm")
plot(irf(p.2pl[1,]))
```

`itf`

Test item fit

Description

Returns a statistic of item fit together with its degrees of freedom and p-value. Optionally produces a plot.

Usage

```
itf(resp, ip, item, stat = "lr", theta, groups,
     standardize=TRUE, mu=0, sigma=1, do.plot = TRUE, main = "Item fit")
```

Arguments

<code>resp</code>	A matrix of responses: persons as rows, items as columns, entries are either 0 or 1, no missing data
<code>ip</code>	Item parameters: a matrix with one row per item, and three columns: [,1] item discrimination a , [,2] item difficulty b , and [,3] asymptote c .
<code>item</code>	A single number pointing to the item (column of <code>resp</code> , row of <code>ip</code>), for which fit is to be tested
<code>stat</code>	The statistic to be computed, either "chi" or "lr". Default is "lr". See details below.
<code>theta</code>	A vector containing some viable estimate of the latent variable for the same persons whose responses are given in <code>resp</code> . If not given (and <code>group</code> is also missing), EAP estimates will be computed from <code>resp</code> and <code>ip</code> .
<code>groups</code>	An object produced by function <code>grp</code> . If not given, <code>grp</code> will be applied on <code>theta</code> with its default values.
<code>standardize</code>	Standardize the distribution of ability estimates?
<code>mu</code>	Mean of the standardized distribution of ability estimates
<code>sigma</code>	Standard deviation of the standardized distribution of ability estimates
<code>do.plot</code>	Whether to do a plot
<code>main</code>	The title of the plot if one is desired

Details

Given a long test, say 20 items or more, a large-test statistic of item fit could be constructed by dividing examinees into groups of similar ability, and comparing the observed proportion of correct answers in each group with the expected proportion under the proposed model. Different statistics have been proposed for this purpose.

The chi-squared statistic

$$X^2 = \sum_g (N_g \frac{(p_g - \pi_g)^2}{\pi_g(1 - \pi_g)}),$$

where N_g is the number of examinees in group g , $p_g = r_g/N_g$, r_g is the number of correct responses to the item in group g , and π_g is the IRF of the proposed model for the median ability in group g , is attributed by Embretson & Reise to R. D. Bock, although the article they cite does not actually mention it. The statistic is the sum of the squares of quantities that are often called "Pearson residuals" in the literature on categorical data analysis.

BILOG uses the likelihood-ratio statistic

$$X^2 = 2 \sum_g \left[r_g \log \frac{p_g}{\pi_g} + (N_g - r_g) \log \frac{(1 - p_g)}{(1 - \pi_g)} \right],$$

where π_g is now the IRF for the mean ability in group g , and all other symbols are as above.

Both statistics are assumed to follow the chi-squared distribution with degrees of freedom equal to the number of groups minus the number of parameters of the model (eg 2 in the case of the 2PL model). The first statistic is obtained in `itf` with `stat="chi"`, and the second with `stat="lr"` (or not specifying `stat` at all).

In the real world we can only work with estimates of ability, not with ability itself, so the approach is a bit circular in defining the groups. I have tried to offer some extra flexibility with the arguments `theta` nor `group`:

- if neither `theta` nor `group` is specified, `item.test` will compute EAP estimates of ability for the proposed model, `group` them, and use medians for "chi" or means for "lr". This is the approximate behaviour of BILOG (assuming `stat="lr"`).
- the EAP abilities can be overridden by computing some other ability estimates, or even the rank quantiles produced by `qrs` and passing them to `item.test` as `theta`.
- the default grouping (medians for "chi", means for "lr") can be overridden by preparing the groups with `grp` and passing them to `item.test` as `group`. In that case, `theta` is not needed.

If the test has less than 20 items, `item.test` will issue a warning. For tests of 10 items or less, BILOG has a special statistic of fit, which can be found in the BILOG output. Also of interest is the fit in 2- and 3-way marginal tables in package `ltm`.

Value

A list of:

<code>statistic</code>	The value of the statistic of item fit
<code>dfr</code>	The degrees of freedom
<code>pvalue</code>	The p-value

Author(s)

Ivailo Partchev

References

S. E. Embretson and S. P. Reise (2000), *Item Response Theory for Psychologists*, Lawrence Erlbaum Associates, Mahwah, NJ

M. F. Zimowski, E. Muraki, R. J. Mislevy and R. D. Bock (1996), *BILOG-MG. Multiple-Group IRT Analysis and Test Maintenance for Binary Items*, SSI Scientific Software International, Chicago, IL

See Also

[grp](#), [eap](#), [qrs](#)

Examples

```
data(Scored)
p.2pl <- est(Scored, model="2PL", engine="ltm")
fit <- itf(resp=Scored, ip=p.2pl, item=7)
```

 l2m

Transform a list of lists... to a matrix

Description

When using `lapply` or `sapply` to avoid explicit loops, one often ends up with complex structures representing lists of lists of lists... Function `l2m` tries to transform such a structure into a matrix with a column for each distinct name in the list.

Usage

```
l2m(x)
```

Arguments

`x` A list of lists...

Value

A matrix with a column for each distinct name in the list

Author(s)

Ivailo Partchev

mlebme

Maximum likelihood and Bayes Modal estimation of ability

Description

Estimates the value of the latent variable ("ability") for each person by direct optimization

Usage

```
mlebme(resp, ip, mu=0, sigma=1, method="ML")
```

Arguments

`resp` A matrix of responses: persons as rows, items as columns, entries are either 0 or 1, no missing data

`ip` Item parameters: a matrix with one row per item, and three columns: [,1] item discrimination a , [,2] item difficulty b , and [,3] asymptote c .

`mu` Mean of the apriori distribution. Ignored when `method="ML"`. Default is 0.

`sigma` Standard deviation of the apriori distribution. Ignored when `method="ML"`. Default is 1.

`method` "ML" for maximum likelihood or "BM" for Bayes Modal estimation. Default is "ML", in which case any values for `mu` and `sigma` will be ignored.

Value

A matrix with the ability estimates in column 1 and their standard errors of measurement (SEM) in column 2, and the number of non-missing responses in column 3

Author(s)

Ivailo Partchev

See Also

[eap](#)

Examples

```
data(Scored)
p.2pl <- est(Scored, model="2PL", engine="ltm")
th.mle <- mlebme(resp=Scored, ip=p.2pl)
```

normal.qu

Normal quadrature points and weights

Description

Quadrature points and weights based on the Normal distribution. Quadrature objects are used when estimating abilities with `eap` and for some of the scaling methods in `sca`.

Usage

```
normal.qu(n = 15, lower = -4, upper = 4, mu = 0, sigma = 1, scaling = "points")
```

Arguments

<code>n</code>	Number of quadrature points
<code>lower</code>	Lower boundary
<code>upper</code>	Upper boundary
<code>mu</code>	Mean
<code>sigma</code>	Standard deviation
<code>scaling</code>	Determines the way in which non-default values of <code>mu</code> and <code>sigma</code> are applied. When <code>scaling="points"</code> , the quadrature points are rescaled, otherwise the quadrature weights are adapted.

Value

A list of:

`quad.points` A vector of `n` quadrature points
`quad.weights` A vector of the corresponding quadrature weights

Author(s)

Ivailo Partchev

See Also[read.qu.icl](#), [eap](#), [sca](#)**Examples**

```
quad <- normal.qu(n=20)
```

 npp

Non-parametric characteristic curves

Description

A plotting routine producing non-parametric analogues of the IRF not unlike those in Jim Ramsay's TestGraf program. The curves are produced by a kernel binomial regression of the actual responses to an item on some estimates of the latent variable, by courtesy of package `sm`.

Usage

```
npp(resp, x, items, from = -4, to = 4,
     co = 1, main = "Non-parametric response function",
     add = FALSE, bands = FALSE, label = FALSE)
```

Arguments

<code>resp</code>	A matrix of responses: persons as rows, items as columns, entries are either 0 or 1, no missing data
<code>x</code>	The values of the latent variable ("ability") for the same persons whose responses are given in <code>resp</code> . If not given, function <code>qrs</code> will be plugged in, which is the approach of TestGraf
<code>items</code>	An index to the items (columns of <code>resp</code>) to be shown on the plot. If not given, all items will be plotted.
<code>from</code>	Lower limit for ability on the plot. Default is -4.
<code>to</code>	Upper limit for ability on the plot. Default is 4.
<code>add</code>	When <code>add=T</code> , the curve is added to a plot, otherwise a new plot is started. Default is F.
<code>main</code>	The main title of the plot, given that <code>add=F</code> .
<code>co</code>	The colour of the curves. Default is 1 for black. Use <code>co=NA</code> to plot each curve in a different colour.
<code>bands</code>	When <code>bands=T</code> , confidence bands are added.
<code>label</code>	When <code>label=T</code> , individual curves will be labeled with the item number.

Author(s)

Ivailo Partchev

References

James O. Ramsay (2000). TestGraf: A program for the graphical analysis of multiple choice test and questionnaire data. McGill University, Montreal, Canada

See Also

[qrs](#), [irf](#), [plot.irf](#)

Examples

```
data(Scored)
p.2pl <- est(Scored, model="2PL", engine="ltm")

# plot items 1:5 in different colours, label
npp(Scored, items=1:5, co=NA, label=TRUE)

# For item 7, compare npp with the 2PL parametric IRF
npp(Scored, items=7, bands=TRUE)
plot(irf(ip=p.2pl[7,]), co=3, add=TRUE)
```

plot.iif

A plot method for item information functions

Description

Useful for plotting item information functions. The `x` argument of `iif` should better be left out unless something special is required.

Usage

```
## S3 method for class 'iif':
plot(x, add = FALSE, main = "Item information function",
     co = 1, label = FALSE, ...)
```

Arguments

<code>x</code>	An object produced by function <code>iif</code>
<code>add</code>	When <code>add=T</code> , the IIF is added to a plot, otherwise a new plot is started. Default is F.
<code>main</code>	The main title of the plot, given that <code>add=F</code> .
<code>co</code>	The colour of the IIF curve. Default is 1 for black. Use <code>co=NA</code> to plot each IIF in a different colour.
<code>label</code>	When <code>label=T</code> , individual curves will be labeled with the item number.
<code>...</code>	Any additional plotting parameters

Author(s)

Ivailo Partchev

See Also[iif](#)**Examples**

```
data(Scored)
p.2pl <- est(Scored, model="2PL", engine="ltm")
# plot IIF for all items in red, label with item number
plot(iif(p.2pl), co="red", label=TRUE)
# plot IIF for items 2, 3, and 7 in different colours
plot(iif(p.2pl[c(2,3,7),]), co=NA)
```

`plot.irf`*A plot method for item response functions*

Description

Useful for plotting item response functions. The `x` argument of `irf` should better be left out unless something special is required.

Usage

```
## S3 method for class 'irf':
plot(x, add = FALSE, main = "Item response function",
     co = 1, label = FALSE, ...)
```

Arguments

<code>x</code>	An object produced by function <code>irf</code>
<code>add</code>	When <code>add=T</code> , the IRF is added to a plot, otherwise a new plot is started. Default is F.
<code>main</code>	The main title of the plot, given that <code>add=F</code> .
<code>co</code>	The colour of the IRF curve. Default is 1 for black. Use <code>co=NA</code> to plot each IRF in a different colour.
<code>label</code>	When <code>label=T</code> , individual curves will be labeled with the item number.
<code>...</code>	Any additional plotting parameters

Author(s)

Ivailo Partchev

See Also[irf](#)**Examples**

```
data(Scored)
p.2pl <- est(Scored, model="2PL", engine="ltm")
# plot IRF for all items in red, label with item number
plot(irf(p.2pl), co="red", label=TRUE)
# plot IRF for items 2, 3, and 7 in different colours
plot(irf(p.2pl[c(2,3,7),]), co=NA)
```

plot.tif

*A plot method for test information functions***Description**

Useful for plotting test information functions. The `x` argument of `tif` should better be left out unless something special is required.

Usage

```
## S3 method for class 'tif':
plot(x, add = FALSE, main = "Test information function", co = 1, ...)
```

Arguments

<code>x</code>	An object produced by function <code>tif</code>
<code>add</code>	When <code>add=T</code> , the TIF is added to a plot, otherwise a new plot is started. Default is F.
<code>main</code>	The main title of the plot, given that <code>add=F</code> .
<code>co</code>	The colour of the TIF curve. Default is 1 for black. Use <code>co=NA</code> to plot each TIF in a different colour.
<code>...</code>	Any additional plotting parameters

Author(s)

Ivailo Partchev

See Also[tif](#)**Examples**

```
data(Scored)
p.2pl <- est(Scored, model="2PL", engine="ltm")
plot(tif(p.2pl))
```

`plot.trf`*A plot method for test response functions*

Description

Useful for plotting test response functions. The `x` argument of `trf` should better be left out unless something special is required.

Usage

```
## S3 method for class 'trf':  
plot(x, add = FALSE, main = "Test response function", co = 1, ...)
```

Arguments

<code>x</code>	An object produced by function <code>trf</code>
<code>add</code>	When <code>add=T</code> , the IRF is added to a plot, otherwise a new plot is started. Default is F.
<code>main</code>	The main title of the plot, given that <code>add=F</code> .
<code>co</code>	The colour of the TRF curve. Default is 1 for black. Use <code>co=NA</code> to plot each TRF in a different colour.
<code>...</code>	Any additional plotting parameters

Author(s)

Ivailo Partchev

See Also

[trf](#)

Examples

```
data(Scored)  
p.2pl <- est(Scored, model="2PL", engine="ltm")  
plot(trf(p.2pl))
```

`qrs`*Quantiles of the ranked sum scores*

Description

A rough estimate of the values of the latent variable ("ability"). The sum scores (number of correct responses) are ranked, breaking ties at random. The ranks are divided by the sample size + 1, and the corresponding quantiles of the standard Normal distribution are returned. Used as default in the non-parametric IRF plots produced by [npp](#) in analogy to Jim Ramsay's TestGraf. Another possible use is in [itf](#).

Usage

```
qrs(resp)
```

Arguments

<code>resp</code>	A matrix of responses: persons as rows, items as columns, entries are either 0 or 1, no missing data
-------------------	--

Value

A one-column matrix of values

Author(s)

Ivailo Partchev

See Also

[npp](#), [itf](#)

Examples

```
data(Scored)
sc <- qrs(Scored)
```

read.ip.bilog *Read in parameter estimates*

Description

From BILOG output, read in estimates of item parameters. Invoked automatically when the model is estimated via `irtoys`. If that is not the case, `file` must be a file produced with the `>SAVE PARM file;` command in BILOG.

Usage

```
read.ip.bilog(file)
```

Arguments

`file` File name

Value

A matrix with one row per item, and three columns: [,1] item discrimination a , [,2] item difficulty b , and [,3] asymptote c . For the 1PL and 2PL models, all asymptotes are equal to 0; for the 1PL, the discriminations are all equal but not necessarily equal to 1.

Author(s)

Ivailo Partchev

read.ip.icl *Read in parameter estimates*

Description

From ICL output, read in estimates of item parameters. Invoked automatically when the model is estimated via `irtoys`. If that is not the case, `file` must be a file produced with the `write_item_param file` command in ICL.

Usage

```
read.ip.icl(file)
```

Arguments

`file` File name

Value

A matrix with one row per item, and three columns: [,1] item discrimination a , [,2] item difficulty b , and [,3] asymptote c . For the 1PL and 2PL models, all asymptotes are equal to 0; for the 1PL, the discriminations are all equal but not necessarily equal to 1.

Author(s)

Ivailo Partchev

read.qu.icl	<i>Read in quadrature</i>
-------------	---------------------------

Description

From ICL output, read in estimates of item parameters. `file` must be a file produced with the `write_latent_dist` file command in ICL. Quadrature objects are used when estimating abilities with `eap` and for some of the scaling methods in `sca`.

Usage

```
read.qu.icl(file)
```

Arguments

<code>file</code>	File name
-------------------	-----------

Value

A list of:

`quad.points` A vector of quadrature points

`quad.weights` A vector of the corresponding quadrature weights

Author(s)

Ivailo Partchev

See Also

[normal.qu](#), [eap](#), [sca](#)

read.resp	<i>Read responses from a file</i>
-----------	-----------------------------------

Description

Reads responses to a questionnaire from a text file

Usage

```
read.resp(file, na=".")
```

Arguments

file	File name
na	The symbol used to represent missing data

Details

Included for those of my students who are too faint-hearted to write as `.matrix(read.table(file, head=F))`. Of course, data can be entered into R in many other ways.

The data values in the `file` must be separated with blanks.

Responses are the empirical data used in IRT. Note that `irttoys` deals with models for dichotomous data, and typically expects data consisting of zeroes and ones, without any missing values (non-responses are considered as wrong responses). In fact, there are only two commands in `irttoys` that accept other kinds of data: `sco` and `tgp`.

`read.resp` does accept missing data and values other than 0 and 1. Use `sco` and a key to score multiple choice responses to 0/1. If you have dichotomous data that contains NAs, you can use `sco` without a key to change all NA to 0.

Value

A matrix, typically of zeroes and ones, representing the correct or wrong responses given by persons (rows) to items (columns).

Author(s)

Ivailo Partchev

See Also

[sco](#), [tgp](#),

Examples

```
## Not run:  
r <- read.resp("c:/myfiles/irt.dat")  
## End(Not run)
```

sca

*Linear transformation of the IRT scale***Description**

Linearly transform a set of IRT parameters to bring them to the scale of another set of parameters. Four methods are implemented: Mean/Mean, Mean/Sigma, Lord-Stocking, and Haebara.

Usage

```
sca(old.ip, new.ip, old.items, new.items,
    old.qu = NULL, new.qu = NULL, method = "MS")
```

Arguments

<code>old.ip</code>	A set of parameters that are already on the desired scale
<code>new.ip</code>	A set of parameters that must be placed on the same scale as <code>old.ip</code>
<code>old.items</code>	A vector of indexes pointing to those items in <code>old.ip</code> that are common to both sets of parameters
<code>new.items</code>	The indexes of the same items in <code>new.ip</code>
<code>old.qu</code>	A quadrature object for <code>old.ip</code> , typically produced by the same program that estimated <code>old.ip</code> . Only needed if <code>method="LS"</code> or <code>method="HB"</code>
<code>new.qu</code>	A quadrature object for <code>new.ip</code> , typically produced by the same program that estimated <code>new.ip</code> . Only needed if <code>method="HB"</code>
<code>method</code>	One of "MM" (Mean/Mean), "MS" (Mean/Sigma), "SL" (Stocking-Lord), or "HB" (Haebara). Default is "MS"

Value

A list of:

<code>slope</code>	The slope of the linear transformation
<code>intercept</code>	The intercept of the linear transformation
<code>scaled.ip</code>	The parameters in <code>new.ip</code> transformed to a scale that is compatible with <code>old.ip</code>

Note

Although the Stocking-Lord and the Haebara methods were programmed after Kolen & Brennan (1995), they do not produce the same results as the program ST. Results for Stocking-Lord come closer to ST than those for the Haebara method.

Author(s)

Ivailo Partchev

References

Kolen, M.J. & R.L. Brennan (1995) *Test Equating: Methods and Practices*. Springer.

Examples

```
# a small simulation to demonstrate transformation to a common scale
# fake 50 2PL items
pa <- cbind(runif(50,.8,2), runif(50,-2.4,2.4), rep(0,50))
# simulate responses with two samples of different ability levels
r.1 <- sim(ip=pa[1:30,], x=rnorm(1000,-.5))
r.2 <- sim(ip=pa[21:50,], x=rnorm(1000,.5))
# estimate item parameters
p.1 <- est(r.1, engine="ltm")
p.2 <- est(r.2, engine="ltm")
# plot difficulties to show difference in scale
plot(c(-3,3), c(-3,3), ty="n", xlab="True",ylab="Estimated",
     main="Achieving common scale")
text(pa[1:30,2], p.1[,2], 1:30)
text(pa[21:50,2], p.2[,2], 21:50, co=2)
# scale with the default Mean/Sigma method
pa.sc = sca(old.ip=p.1, new.ip=p.2, old.items=21:30, new.items=1:10)
# add difficulties of scaled items to plot
text(pa[21:50,2], pa.sc$scaled.ip[,2], 21:50, co=3)
```

sco

Score a multiple choice test

Description

Given a key, score a multiple choice test, i.e. recode the original choices to right (1) or wrong (0). Missing responses are treated as wrong.

Usage

```
sco(choices, key, na.false=FALSE)
```

Arguments

choices	The original responses to the items in the test: persons as rows, items as columns. May contain NA.
key	A vector containing the key (correct answers) to the items in choices. If not given, the function will check if all data are either 0, 1, or NA: if yes, NA are recoded as 0, else an error message is returned.
na.false	Recode non-responses to false responses?

Author(s)

Ivailo Partchev

Examples

```
data(Unscored)
res <- sco(Unscored, key=c(2,3,1,1,4,1,2,1,2,3,3,4,3,4,2,2,4,3))
```

Scored	<i>Binary (true/false) responses to a test</i>
--------	--

Description

Real-life data set containing the responses to a test, scored as true or false.

Usage

```
data(Scored)
```

Format

A data set with 472 persons and 18 items.

sim	<i>Simulate response data</i>
-----	-------------------------------

Description

Simulate responses from the 1PL, 2PL, or 3PL model

Usage

```
sim(ip, x = NULL)
```

Arguments

ip	Item parameters: a matrix with one row per item, and three columns: [,1] item discrimination a , [,2] item difficulty b , and [,3] asymptote c .
x	A vector of values of the latent variable ("abilities").

Value

A matrix of responses: persons as rows, items as columns, entries are either 0 or 1, no missing data

Author(s)

Ivailo Partchev

Examples

```
pa <- cbind(runif(20, .8, 2), runif(20, -2.4, 2.4), rep(0, 50))
rs <- sim(ip=pa, x=rnorm(1000))
```

tgp

*Non-parametric option curves***Description**

A plotting function producing non-parametric analogues of the IRF for each option in a multiple choice item not unlike those in Jim Ramsay's TestGraf program.

Usage

```
tgp(choices, key, item,
     main = "Non-parametric response function", co = 1, label = FALSE)
```

Arguments

choices	A matrix of responses to multiple-choice items: persons as rows, items as columns. As a rare exception in <code>irtoys</code> , responses must not be recoded to 0/1, and there may be missing responses.
key	A vector containing the key (correct answers) to the items in <code>choices</code> .
item	A single number pointing to the item (column of <code>choices</code>) to plot.
main	The main title of the plot, given that <code>add=F</code> .
co	The colour of the curves. Default is 1 for black. Use <code>co=NA</code> to plot each curve in a different colour.
label	When <code>label=T</code> , individual curves will be labeled with the item number.

Author(s)

Ivailo Partchev

References

James O. Ramsay (2000). TestGraf: A program for the graphical analysis of multiple choice test and questionnaire data. McGill University, Montreal, Canada

See Also

[qrs](#), [irf](#), [plot.irf](#)

Examples

```
data(Unscored)
key=c(2,3,1,1,4,1,2,1,2,3,3,4,3,4,2,2,4,3)
tgp(choices=Unscored, key=key, item=4, co=NA, label=TRUE)
```

`tif`*Test information function*

Description

Returns the test information function (TIF) of the 3PL (1PL, 2PL) model. The TIF is the sum of the item information functions (IIF) in a test, and indicates the precision of measurement that can be achieved with the test at any value of the latent variable, being inversely related to measurement variance.

A common use of this function would be to obtain a plot of the TIF.

Usage

```
tif(ip, x = NULL)
```

Arguments

<code>ip</code>	Item parameters: a matrix with one row per item, and three columns: [,1] item discrimination a , [,2] item difficulty b , and [,3] asymptote c .
<code>x</code>	The values of the latent variable (θ in the equation above), at which the TIF will be evaluated. If not given, 99 values spaced evenly between -4 and +4 will be used, handy for plotting.

Value

A list of:

<code>x</code>	A copy of the argument <code>x</code>
<code>f</code>	A vector containing the TIF values

Author(s)

Ivailo Partchev

See Also

[plot.tif](#), [iif](#)

Examples

```
data(Scored)
p.2pl <- est(Scored, model="2PL", engine="ltm")
plot(trf(p.2pl))
```

trf *Test response function*

Description

Returns the test response function (TRF) of the 3PL (1PL, 2PL) model. The TRF is the sum of the item response functions (IRF) in a test, and represents the expected score as a function of the latent variable θ .

A common use of this function would be to obtain a plot of the TRF.

Usage

```
trf(ip, x = NULL)
```

Arguments

ip	Item parameters: a matrix with one row per item, and three columns: [,1] item discrimination a , [,2] item difficulty b , and [,3] asymptote c .
x	The values of the latent variable (θ in the equation above), at which the IRF will be evaluated. If not given, 99 values spaced evenly between -4 and +4 will be used, handy for plotting.

Value

A list of:

x	A copy of the argument x
f	A vector containing the TRF values

Author(s)

Ivailo Partchev

See Also

[plot.trf](#), [irf](#)

Examples

```
data(Scored)
p.2pl <- est(Scored, model="2PL", engine="ltm")
plot(trf(p.2pl))
```

tsc	<i>True scores with standard errors</i>
-----	---

Description

Computes the IRT true scores (test response function at the estimated ability) and an estimate of their standard error via the delte theorem, treating item parameters as known).

Usage

```
tsc(ip, theta)
```

Arguments

ip	Item parameters: a matrix with one row per item, and three columns: [,1] item discrimination a , [,2] item difficulty b , and [,3] asymptote c .
theta	An object containing ability estimates, as output by function <code>mlebme</code> or <code>eap</code>

Value

A matrix with the true scores in column 1, and their standard errors of measurement (SEM) in column 2

Author(s)

Ivailo Partchev

See Also

[mlebme](#), [eap](#), [trf](#)

Examples

```
data(Scored)
p.2pl <- est(Scored, model="2PL", engine="ltm")
th <- mlebme(resp=Scored, ip=p.2pl)
tsc(p.2pl, th)
```

Unscored

Original, unscored multiple-choice responses to a test

Description

Real-life data set containing the responses to a test, before they have been recoded as true or false. Can be used with only two functions in the package: `sco` and `npp`. All other functions expect binary data, which can be produced with `sco`.

Usage

```
data(Unscored)
```

Format

A data set with 472 persons and 18 items. Each item has 4 possible answers, of which only one is true. There are many NA, which can be treated as wrong responses.

Index

- *Topic **IO**
 - read.resp, 24
- *Topic **datasets**
 - Scored, 27
 - Unscored, 32
- *Topic **data**
 - read.ip.bilog, 22
- *Topic **list**
 - l2m, 14
- *Topic **models**
 - api, 3
 - dpv, 4
 - eap, 5
 - est, 6
 - grp, 8
 - iif, 9
 - internal, 10
 - irf, 10
 - irtoys-package, 2
 - itf, 11
 - mlebme, 14
 - normal.qu, 15
 - npp, 16
 - plot.iif, 17
 - plot.irf, 18
 - plot.tif, 19
 - plot.trf, 20
 - qrs, 21
 - read.ip.icl, 22
 - read.qu.icl, 23
 - sca, 25
 - sco, 26
 - sim, 27
 - tgp, 28
 - tif, 29
 - trf, 30
 - tsc, 31
- adv.scale(*internal*), 10
- api, 3
- ddf(*internal*), 10
- dpv, 4
- dpv.one(*internal*), 10
- eap, 4, 5, 13, 15, 16, 23, 31
- eap.one(*internal*), 10
- est, 6
- est.blm(*internal*), 10
- est.icl(*internal*), 10
- est.ltm(*internal*), 10
- grp, 8, 13
- hb(*internal*), 10
- iif, 9, 18, 29
- internal, 10
- irf, 10, 10, 17, 19, 28, 30
- irtoys(*irtoys-package*), 2
- irtoys-package, 2
- itf, 9, 11, 21
- l2m, 14
- like(*internal*), 10
- llf(*internal*), 10
- mle.one(*internal*), 10
- mlebme, 4, 5, 14, 31
- normal.qu, 5, 15, 23
- npp, 16, 21
- plot.iif, 10, 17
- plot.irf, 11, 17, 18, 28
- plot.tif, 19, 29
- plot.trf, 20, 30
- qrs, 13, 17, 21, 28
- read.ip.bilog, 22
- read.ip.icl, 22

read.qu.icl, 5, 16, 23
read.resp, 24

sca, 16, 23, 25
sco, 24, 26
Scored, 27
sim, 27
simple.scale(*internal*), 10
sl(*internal*), 10

tgp, 24, 28
tif, 19, 29
trf, 20, 30, 31
tsc, 31

Unscored, 32