

Package ‘ltm’

October 12, 2009

Title Latent Trait Models under IRT

Version 0.9-2

Date 2009-10-12

Author Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

Maintainer Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

Description Analysis of multivariate dichotomous and polytomous data using latent trait models under the Item Response Theory approach. It includes the Rasch, the Two-Parameter Logistic, the Birnbaum’s Three-Parameter, the Graded Response, and the Generalized Partial Credit Models.

Depends R (>= 2.5.0), MASS, msm, mvtnorm, polycor

LazyLoad yes

LazyData yes

License GPL (>= 2)

URL <http://wiki.r-project.org/rwiki/doku.php?id=packages:cran:ltm>

Repository CRAN

Date/Publication 2009-10-12 10:23:12

R topics documented:

ltm-package	2
Abortion	3
anova	4
biserial.cor	7
coef	8
cronbach.alpha	10
descript	11
Environment	13
factor.scores	14

fitted	17
GoF	19
gpcm	20
grm	23
information	26
item.fit	27
LSAT	30
ltm	30
margins	34
Mobility	36
mult.choice	37
person.fit	38
plot descript	41
plot fscores	42
plot IRT	43
rasch	48
rcor.test	51
residuals	52
rmvlogis	53
Science	55
summary	56
testEquatingData	58
tpm	59
unidimTest	62
vcov	64
WIRS	66
Index	67

lrm-package

Latent Trait Models for Item Response Theory Analyses

Description

This package provides a flexible framework for Item Response Theory analyses for dichotomous and polytomous data under a Marginal Maximum Likelihood approach. The fitting algorithms provide valid inferences under Missing At Random missing data mechanisms.

Details

```

Package: ltm
Type: Package
Version: 0.9-2
Date: 2009-10-05
License: GPL

```

The following options are available:

Descriptives: samples proportions, missing values information, biserial correlation of items with total score, pairwise associations between items, Cronbach's α , unidimensionality check using modified parallel analysis, nonparametric correlation coefficient, plotting of sample proportions versus total score.

Dichotomous data: Rasch Model, Two Parameter Logistic Model, Birnbaum's Three Parameter Model, and Latent Trait Model up to two latent variables (allowing also for nonlinear terms between the latent traits).

Polytomous data: Samejima's Graded Response Model and the Generalized Partial Credit Model.

Goodness-of-Fit: Bootstrapped Pearson χ^2 for Rasch and Generalized Partial Credit models, fit on the two- and three-way margins for all models, likelihood ratio tests between nested models (including AIC and BIC criteria values), and item- and person-fit statistics.

Factor Scoring - Ability Estimates: Empirical Bayes (i.e., posterior modes), Expected a posteriori (i.e., posterior means), Multiple Imputed Empirical Bayes, and Component Scores for dichotomous data.

Test Equating: Alternate Form Equating (where common and unique items are analyzed simultaneously) and Across Sample Equating (where different sets of unique items are analyzed separately based on previously calibrated anchor items).

Plotting: Item Characteristic Curves, Item Information Curves, Test Information Functions, Standard Error of Measurement, Standardized Loadings Scatterplot (for the two-factor latent trait model), Item Operation Characteristic Curves (for ordinal polytomous data), Item Person Maps.

More information as well as .R files containing sample analyses can be found in the Rwiki page of package **ltm** available at:

<http://wiki.r-project.org/rwiki/doku.php?id=packages:cran:ltm>.

Author(s)

Dimitris Rizopoulos

Maintainer: Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

References

Baker, F. and Kim, S-H. (2004) *Item Response Theory*, 2nd ed. New York: Marcel Dekker.

Rizopoulos, D. (2006) **ltm**: An R package for latent variable modelling and item response theory analyses. *Journal of Statistical Software*, **17(5)**, 1–25. URL <http://www.jstatsoft.org/v17/i05/>

Description

The data contain responses given by 410 individuals to four out of seven items concerning attitude to abortion. A small number of individual did not answer to some of the questions and this data set contains only the complete cases.

Format

379 individuals answered to the following questions after being asked if the law should allow abortion under the circumstances presented under each item,

Item 1 The woman decides on her own that she does not.

Item 2 The couple agree that they do not wish to have a child.

Item 3 The woman is not married and does not wish to marry the man.

Item 4 The couple cannot afford any more children.

Source

1986 British Social Attitudes Survey (McGrath and Waterton, 1986).

References

Bartholomew, D., Steel, F., Moustaki, I. and Galbraith, J. (2002) *The Analysis and Interpretation of Multivariate Data for Social Scientists*. London: Chapman and Hall.

Knott, M., Albanese, M. and Galbraith, J. (1990) Scoring attitudes to abortion. *The Statistician*, **40**, 217–223.

McGrath, K. and Waterton, J. (1986) *British social attitudes, 1983-86 panel survey*. London: SCPR.

Examples

```
## Descriptive statistics for Abortion data
dsc <- descript(Abortion)
dsc
plot(dsc)
```

anova

Anova method for fitted IRT models

Description

Performs a Likelihood Ratio Test between two nested IRT models.

Usage

```
## S3 method for class 'gpcm':
anova(object, object2, simulate.p.value = FALSE,
       B = 200, verbose = getOption("verbose"), seed = NULL, ...)

## S3 method for class 'grm':
anova(object, object2, ...)
```

```
## S3 method for class 'ltm':
anova(object, object2, ...)

## S3 method for class 'rasch':
anova(object, object2, ...)

## S3 method for class 'tpm':
anova(object, object2, ...)
```

Arguments

<code>object</code>	an object inheriting from either class <code>gpcm</code> , class <code>grm</code> , class <code>ltm</code> , class <code>rasch</code> or class <code>tpm</code> , representing the model under the null hypothesis.
<code>object2</code>	an object inheriting from either class <code>gpcm</code> , class <code>grm</code> , class <code>ltm</code> , class <code>rasch</code> , or class <code>tpm</code> , representing the model under the alternative hypothesis.
<code>simulate.p.value</code>	logical; if TRUE, the reported p -value is based on a parametric Bootstrap approach.
<code>B</code>	the number of Bootstrap samples.
<code>verbose</code>	logical; if TRUE, information is printed in the console during the parametric Bootstrap.
<code>seed</code>	the seed to be used during the parametric Bootstrap; if NULL, a random seed is used.
<code>...</code>	additional arguments; currently none is used.

Details

`anova.gpcm()` also includes the option to estimate the p -value of the LRT using a parametric Bootstrap approach. In particular, `B` data sets are simulated under the null hypothesis (i.e., under the generalized partial credit model `object`), and both the null and alternative models are fitted and the value of LRT is computed. Then the p -value is approximate using $[1 + \sum_{i=1}^B I(T_i > T_{obs})]/(B + 1)$, where T_{obs} is the value of the likelihood ratio statistic in the original data set, and T_i the value of the statistic in the i th Bootstrap sample.

In addition, when `simulate.p.value = TRUE` objects of class `aov.gpcm` have a method for the `plot()` generic function that produces a QQ plot comparing the Bootstrap sample of likelihood ration statistic with the asymptotic chi-squared distribution. For instance, you can use something like the following: `lrt <- anova(obj1, obj2, simulate.p.value = TRUE); plot(lrt)`.

Value

An object of either class `aov.gpcm`, `aov.grm`, class `aov.ltm` or class `aov.rasch` with components,

<code>nam0</code>	the name of <code>object</code> .
<code>L0</code>	the log-likelihood under the null hypothesis (<code>object</code>).
<code>nb0</code>	the number of parameter in <code>object</code> ; returned only in <code>aov.gpcm</code> .

aic0	the AIC value for the model given by <code>object</code> .
bic0	the BIC value for the model given by <code>object</code> .
nam1	the name of <code>object2</code> .
L1	the log-likelihood under the alternative hypothesis (<code>object2</code>).
nb1	the number of parameter in <code>object</code> ; returned only in <code>aov.gpcm</code> .
aic1	the AIC value for the model given by <code>object2</code> .
bic1	the BIC value for the model given by <code>object2</code> .
LRT	the value of the Likelihood Ratio Test statistic.
df	the degrees of freedom for the test (i.e., the difference in the number of parameters).
p.value	the p -value of the test.

Warning

The code does not check if the models are nested! The user is responsible to supply nested models in order the LRT to be valid.

When `object2` represents a three parameter model, note that the null hypothesis is on the boundary of the parameter space for the guessing parameters. Thus, the Chi-squared reference distribution used by these function might not be totally appropriate.

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

See Also

[GoF.gpcm](#), [GoF.rasch](#), [gpcm](#), [grm](#), [ltm](#), [rasch](#), [tpm](#)

Examples

```
## LRT between the constrained and unconstrained GRMs
## for the Science data:
fit0 <- grm(Science[c(1,3,4,7)], constrained = TRUE)
fit1 <- grm(Science[c(1,3,4,7)])
anova(fit0, fit1)

## LRT between the one- and two-factor models
## for the WIRS data:
anova(ltm(WIRS ~ z1), ltm(WIRS ~ z1 + z2))

## An LRT between the Rasch and a constrained
## two-parameter logistic model for the WIRS data:
fit0 <- rasch(WIRS)
fit1 <- ltm(WIRS ~ z1, constraint = cbind(c(1, 3, 5), 2, 1))
anova(fit0, fit1)

## An LRT between the constrained (discrimination
```

```
## parameter equals 1) and the unconstrained Rasch
## model for the LSAT data:
fit0 <- rasch(LSAT, constraint = rbind(c(6, 1)))
fit1 <- rasch(LSAT)
anova(fit0, fit1)

## An LRT between the Rasch and the two-parameter
## logistic model for the LSAT data:
anova(rasch(LSAT), ltm(LSAT ~ z1))
```

biserial.cor *Point-Biserial Correlation*

Description

Computes the point-biserial correlation between a dichotomous and a continuous variable.

Usage

```
biserial.cor(x, y, use = c("all.obs", "complete.obs"), level = 1)
```

Arguments

<code>x</code>	a numeric vector representing the continuous variable.
<code>y</code>	a factor or a numeric vector (that will be converted to a factor) representing the dichotomous variable.
<code>use</code>	If <code>use</code> is "all.obs", then the presence of missing observations will produce an error. If <code>use</code> is "complete.obs" then missing values are handled by casewise deletion.
<code>level</code>	which level of <code>y</code> to use.

Details

The point biserial correlation computed by `biserial.cor()` is defined as follows

$$r = \frac{(\bar{X}_1 - \bar{X}_0)\sqrt{\pi(1-\pi)}}{S_x},$$

where \bar{X}_1 and \bar{X}_0 denote the sample means of the X -values corresponding to the first and second level of Y , respectively, S_x is the sample standard deviation of X , and π is the sample proportion for $Y = 1$. The first level of Y is defined by the `level` argument; see **Examples**.

Value

the (numeric) value of the point-biserial correlation.

Note

Changing the order of the levels for y will produce a different result. By default, the first level is used as a reference level

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

Examples

```
# the point-biserial correlation between
# the total score and the first item, using
# '0' as the reference level
biserial.cor(rowSums(LSAT), LSAT[[1]])

# and using '1' as the reference level
biserial.cor(rowSums(LSAT), LSAT[[1]], level = 2)
```

 coef

Extract Estimated Loadings

Description

Extracts the estimated parameters from either `grm`, `ltm`, `rasch` or `tpm` objects.

Usage

```
## S3 method for class 'gpcm':
coef(object, ...)

## S3 method for class 'grm':
coef(object, ...)

## S3 method for class 'ltm':
coef(object, standardized = FALSE, prob = FALSE, order = FALSE, ...)

## S3 method for class 'rasch':
coef(object, prob = FALSE, order = FALSE, ...)

## S3 method for class 'tpm':
coef(object, prob = FALSE, order = FALSE, ...)
```

Arguments

object	an object inheriting from either class <code>gpcm</code> , class <code>grm</code> , class <code>ltm</code> , class <code>rasch</code> or class <code>tpm</code> .
standardized	logical; if TRUE the standardized loadings are also returned. See Details for more info.
prob	logical; if TRUE the probability of a positive response for the median individual (i.e., $Pr(x_i = 1 z = 0)$, with $i = 1, \dots, p$ denoting the items) is also returned.
order	logical; if TRUE the items are sorted according to the difficulty estimates.
...	additional arguments; currently none is used.

Details

The standardization of the factor loadings is useful in order to form a link to the Underlying Variable approach. In particular, the standardized form of the factor loadings represents the correlation coefficient between the latent variables and the underlying continuous variables based on which the dichotomous outcomes arise (see Bartholomew and Knott, 1999, p.87-88 or Bartholomew *et al.*, 2002, p.191).

The standardized factor loadings are computed only for the linear one- and two-factor models, fitted by `ltm()`.

Value

A list or a matrix of the estimated parameters for the fitted model.

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

References

- Bartholomew, D. and Knott, M. (1999) *Latent Variable Models and Factor Analysis*, 2nd ed. London: Arnold.
- Bartholomew, D., Steel, F., Moustaki, I. and Galbraith, J. (2002) *The Analysis and Interpretation of Multivariate Data for Social Scientists*. London: Chapman and Hall.

See Also

[gpcm](#), [grm](#), [ltm](#), [rasch](#), [tpm](#)

Examples

```
fit <- grm(Science[c(1,3,4,7)])
coef(fit)

fit <- ltm(LSAT ~ z1)
coef(fit, TRUE, TRUE)
```

```
m <- rasch(LSAT)
coef(fit, TRUE, TRUE)
```

cronbach.alpha *Cronbach's alpha*

Description

Computes Cronbach's alpha for a given data-set.

Usage

```
cronbach.alpha(data, standardized = FALSE, CI = FALSE,
  probs = c(0.025, 0.975), B = 1000, na.rm = FALSE)
```

Arguments

`data` a matrix or a data.frame containing the items as columns.

`standardized` logical; if TRUE the standardized Cronbach's alpha is computed.

`CI` logical; if TRUE a Bootstrap confidence interval for Cronbach's alpha is computed.

`probs` a numeric vector of length two indicating which quantiles to use for the Bootstrap CI.

`B` the number of Bootstrap samples to use.

`na.rm` logical; what to do with NA's.

Details

The Cronbach's alpha computed by `cronbach.alpha()` is defined as follows

$$\alpha = \frac{p}{p-1} \left(1 - \frac{\sum_{i=1}^p \sigma_{y_i}^2}{\sigma_x^2} \right),$$

where p is the number of items σ_x^2 is the variance of the observed total test scores, and $\sigma_{y_i}^2$ is the variance of the i th item.

The standardized Cronbach's alpha computed by `cronbach.alpha()` is defined as follows

$$\alpha_s = \frac{p \cdot \bar{r}}{1 + (p-1) \cdot \bar{r}},$$

where p is the number of items, and \bar{r} is the average of all (Pearson) correlation coefficients between the items. In this case if `na.rm = TRUE`, then the complete observations (i.e., rows) are used.

The Bootstrap confidence interval is calculated by simply taking B samples with replacement from data, calculating for each α or α_s , and computing the quantiles according to `probs`.

Value

`cronbach.alpha()` returns an object of class `cronbachAlpha` with components

<code>alpha</code>	the value of Cronbach's alpha.
<code>n</code>	the number of sample units.
<code>p</code>	the number of items.
<code>standardized</code>	a copy of the <code>standardized</code> argument.
<code>name</code>	the name of argument <code>data</code> .
<code>ci</code>	the confidence interval for alpha; returned if <code>CI = TRUE</code> .
<code>probs</code>	a copy of the <code>probs</code> argument; returned if <code>CI = TRUE</code> .
<code>B</code>	a copy of the <code>B</code> argument; returned if <code>CI = TRUE</code> .

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

References

Cronbach, L. J. (1951) Coefficient alpha and the internal structure of tests. *Psychometrika*, **16**, 297–334.

Examples

```
# Cronbach's alpha for the LSAT data-set
# with a Bootstrap 95% CI
cronbach.alpha(LSAT, CI = TRUE, B = 500)
```

descript

Descriptive Statistics

Description

Computes descriptive statistics for dichotomous and polytomous response matrices.

Usage

```
descript(data, n.print = 10, chi.squared = TRUE, B = 1000)
```

Arguments

<code>data</code>	a matrix or a <code>data.frame</code> containing the manifest variables as columns.
<code>n.print</code>	numeric indicating the number of pairwise associations with the highest p -values to be printed.
<code>chi.squared</code>	logical; if TRUE the chi-squared test for the pairwise associations between items is performed. See Details for more info.
<code>B</code>	an integer specifying the number of replicates used in the Monte Carlo test (i.e., this is the <code>B</code> argument of <code>chisq.test()</code>).

Details

The following descriptive statistics are returned by `descript()`:

- (i) the proportions for all the possible response categories for each item. In case all items are dichotomous, the logit of the proportion for the positive responses is also included.
- (ii) the frequencies of all possible total scores. The total score of a response pattern is simply its sum. For dichotomous items this is the number of positive responses, whereas for polytomous items this is the sum of the levels represented as numeric values (e.g., the response categories "very concerned", "slightly concerned", and "not very concerned" in `Environment` are represented as 1, 2, and 3).
- (iii) Cronbach's alpha, for all items and excluding each time one of the items.
- (iv) for dichotomous response matrices two versions of the biserial correlation of each item with the total score are returned. In the first one the item is included in the computation of the total score, and in the second one is excluded.
- (v) pairwise associations between items. Before an analysis with latent variable models, it is useful to inspect the data for evidence of positive correlations. In the case of binary or polytomous data, this ad hoc check is performed by constructing the 2×2 contingency tables for all possible pairs of items and examine the Chi-squared p -values. In case any expected frequencies are smaller than 5, `simulate.p.value` is turned to TRUE in `chisq.test()`, using `B` resamples.

Value

`descript()` returns an object of class `descript` with components,

<code>sample</code>	a numeric vector of length 2, with elements the number of items and the number of sample units.
<code>perc</code>	a numeric matrix containing the percentages of negative and positive responses for each item. If <code>data</code> contains only dichotomous manifest variables the logit of the positive responses (i.e., second row) is also included.
<code>items</code>	a numeric matrix containing the frequencies for the total scores.
<code>pw.ass</code>	a matrix containing the p -values for the pairwise association between the items.
<code>n.print</code>	the value of the <code>n.print</code> argument.
<code>name</code>	the name of argument <code>data</code> .

<code>missin</code>	a numeric matrix containing the frequency and percentages of missing values for each item; returned only if any NA's exist in <code>data</code> .
<code>bisCorr</code>	a numeric vector containing sample estimates of the biserial correlation of dichotomous manifest variables with the total score.
<code>ExBisCorr</code>	a numeric vector containing sample estimates of the biserial correlation of dichotomous manifest variables with the total score, where the latter is computed by excluding the specific item.
<code>data</code>	a copy of the <code>data</code> .
<code>alpha</code>	a numeric matrix with one column containing the sample estimates of Cronbach's alpha, for all items and excluding each time one item.

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

See Also

[plot.descript](#), [unidimTest](#)

Examples

```
## Descriptives for LSAT data:
dsc <- descript(LSAT, 3)
dsc
plot(dsc, type = "b", lty = 1, pch = 1:5)
legend("topleft", names(LSAT), pch = 1:5, col = 1:5, lty = 1, bty = "n")
```

Environment

Attitude to the Environment

Description

This data set comes from the Environment section of the 1990 British Social Attitudes Survey (Brook et al., 1991). A sample of 291 responded to the questions below:

Format

All of the below items were measured on a three-group scale with response categories "very concerned", "slightly concerned" and "not very concerned":

LeadPetrol Lead from petrol.

RiverSea River and sea pollution.

RadioWaste Transport and storage of radioactive waste.

AirPollution Air pollution.

Chemicals Transport and disposal of poisonous chemicals.

Nuclear Risks from nuclear power station.

References

- Bartholomew, D., Steel, F., Moustaki, I. and Galbraith, J. (2002) *The Analysis and Interpretation of Multivariate Data for Social Scientists*. London: Chapman and Hall.
- Brook, L., Taylor, B. and Prior, G. (1991) *British Social Attitudes, 1990, Survey*. London: SCPR.

Examples

```
## Descriptive statistics for Environment data
descript(Environment)
```

factor.scores	<i>Factor Scores - Ability Estimates</i>
---------------	--

Description

Computation of factor scores for grm, ltm, rasch and tpm models.

Usage

```
factor.scores(object, ...)

## S3 method for class 'gpcm':
factor.scores(object, resp.patterns = NULL,
              method = c("EB", "EAP", "MI"), B = 5, robust.se = FALSE,
              prior = TRUE, return.MIvalues = FALSE, ...)

## S3 method for class 'grm':
factor.scores(object, resp.patterns = NULL,
              method = c("EB", "EAP", "MI"), B = 5, prior = TRUE,
              return.MIvalues = FALSE, ...)

## S3 method for class 'ltm':
factor.scores(object, resp.patterns = NULL,
              method = c("EB", "EAP", "MI", "Component"), B = 5,
              robust.se = FALSE, prior = TRUE, return.MIvalues = FALSE,
              ...)

## S3 method for class 'rasch':
factor.scores(object, resp.patterns = NULL,
              method = c("EB", "EAP", "MI"), B = 5, robust.se = FALSE,
              prior = TRUE, return.MIvalues = FALSE, ...)

## S3 method for class 'tpm':
factor.scores(object, resp.patterns = NULL,
              method = c("EB", "EAP", "MI"), B = 5, prior = TRUE,
              return.MIvalues = FALSE, ...)
```

Arguments

<code>object</code>	an object inheriting from either class <code>gpcm</code> , class <code>grm</code> , class <code>ltm</code> , class <code>rasch</code> or class <code>tpm</code> .
<code>resp.patterns</code>	a matrix or a data.frame of response patterns with columns denoting the items; if <code>NULL</code> the factor scores are computed for the observed response patterns.
<code>method</code>	a character supplying the scoring method; available methods are: Empirical Bayes, Expected a Posteriori, Multiple Imputation, and Component. See Details section for more info.
<code>B</code>	the number of multiple imputations to be used if <code>method = "MI"</code> .
<code>robust.se</code>	logical; if <code>TRUE</code> the sandwich estimator is used for the estimation of the covariance matrix of the MLEs. See Details section for more info.
<code>prior</code>	logical. If <code>TRUE</code> , then the prior normal distribution for the latent abilities is taken into account in the calculation of the posterior modes, when <code>method = "EB"</code> .
<code>return.MIvalues</code>	logical. If <code>TRUE</code> , then the estimated z-values and their covariance matrix are contained as extra attributes <code>"zvalues.MI"</code> and <code>"var.zvalues.MI"</code> , respectively, in the returned <code>score.dat</code> data frame.
<code>...</code>	additional argument; currently none is used.

Details

Factor scores or ability estimates are summary measures of the posterior distribution $p(z|x)$, where z denotes the vector of latent variables and x the vector of manifest variables.

Usually as factor scores we assign the modes of the above posterior distribution evaluated at the MLEs. These Empirical Bayes estimates (use `method = "EB"`) and their associated variance are good measures of the posterior distribution while $p \rightarrow \infty$, where p is the number of items. This is based on the result

$$p(z|x) = p(z|x; \hat{\theta})(1 + O(1/p)),$$

where $\hat{\theta}$ are the MLEs. However, in cases where p and/or n (the sample size) is small we ignore the variability of plugging-in estimates but not the *true* parameter values. A solution to this problem can be given using Multiple Imputation (MI; use `method = "MI"`). In particular, MI is used the other way around, i.e.,

Step 1: Simulate new parameter values, say θ^* , from $N(\hat{\theta}, C(\hat{\theta}))$, where $C(\hat{\theta})$ is the large sample covariance matrix of $\hat{\theta}$ (if `robust.se = TRUE`, $C(\hat{\theta})$ is based on the sandwich estimator).

Step 2: Maximize $p(z|x; \theta^*)$ wrt z and also compute the associated variance to this mode.

Step 3: Repeat steps 1-2 B times and combine the estimates using the known formulas of MI.

This scheme explicitly acknowledges the ignorance of the true parameter values by drawing from their large sample posterior distribution while taking into account the sampling error. The modes of the posterior distribution $p(z|x; \theta)$ are numerically approximated using the BFGS algorithm in `optim()`.

The Expected a posteriori scores (use `method = "EAP"`) computed by `factor.scores()` are defined as follows:

$$\int zp(z|x;\hat{\theta})dz.$$

The Component scores (use `method = "Component"`) proposed by Bartholomew (1984) is an alternative method to scale the sample units in the latent dimensions identified by the model that avoids the calculation of the posterior mode. However, this method is not valid in the general case where nonlinear latent terms are assumed.

Value

An object of class `fscores` is a list with components,

<code>score.dat</code>	the <code>data.frame</code> of observed response patterns including, observed and expected frequencies (only if the observed data response matrix contains no missing vales), the factor scores and their standard errors.
<code>method</code>	a character giving the scoring method used.
<code>B</code>	the number of multiple imputations used; relevant only if <code>method = "MI"</code> .
<code>call</code>	a copy of the matched call of <code>object</code> .
<code>resp.pats</code>	logical; is <code>TRUE</code> if <code>resp.patterns</code> argument has been specified.
<code>coef</code>	the parameter estimates returned by <code>coef(object)</code> ; this is <code>NULL</code> when <code>object</code> inherits from class <code>grm</code> .

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

References

- Bartholomew, D. (1984) Scaling binary data using a factor model. *Journal of the Royal Statistical Society, Series B*, **46**, 120–123.
- Bartholomew, D. and Knott, M. (1999) *Latent Variable Models and Factor Analysis*, 2nd ed. London: Arnold.
- Bartholomew, D., Steel, F., Moustaki, I. and Galbraith, J. (2002) *The Analysis and Interpretation of Multivariate Data for Social Scientists*. London: Chapman and Hall.
- Rizopoulos, D. (2006) **Itm**: An R package for latent variable modelling and item response theory analyses. *Journal of Statistical Software*, **17(5)**, 1–25. URL <http://www.jstatsoft.org/v17/i05/>
- Rizopoulos, D. and Moustaki, I. (2008) Generalized latent variable models with nonlinear effects. *British Journal of Mathematical and Statistical Psychology*, **61**, 415–438.

See Also

[plot.fscores](#), [gpcm](#), [grm](#), [ltm](#), [rasch](#), [tpm](#)

Examples

```
## Factor Scores for the Rasch model
fit <- rasch(LSAT)
factor.scores(fit) # Empirical Bayes

## Factor scores for specific patterns,
## including NA's, can be obtained by
factor.scores(fit, resp.patterns = rbind(c(1,0,1,0,1), c(NA,1,0,NA,1)))

## Factor Scores for the two-parameter logistic model
fit <- ltm(Abortion ~ z1)
factor.scores(fit, method = "MI", B = 20) # Multiple Imputation

## Factor Scores for the graded response model
fit <- grm(Science[c(1,3,4,7)])
factor.scores(fit, resp.patterns = rbind(1:4, c(NA,1,2,3)))
```

fitted

Fitted Values for IRT model

Description

Computes the expected frequencies for vectors of response patterns.

Usage

```
## S3 method for class 'gpcm':
fitted(object, resp.patterns = NULL,
       type = c("expected", "marginal-probabilities",
               "conditional-probabilities"), ...)

## S3 method for class 'grm':
fitted(object, resp.patterns = NULL,
       type = c("expected", "marginal-probabilities",
               "conditional-probabilities"), ...)

## S3 method for class 'ltm':
fitted(object, resp.patterns = NULL,
       type = c("expected", "marginal-probabilities",
               "conditional-probabilities"), ...)

## S3 method for class 'rasch':
fitted(object, resp.patterns = NULL,
       type = c("expected", "marginal-probabilities",
               "conditional-probabilities"), ...)
```

```
## S3 method for class 'tpm':
fitted(object, resp.patterns = NULL,
       type = c("expected", "marginal-probabilities",
               "conditional-probabilities"), ...)
```

Arguments

object an object inheriting either from class `gpcm`, class `grm`, class `ltm`, class `rasch`, or class `tpm`.

resp.patterns a matrix or a `data.frame` of response patterns with columns denoting the items; if `NULL` the expected frequencies are computed for the observed response patterns.

type if `type == "marginal-probabilities"` the marginal probabilities for each response are computed; these are given by $\int \{\prod_{i=1}^p Pr(x_i = 1|z)^{x_i} \times (1 - Pr(x_i = 1|z))^{1-x_i}\} p(z) dz$, where x_i denotes the i th item and z the latent variable. If `type == "expected"` the expected frequencies for each response are computed, which are the marginal probabilities times the number of sample units. If `type == "conditional-probabilities"` the conditional probabilities for each response and item are computed; these are $Pr(x_i = 1|\hat{z})$, where \hat{z} is the ability estimate .

... additional arguments; currently none is used.

Value

a numeric matrix or a list containing either the response patterns of interest with their expected frequencies or marginal probabilities, if `type == "expected" || "marginal-probabilities"` or the conditional probabilities for each response pattern and item, if `type == "conditional-probabilities"`.

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

See Also

[residuals.gpcm](#), [residuals.grm](#), [residuals.ltm](#), [residuals.rasch](#), [residuals.tpm](#)

Examples

```
fit <- grm(Science[c(1, 3, 4, 7)])
fitted(fit, resp.patterns = matrix(1:4, nr = 4, nc = 4))

fit <- rasch(LSAT)
fitted(fit, type = "conditional-probabilities")
```

Description

Performs a parametric Bootstrap test for Rasch and Generalized Partial Credit models.

Usage

```
GoF.gpcm(object, simulate.p.value = TRUE, B = 99, seed = NULL, ...)
```

```
GoF.rasch(object, B = 49, ...)
```

Arguments

<code>object</code>	an object inheriting from either class <code>gpcm</code> or class <code>rasch</code> .
<code>simulate.p.value</code>	logical; if <code>TRUE</code> , the reported p -value is based on a parametric Bootstrap approach. Otherwise the p -value is based on the asymptotic chi-squared distribution.
<code>B</code>	the number of Bootstrap samples. See Details section for more info.
<code>seed</code>	the seed to be used during the parametric Bootstrap; if <code>NULL</code> , a random seed is used.
<code>...</code>	additional arguments; currently none is used.

Details

`GoF.gpcm` and `GoF.rasch` perform a parametric Bootstrap test based on Pearson's chi-squared statistic defined as

$$\sum_{r=1}^{2^p} \frac{\{O(r) - E(r)\}^2}{E(r)},$$

where r represents a response pattern, $O(r)$ and $E(r)$ represent the observed and expected frequencies, respectively and p denotes the number of items. The Bootstrap approximation to the reference distribution is preferable compared with the ordinary Chi-squared approximation since the latter is not valid especially for large number of items (\Rightarrow many response patterns with expected frequencies smaller than 1).

In particular, the Bootstrap test is implemented as follows:

Step 0: Based on `object` compute the observed value of the statistic T_{obs} .

Step 1: Simulate new parameter values, say θ^* , from $N(\hat{\theta}, C(\hat{\theta}))$, where $\hat{\theta}$ are the MLEs and $C(\hat{\theta})$ their large sample covariance matrix.

Step 2: Using θ^* simulate new data (with the same dimensions as the observed ones), fit the generalized partial credit or the Rasch model and based on this fit calculate the value of the statistic T_i .

Step 3: Repeat steps 1-2 B times and estimate the p -value using $[1 + \sum_{i=1}^B I(T_i > T_{obs})]/(B + 1)$.

Furthermore, in `GoF.gpcm` when `simulate.p.value = FALSE`, then the p -value is based on the asymptotic chi-squared distribution.

Value

An object of class `GoF.gpcm` or `GoF.rasch` with components,

<code>Tobs</code>	the value of the Pearson's chi-squared statistic for the observed data.
<code>B</code>	the B argument specifying the number of Bootstrap samples used.
<code>call</code>	the matched call of object.
<code>p.value</code>	the p -value of the test.
<code>simulate.p.value</code>	the value of <code>simulate.p.value</code> argument (returned on for class <code>GoF.gpcm</code>).
<code>df</code>	the degrees of freedom for the asymptotic chi-squared distribution (returned on for class <code>GoF.gpcm</code>).

Author(s)

Dimitris Rizopoulos (d.rizopoulos@erasmusmc.nl)

See Also

[person.fit](#), [item.fit](#), [margins](#), [gpcm](#), [rasch](#)

Examples

```
## GoF for the Rasch model for the LSAT data:
fit <- rasch(LSAT)
GoF.rasch(fit)
```

gpcm

Generalized Partial Credit Model - Polytomous IRT

Description

Fits the Generalized Partial Credit model for ordinal polytomous data, under the Item Response Theory approach.

Usage

```
gpcm(data, constraint = c("gpcm", "1PL", "rasch"), IRT.param = TRUE,
      start.val = NULL, na.action = NULL, control = list())
```

Arguments

<code>data</code>	a <code>data.frame</code> or a numeric matrix of manifest variables.
<code>constraint</code>	a character string specifying which version of the Generalized Partial Credit Model to fit. See Details and Examples for more info.
<code>IRT.param</code>	logical; if <code>TRUE</code> then the coefficients' estimates are reported under the usual IRT parameterization. See Details for more info.
<code>start.val</code>	a list of starting values or the character string "random". If a list, each one of its elements corresponds to each item and should contain a numeric vector with initial values for the threshold parameters and discrimination parameter; even if <code>constraint = "rasch"</code> or <code>constraint = "1PL"</code> , the discrimination parameter should be provided for all the items. If "random", random starting values are computed.
<code>na.action</code>	the <code>na.action</code> to be used on <code>data</code> ; default <code>NULL</code> the model uses the available cases, i.e., it takes into account the observed part of sample units with missing values (valid under MAR mechanisms if the model is correctly specified).
<code>control</code>	a named list of control values with components, <ul style="list-style-type: none"> iter.qN the number of quasi-Newton iterations. Default 150. GHk the number of Gauss-Hermite quadrature points. Default 21. optimizer which optimization routine to use; options are "optim" and "nlminb", the latter being the default. optimMethod the optimization method to be used in <code>optim()</code>. Default is "BFGS". numrDeriv which numerical derivative algorithm to use to approximate the Hessian matrix; options are "fd" for forward difference approximation and "cd" for central difference approximation. Default is "fd". epsHes step size to be used in the numerical derivative. Default is 1e-06. If you choose <code>numrDeriv = "cd"</code>, then change this to a larger value, e.g., 1e-03 or 1e-04. parscale the <code>parscale</code> control argument of <code>optim()</code>. Default is 0.5 for all parameters. verbose logical; if <code>TRUE</code> info about the optimization procedure are printed.

Details

The Generalized Partial Credit Model is an IRT model, that can handle ordinal manifest variables. This model was discussed by Masters (1982) and it was extended by Muraki (1992).

The model is defined as follows

$$P_{ik}(z) = \frac{\exp \sum_{c=0}^k \beta_i(z - \beta_{ic}^*)}{\sum_{r=0}^{m_i} \exp \sum_{c=0}^r \beta_i(z - \beta_{ic}^*)},$$

where $P_{ik}(z)$ denotes the probability of responding in category k for item i , given the latent ability z , β_{ic}^* are the item-category parameters, β_i is the discrimination parameter, m_i is the number of

categories for item i , and

$$\sum_{c=0}^0 \beta_i(z - \beta_{ic}^*) \equiv 0.$$

If `constraint = "rasch"`, then the discrimination parameter β_i is assumed equal for all items and fixed at one. If `constraint = "1PL"`, then the discrimination parameter β_i is assumed equal for all items but is estimated. If `constraint = "gpcm"`, then each item has its one discrimination parameter β_i that is estimated. See **Examples** for more info.

If `IRT.param = FALSE`, then the linear predictor is of the form $\beta_i z + \beta_{ic}$.

The fit of the model is based on approximate marginal Maximum Likelihood, using the Gauss-Hermite quadrature rule for the approximation of the required integrals.

Value

An object of class `gpcm` with components,

<code>coefficients</code>	a named list with components the parameter values at convergence for each item.
<code>log.Lik</code>	the log-likelihood value at convergence.
<code>convergence</code>	the convergence identifier returned by <code>optim()</code> or <code>nlminb()</code> .
<code>hessian</code>	the approximate Hessian matrix at convergence.
<code>counts</code>	the number of function and gradient evaluations used by the quasi-Newton algorithm.
<code>patterns</code>	a list with two components: (i) <code>X</code> : a numeric matrix that contains the observed response patterns, and (ii) <code>obs</code> : a numeric vector that contains the observed frequencies for each observed response pattern.
<code>GH</code>	a list with two components used in the Gauss-Hermite rule: (i) <code>Z</code> : a numeric matrix that contains the abscissas, and (ii) <code>GHw</code> : a numeric vector that contains the corresponding weights.
<code>max.sc</code>	the maximum absolute value of the score vector at convergence.
<code>constraint</code>	the value of the <code>constraint</code> argument.
<code>IRT.param</code>	the value of the <code>IRT.param</code> argument.
<code>X</code>	a copy of the response data matrix.
<code>control</code>	the values used in the <code>control</code> argument.
<code>na.action</code>	the value of the <code>na.action</code> argument.
<code>call</code>	the matched call.

Warning

In case the Hessian matrix at convergence is not positive definite try to re-fit the model by specifying the starting values or using `start.val = "random"`.

Note

`gpcm()` can also handle binary items and can be used instead of `rasch` and `ltm` though it is less efficient. However, `gpcm()` can handle a mix of dichotomous and polytomous items that neither `rasch` nor `ltm` can.

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

References

- Masters, G. (1982). A Rasch model for partial credit scoring. *Psychometrika*, **47**, 149–174.
- Muraki, E. (1992). A generalized partial credit model: application of an EM algorithm. *Applied Psychological Measurement*, **16**, 159–176.

See Also

[coef.gpcm](#), [fitted.gpcm](#), [summary.gpcm](#), [anova.gpcm](#), [plot.gpcm](#), [vcov.gpcm](#), [GoF.gpcm](#), [margins](#), [factor.scores](#)

Examples

```
## The Generalized Partial Credit Model for the Science data:
gpcm(Science[c(1,3,4,7)])

## The Generalized Partial Credit Model for the Science data,
## assuming equal discrimination parameters across items:
gpcm(Science[c(1,3,4,7)], constraint = "1PL")

## The Generalized Partial Credit Model for the Science data,
## assuming equal discrimination parameters across items
## fixed at 1:
gpcm(Science[c(1,3,4,7)], constraint = "rasch")

## more examples can be found at:
## http://wiki.r-project.org/rwiki/doku.php?id=packages:cran:ltm#sample_analyses
```

grm

Graded Response Model - Polytomous IRT

Description

Fits the Graded Response model for ordinal polytomous data, under the Item Response Theory approach.

Usage

```
grm(data, constrained = FALSE, IRT.param = TRUE, Hessian = FALSE,
     start.val = NULL, na.action = NULL, control = list())
```

Arguments

<code>data</code>	a <code>data.frame</code> (that will be converted to a numeric matrix using <code>data.matrix()</code>) or a numeric matrix of manifest variables.
<code>constrained</code>	logical; if <code>TRUE</code> the model with equal discrimination parameters across items is fitted. See Examples for more info.
<code>IRT.param</code>	logical; if <code>TRUE</code> then the coefficients' estimates are reported under the usual IRT parameterization. See Details for more info.
<code>Hessian</code>	logical; if <code>TRUE</code> the Hessian matrix is computed.
<code>start.val</code>	a list of starting values or the character string "random". If a list, each one of its elements corresponds to each item and should contain a numeric vector with initial values for the extremity parameters and discrimination parameter; even if <code>constrained = TRUE</code> the discrimination parameter should be provided for all the items. If "random" random starting values are computed.
<code>na.action</code>	the <code>na.action</code> to be used on <code>data</code> ; default <code>NULL</code> the model uses the available cases, i.e., it takes into account the observed part of sample units with missing values (valid under MAR mechanisms if the model is correctly specified)..
<code>control</code>	a list of control values, iter.qn the number of quasi-Newton iterations. Default 150. GHk the number of Gauss-Hermite quadrature points. Default 21. method the optimization method to be used in <code>optim()</code> . Default "BFGS". verbose logical; if <code>TRUE</code> info about the optimization procedure are printed. digits.abbrev numeric value indicating the number of digits used in abbreviating the Item's names. Default 6.

Details

The Graded Response Model is a type of polytomous IRT model, specifically designed for ordinal manifest variables. This model was first discussed by Samejima (1969) and it is mainly used in cases where the assumption of ordinal levels of response options is plausible.

The model is defined as follows

$$\log\left(\frac{\gamma_{ik}}{1 - \gamma_{ik}}\right) = \beta_{ik} - \beta_i z,$$

where γ_{ik} denotes the cumulative probability of a response in category k th or lower to the i th item, given the latent ability z . If `constrained = TRUE` it is assumed that $\beta_i = \beta$ for all i .

If `IRT.param = TRUE`, then the parameters estimates are reported under the usual IRT parameterization, i.e.,

$$\log\left(\frac{\gamma_{ik}}{1 - \gamma_{ik}}\right) = \beta_i(z - \beta_{ik}^*),$$

where $\beta_{ik}^* = \beta_{ik}/\beta_i$.

The fit of the model is based on approximate marginal Maximum Likelihood, using the Gauss-Hermite quadrature rule for the approximation of the required integrals.

Value

An object of class `grm` with components,

<code>coefficients</code>	a named list with components the parameter values at convergence for each item. These are always the estimates of β_{ik}, β_i parameters, even if <code>IRT.param = TRUE</code> .
<code>log.Lik</code>	the log-likelihood value at convergence.
<code>convergence</code>	the convergence identifier returned by <code>optim()</code> .
<code>hessian</code>	the approximate Hessian matrix at convergence returned by <code>optim()</code> ; returned only if <code>Hessian = TRUE</code> .
<code>counts</code>	the number of function and gradient evaluations used by the quasi-Newton algorithm.
<code>patterns</code>	a list with two components: (i) <code>X</code> : a numeric matrix that contains the observed response patterns, and (ii) <code>obs</code> : a numeric vector that contains the observed frequencies for each observed response pattern.
<code>GH</code>	a list with two components used in the Gauss-Hermite rule: (i) <code>Z</code> : a numeric matrix that contains the abscissas, and (ii) <code>GHw</code> : a numeric vector that contains the corresponding weights.
<code>max.sc</code>	the maximum absolute value of the score vector at convergence.
<code>constrained</code>	the value of the <code>constrained</code> argument.
<code>IRT.param</code>	the value of the <code>IRT.param</code> argument.
<code>X</code>	a copy of the response data matrix.
<code>control</code>	the values used in the <code>control</code> argument.
<code>na.action</code>	the value of the <code>na.action</code> argument.
<code>call</code>	the matched call.

Warning

In case the Hessian matrix at convergence is not positive definite try to re-fit the model, using `start.val = "random"`.

Note

`grm()` returns the parameter estimates such that the discrimination parameter for the first item β_1 is positive.

When the coefficients' estimates are reported under the usual IRT parameterization (i.e., `IRT.param = TRUE`), their standard errors are calculated using the Delta method.

`grm()` can also handle binary items, which should be coded as '1, 2' instead of '0, 1'.

Some parts of the code used for the calculation of the log-likelihood and the score vector have been based on `polr()` from package MASS.

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

References

- Baker, F. and Kim, S-H. (2004) *Item Response Theory*, 2nd ed. New York: Marcel Dekker.
- Samejima, F. (1969). Estimation of latent ability using a response pattern of graded scores. *Psychometrika Monograph Supplement*, **34**, 100–114.
- Rizopoulos, D. (2006) **ltm**: An R package for latent variable modelling and item response theory analyses. *Journal of Statistical Software*, **17(5)**, 1–25. URL <http://www.jstatsoft.org/v17/i05/>

See Also

[coef.grm](#), [fitted.grm](#), [summary.grm](#), [anova.grm](#), [plot.grm](#), [vcov.grm](#), [margins](#), [factor.scores](#)

Examples

```
## The Graded Response model for the Science data:
grm(Science[c(1, 3, 4, 7)])

## The Graded Response model for the Science data,
## assuming equal discrimination parameters across items:
grm(Science[c(1, 3, 4, 7)], constrained = TRUE)

## The Graded Response model for the Environment data
grm(Environment)
```

information

Area under the Test or Item Information Curves

Description

Computes the amount of test or item information for a fitted IRT model, in a specified range.

Usage

```
information(object, range, items = NULL, ...)
```

Arguments

<code>object</code>	an object inheriting from either class <code>gpcm</code> , class <code>grm</code> , class <code>ltm</code> , class <code>rasch</code> or class <code>tpm</code> .
<code>range</code>	a numeric interval for which the test information should be computed.
<code>items</code>	the items for which the information should be computed; the default <code>NULL</code> corresponds to all the items, which is equivalent to the test information.
<code>...</code>	extra arguments passed to <code>integrate()</code> .

Details

The amount of information is computed as the area under the Item or Test Information Curve in the specified interval, using `integrate()`.

Value

A list of class `information` with components,

<code>InfoRange</code>	the amount of information in the specified interval.
<code>InfoTotal</code>	the total amount of information; typically this is computed as the amount of information in the interval $(-10, 10)$.
<code>PropRange</code>	the proportion of information in the specified range, i.e., "Info in range" / "Total Info".
<code>range</code>	the value of <code>range</code> argument.
<code>items</code>	the value of <code>items</code> argument.
<code>call</code>	the matched call for object.

Author(s)

Dimitris Rizopoulos (d.rizopoulos@erasmusmc.nl)

See Also

[plot.gpcm](#), [plot.grm](#), [plot.ltm](#), [plot.rasch](#)

Examples

```
fit <- rasch(LSAT)
information(fit, c(-2, 0))
information(fit, c(0, 2), items = c(3, 5))
```

item.fit

Item-Fit Statistics and P-values

Description

Computation of item fit statistics for `ltm`, `rasch` and `tpm` models.

Usage

```
item.fit(object, G = 10, FUN = median,
         simulate.p.value = FALSE, B = 100)
```

Arguments

<code>object</code>	a model object inheriting either from class <code>ltm</code> , class <code>rasch</code> or class <code>tpm</code> .
<code>G</code>	either a number or a numeric vector. If a number, then it denotes the number of categories sample units are grouped according to their ability estimates.
<code>FUN</code>	a function to summarize the ability estimate with each group (e.g., median, mean, etc.).
<code>simulate.p.value</code>	logical; if <code>TRUE</code> , then the Monte Carlo procedure described in the Details section is used to approximate the the distribution of the item-fit statistic under the null hypothesis.
<code>B</code>	the number of replications in the Monte Carlo procedure.

Details

The item-fit statistic computed by `item.fit()` has the form:

$$\sum_{j=1}^G \frac{N_j(O_{ij} - E_{ij})^2}{E_{ij}(1 - E_{ij})},$$

where i is the item, j is the interval created by grouping sample units on the basis of their ability estimates, G is the number of sample units groupings (i.e., `G` argument), N_j is the number of sample units with ability estimates falling in a given interval j , O_{ij} is the observed proportion of keyed responses on item i for interval j , and E_{ij} is the expected proportion of keyed responses on item i for interval j based on the IRT model (i.e., `object`) evaluated at the ability estimate z^* within the interval, with z^* denoting the result of `FUN` applied to the ability estimates in group j .

If `simulate.p.value = FALSE`, then the p -values are computed assuming a chi-squared distribution with degrees of freedom equal to the number of groups `G` minus the number of estimated parameters. If `simulate.p.value = TRUE`, a Monte Carlo procedure is used to approximate the distribution of the item-fit statistic under the null hypothesis. In particular, the following steps are replicated `B` times:

Step 1: Simulate a new data-set of dichotomous responses under the assumed IRT model, using the maximum likelihood estimates $\hat{\theta}$ in the original data-set, extracted from `object`.

Step 2: Fit the model to the simulated data-set, extract the maximum likelihood estimates θ^* and compute the ability estimates z^* for each response pattern.

Step 3: For the new data-set, and using z^* and θ^* , compute the value of the item-fit statistic.

Denote by T_{obs} the value of the item-fit statistic for the original data-set. Then the p -value is approximated according to the formula

$$\left(1 + \sum_{b=1}^B I(T_b \geq T_{obs}) \right) / (1 + B),$$

where $I(\cdot)$ denotes the indicator function, and T_b denotes the value of the item-fit statistic in the b th simulated data-set.

Value

An object of class `itemFit` is a list with components,

<code>Tobs</code>	a numeric vector with item-fit statistics.
<code>p.values</code>	a numeric vector with the corresponding p -values.
<code>G</code>	the value of the <code>G</code> argument.
<code>simulate.p.value</code>	the value of the <code>simulate.p.value</code> argument.
<code>B</code>	the value of the <code>B</code> argument.
<code>call</code>	a copy of the matched call of object.

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

References

Reise, S. (1990) A comparison of item- and person-fit methods of assessing model-data fit in IRT. *Applied Psychological Measurement*, **14**, 127–137.

Yen, W. (1981) Using simulation results to choose a latent trait model. *Applied Psychological Measurement*, **5**, 245–262.

See Also

[person.fit](#), [margins](#), [GoF.gpcm](#), [GoF.rasch](#)

Examples

```
# item-fit statistics for the Rasch model
# for the Abortion data-set
item.fit(rasch(Abortion))

# Yen's Q1 item-fit statistic (i.e., 10 latent ability groups; the
# mean ability in each group is used to compute fitted proportions)
# for the two-parameter logistic model for the LSAT data-set
item.fit(ltm(LSAT ~ z1), FUN = mean)
```

LSAT

The Law School Admission Test (LSAT), Section VI

Description

The LSAT is a classical example in educational testing for measuring ability traits. This test was designed to measure a *single* latent ability scale.

Format

A data frame with the responses of 1000 individuals to 5 questions.

Source

This LSAT example is a part of a data set given in Bock and Lieberman (1970).

References

Bartholomew, D., Steel, F., Moustaki, I. and Galbraith, J. (2002) *The Analysis and Interpretation of Multivariate Data for Social Scientists*. London: Chapman and Hall.

Bock, R. and Lieberman, M. (1970) Fitting a response model for n dichotomously scored items. *Psychometrika*, **35**, 179–197.

Examples

```
## Descriptive statistics for LSAT data
dsc <- descript(LSAT)
dsc
plot(dsc)
```

ltm

Latent Trait Model - Latent Variable Model for Binary Data

Description

Fit a latent trait model under the Item Response Theory (IRT) approach.

Usage

```
ltm(formula, constraint = NULL, IRT.param, start.val,
     na.action = NULL, control = list())
```

Arguments

<code>formula</code>	a two-sided formula providing the responses data matrix and describing the latent structure. In the left side of <code>formula</code> either a <code>data.frame</code> (that will be converted to a numeric matrix using <code>data.matrix()</code>) or a numeric matrix of manifest variables must be supplied. In the right side of <code>formula</code> only two latent variables are allowed with codenames <code>z1</code> , <code>z2</code> . Interaction and quadratic terms can also be used (see Details and Examples for more info).
<code>constraint</code>	a three-column numeric matrix with at most $pq - 1$ rows (where p is the number of items and q the number of latent components plus the intercept), specifying fixed-value constraints. The first column represents the item (i.e., 1 denotes the first item, 2 the second, etc.), the second column represents the component of the latent structure (i.e., 1 denotes the intercept β_{0i} , 2 the loadings of the first factor β_{1i} , etc.) and the third column denotes the value at which the corresponding parameter should be fixed. See Details and Examples for more info.
<code>IRT.param</code>	logical; if <code>TRUE</code> then the coefficients' estimates for the two-parameter logistic model are reported under the usual IRT parameterization. See Details for more info.
<code>start.val</code>	the character string "random" or a numeric matrix supplying starting values with p rows and q columns, with p denoting the number of items, and q denoting the number of terms in the right-hand side of <code>formula</code> . If <code>NULL</code> starting values are automatically computed. If "random", random starting values are used. If a matrix, then depending on the latent structure specified in <code>formula</code> , the first column should contain β_{0i} , the second β_{1i} , the third β_{2i} , and the remaining columns $\beta_{nl,i}$ (see Details)
<code>na.action</code>	the <code>na.action</code> to be used on the data frame in the left side of <code>formula</code> . In case of missing data, if <code>na.action = NULL</code> the model uses the available cases, i.e., it takes into account the observed part of sample units with missing values (valid under MAR mechanisms if the model is correctly specified). If you want to apply a complete case analysis then use <code>na.action = na.exclude</code> .
<code>control</code>	a list of control values, iter.em the number of EM iterations. Default 40. iter.qN the number of quasi-Newton iterations. Default 150. GHk the number of Gauss-Hermite quadrature points. Default 15. method the optimization method to be used in <code>optim()</code> . Default "BFGS". verbose logical; if <code>TRUE</code> info about the optimization procedure are printed.

Details

The latent trait model is the analogue of the factor analysis model for binary observed data. The model assumes that the dependencies between the observed response variables (known as items) can be interpreted by a small number of latent variables. The model formulation is under the IRT approach; in particular,

$$\log\left(\frac{\pi_i}{1 - \pi_i}\right) = \beta_{0i} + \beta_{1i}z_1 + \beta_{2i}z_2,$$

where π_i is the probability of a positive response in the i th item, β_{i0} is the easiness parameter, β_{ij} ($j = 1, 2$) are the discrimination parameters and z_1, z_2 denote the two latent variables.

The usual form of the latent trait model assumes linear latent variable effects (Bartholomew and Knott, 1999; Moustaki and Knott, 2000). `ltm()` fits the linear one- and two-factor models but also provides extensions described by Rizopoulos and Moustaki (2006) to include nonlinear latent variable effects. These are incorporated in the linear predictor of the model, i.e.,

$$\log\left(\frac{\pi_i}{1 - \pi_i}\right) = \beta_{0i} + \beta_{1i}z_1 + \beta_{2i}z_2 + \beta_{nl}^t f(z_1, z_2),$$

where $f(z_1, z_2)$ is a function of z_1 and z_2 (e.g., $f(z_1, z_2) = z_1z_2$, $f(z_1, z_2) = z_1^2$, etc.) and β_{nl} is a matrix of nonlinear terms parameters (look also at the **Examples**).

If `IRT.param = TRUE`, then the parameters estimates for the two-parameter logistic model (i.e., the model with one factor) are reported under the usual IRT parameterization, i.e.,

$$\log\left(\frac{\pi_i}{1 - \pi_i}\right) = \beta_{1i}(z - \beta_{0i}^*).$$

The linear two-factor model is unidentified under orthogonal rotations on the factors' space. To achieve identifiability you can fix the value of one loading using the `constraint` argument.

The parameters are estimated by maximizing the approximate marginal log-likelihood under the conditional independence assumption, i.e., conditionally on the latent structure the items are independent Bernoulli variates under the logit link. The required integrals are approximated using the Gauss-Hermite rule. The optimization procedure used is a hybrid algorithm. The procedure initially uses a moderate number of EM iterations (see `control` argument `iter.em`) and then switches to quasi-Newton (see `control` arguments `method` and `iter.qN`) iterations until convergence.

Value

An object of class `ltm` with components,

<code>coefficients</code>	a matrix with the parameter values at convergence. These are always the estimates of $\beta_{li}, l = 0, 1, \dots$ parameters, even if <code>IRT.param = TRUE</code> .
<code>log.Lik</code>	the log-likelihood value at convergence.
<code>convergence</code>	the convergence identifier returned by <code>optim()</code> .
<code>hessian</code>	the approximate Hessian matrix at convergence returned by <code>optim()</code> .
<code>counts</code>	the number of function and gradient evaluations used by the quasi-Newton algorithm.
<code>patterns</code>	a list with two components: (i) <code>X</code> : a numeric matrix that contains the observed response patterns, and (ii) <code>obs</code> : a numeric vector that contains the observed frequencies for each observed response pattern.
<code>GH</code>	a list with two components used in the Gauss-Hermite rule: (i) <code>Z</code> : a numeric matrix that contains the abscissas, and (ii) <code>GHw</code> : a numeric vector that contains the corresponding weights.
<code>max.sc</code>	the maximum absolute value of the score vector at convergence.
<code>ltst</code>	a list describing the latent structure.

<code>X</code>	a copy of the response data matrix.
<code>control</code>	the values used in the <code>control</code> argument.
<code>IRT.param</code>	the value of the <code>IRT.param</code> argument.
<code>constraint</code>	<code>if(!is.null(constraint))</code> , then it contains the value of the <code>constraint</code> argument.
<code>call</code>	the matched call.

Warning

In case the Hessian matrix at convergence is not positive definite, try to re-fit the model; `ltm()` will use new random starting values.

The inclusion of nonlinear latent variable effects produces more complex likelihood surfaces which might possess a number of local maxima. To ensure that the maximum likelihood value has been reached re-fit the model a number of times (simulations showed that usually 10 times are adequate to ensure global convergence).

Conversion of the parameter estimates to the usual IRT parameterization works only for the two-parameter logistic model.

Note

In the case of the one-factor model, the optimization algorithm works under the constraint that the discrimination parameter of the first item β_{11} is always positive. If you wish to change its sign, then in the fitted model, say `m`, use `m$coef[, 2] <- -m$coef[, 2]`.

When the coefficients' estimates are reported under the usual IRT parameterization (i.e., `IRT.param = TRUE`), their standard errors are calculated using the Delta method.

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

References

- Baker, F. and Kim, S-H. (2004) *Item Response Theory*, 2nd ed. New York: Marcel Dekker.
- Bartholomew, D. and Knott, M. (1999) *Latent Variable Models and Factor Analysis*, 2nd ed. London: Arnold.
- Bartholomew, D., Steel, F., Moustaki, I. and Galbraith, J. (2002) *The Analysis and Interpretation of Multivariate Data for Social Scientists*. London: Chapman and Hall.
- Moustaki, I. and Knott, M. (2000) Generalized latent trait models. *Psychometrika*, **65**, 391–411.
- Rizopoulos, D. (2006) **ltm**: An R package for latent variable modelling and item response theory analyses. *Journal of Statistical Software*, **17(5)**, 1–25. URL <http://www.jstatsoft.org/v17/i05/>
- Rizopoulos, D. and Moustaki, I. (2008) Generalized latent variable models with nonlinear effects. *British Journal of Mathematical and Statistical Psychology*, **61**, 415–438.

See Also

[coef.ltm](#), [fitted.ltm](#), [summary.ltm](#), [anova.ltm](#), [plot.ltm](#), [vcov.ltm](#), [item.fit](#), [person.fit](#), [margins](#), [factor.scores](#)

Examples

```
## The two-parameter logistic model for the WIRS data
## with the constraint that (i) the easiness parameter
## for the 1st item equals 1 and (ii) the discrimination
## parameter for the 6th item equals -0.5

ltm(WIRS ~ z1, constr = rbind(c(1, 1, 1), c(6, 2, -0.5)))

## One-factor and a quadratic term
## using the Mobility data
ltm(Mobility ~ z1 + I(z1^2))

## Two-factor model with an interaction term
## using the WIRS data
ltm(WIRS ~ z1 * z2)

## The two-parameter logistic model for the Abortion data
## with 20 quadrature points and 20 EM iterations;
## report results under the usual IRT parameterization
ltm(Abortion ~ z1, control = list(GHk = 20, iter.em = 20))
```

margins

Fit of the model on the margins

Description

Checks the fit on the two- and three-way margins for `grm`, `ltm`, `rasch` and `tpm` objects.

Usage

```
margins(object, ...)

## S3 method for class 'gpcm':
margins(object, type = c("two-way", "three-way"), rule = 3.5, ...)

## S3 method for class 'grm':
margins(object, type = c("two-way", "three-way"), rule = 3.5, ...)

## S3 method for class 'ltm':
margins(object, type = c("two-way", "three-way"), rule = 3.5,
        nprint = 3, ...)
```

```
## S3 method for class 'rasch':
margins(object, type = c("two-way", "three-way"), rule = 3.5,
        nprint = 3, ...)

## S3 method for class 'tpm':
margins(object, type = c("two-way", "three-way"), rule = 3.5,
        nprint = 3, ...)
```

Arguments

object	an object inheriting either from class <code>gpcm</code> , class <code>grm</code> , class <code>ltm</code> or class <code>rasch</code> .
type	the type of margins to be used. See Details for more info.
rule	the rule of thumb used in determining the indicative goodness-of-fit.
nprint	a numeric value determining the number of margins with the largest Chi-squared residuals to be printed; only for <code>ltm</code> and <code>rasch</code> objects.
...	additional argument; currently none is used.

Details

Rather than looking at the whole set of response patterns, we can look at the two- and three-way margins. For the former, we construct the 2×2 contingency tables obtained by taking the variables two at a time. Comparing the observed and expected two-way margins is analogous to comparing the observed and expected correlations when judging the fit of a factor analysis model. For Bernoulli and Ordinal variates, the comparison is made using the so called Chi-squared residuals. As a rule of thumb residuals greater than 3.5 are indicative of poor fit. For a more strict rule of thumb use the `rule` argument. The analogous procedure is followed for the three-way margins.

Value

An object of either class `margins.ltm` if `object` inherits from class `ltm`, class `rasch` or class `tpm`, or an object of class `margins.grm` if `object` inherits from class `grm`, with components,

margins	for <code>margins.ltm</code> is an array containing the values of chi-squared residuals; for <code>margins.gpcm</code> and <code>margins.grm</code> is a list of length either the number of all possible pairs or all possible triplets of items, containing the observed and expected frequencies, the values of chi-squared residuals, the value of the total residual and the value of the rule of thumb times the product of the number of categories of the items under consideration.
type	the type of margins that were calculated.
nprint	the value of the <code>nprint</code> argument; returned only from <code>margins.ltm</code> .
combs	all possible two- or three-way combinations of the items; returned only from <code>margins.ltm</code> .
rule	the value of the <code>rule</code> argument; returned only from <code>margins.ltm</code> .
nitems	the number of items in <code>object</code> ; returned only from <code>margins.grm</code> .
names	the names of items in <code>object</code> ; returned only from <code>margins.grm</code> .
call	a copy of the matched call of <code>object</code> .

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

References

- Bartholomew, D. (1998) Scaling unobservable constructs in social science. *Applied Statistics*, **47**, 1–13.
- Bartholomew, D. and Knott, M. (1999) *Latent Variable Models and Factor Analysis*, 2nd ed. London: Arnold.
- Bartholomew, D., Steel, F., Moustaki, I. and Galbraith, J. (2002) *The Analysis and Interpretation of Multivariate Data for Social Scientists*. London: Chapman and Hall.
- Rizopoulos, D. (2006) **Itm**: An R package for latent variable modelling and item response theory analyses. *Journal of Statistical Software*, **17(5)**, 1–25. URL <http://www.jstatsoft.org/v17/i05/>

See Also

[person.fit](#), [item.fit](#), [GoF.rasch](#),

Examples

```
## Two- and Three-way residuals for the Rasch model
fit <- rasch(LSAT)
margins(fit)
margins(fit, "three")

## Two- and Three-way residuals for the one-factor model
fit <- ltm(WIRS ~ z1)
margins(fit)
margins(fit, "three")

## Two- and Three-way residuals for the graded response model
fit <- grm(Science[c(1,3,4,7)])
margins(fit)
margins(fit, "three")
```

Description

A rural subsample of 8445 women from the Bangladesh Fertility Survey of 1989.

Format

The dimension of interest is women's mobility of social freedom. Women were asked whether they could engage in the following activities alone (1 = yes, 0 = no):

Item 1 Go to any part of the village/town/city.

Item 2 Go outside the village/town/city.

Item 3 Talk to a man you do not know.

Item 4 Go to a cinema/cultural show.

Item 5 Go shopping.

Item 6 Go to a cooperative/mothers' club/other club.

Item 7 Attend a political meeting.

Item 8 Go to a health centre/hospital.

Source

Bangladesh Fertility Survey of 1989 (Huq and Cleland, 1990).

References

Bartholomew, D., Steel, F., Moustaki, I. and Galbraith, J. (2002) *The Analysis and Interpretation of Multivariate Data for Social Scientists*. London: Chapman and Hall.

Huq, N. and Cleland, J. (1990) *Bangladesh Fertility Survey, 1989*. Dhaka: National Institute of Population Research and Training (NIPORT).

Examples

```
## Descriptive statistics for Mobility data
descript(Mobility)
```

mult.choice

Multiple Choice Items to Binary Responses

Description

It converts multiple choice items to a matrix of binary responses.

Usage

```
mult.choice(data, correct)
```

Arguments

`data` a matrix or a `data.frame` containing the manifest variables as columns.
`correct` a vector of length `ncol(data)` with the correct responses.

Value

a matrix of 0/1 values indicating wrong/correct answers.

Author(s)

Dimitris Rizopoulos (d.rizopoulos@erasmusmc.nl)

Examples

```
dat <- data.frame(It1 = sample(4, 100, TRUE),
                 It2 = sample(4, 100, TRUE),
                 It3 = sample(5, 100, TRUE),
                 It4 = sample(5, 100, TRUE),
                 It5 = sample(4, 100, TRUE),
                 It6 = sample(5, 100, TRUE))
dat[] <- lapply(dat, function (x) { x[sample(100, 4)] <- NA; x })
crcr <- c(3, 2, 5, 3, 4, 5)
#####
mult.choice(dat, crcr)
```

person.fit

Person-Fit Statistics and P-values

Description

Computation of person fit statistics for ltm, rasch and tpm models.

Usage

```
person.fit(object, alternative = c("less", "greater", "two.sided"),
          resp.patterns = NULL, FUN = NULL, simulate.p.value = FALSE,
          B = 1000)
```

Arguments

object a model object inheriting either from class ltm, class rasch or class tpm.

alternative the alternative hypothesis; see **Details** for more info.

resp.patterns a matrix or a data.frame of response patterns with columns denoting the items; if NULL the person fit statistics are computed for the observed response patterns.

FUN a function with three arguments calculating a user-defined person-fit statistic. The first argument must be a numeric matrix of (0, 1) response patterns. The second argument must be a numeric vector of length equal to the number of rows of the first argument, providing the ability estimates for each response pattern. The third argument must be a numeric matrix with number of rows equal to the

number of items, providing the IRT model parameters. For `ltm` and `rasch` objects, this should be a two-column matrix, where the first column contains the easiness and the second one the discrimination parameters (i.e., the additive parameterization is assumed, which has the form $\beta_{i0} + \beta_{i1}z$, where β_{i0} is the easiness and β_{i1} the discrimination parameter for the i th item). For `tpm` objects the first column of the third argument of `FUN` should contain the logit (i.e., use `qlogis()`) of the guessing parameters, the second column the easiness, and the third column the discrimination parameters. The function should return a numeric vector of length equal to the number of response patterns, containing the values of the user-defined person-fit statistics.

`simulate.p.value`

logical; if `TRUE`, then the Monte Carlo procedure described in the **Details** section is used to approximate the the distribution of the person-fit statistic(s) under the null hypothesis.

`B`

the number of replications in the Monte Carlo procedure.

Details

The statistics calculated by default (i.e., if `FUN = NULL`) by `person.fit()` are the L_0 statistic of Levine and Rubin (1979) and its standardized version L_z proposed by Drasgow et al. (1985). If `simulate.p.value = FALSE`, the p -values are calculated for the L_z assuming a standard normal distribution for the statistic under the null. If `simulate.p.value = TRUE`, a Monte Carlo procedure is used to approximate the distribution of the person-fit statistic(s) under the null hypothesis. In particular, the following steps are replicated `B` times for each response pattern:

Step 1: Simulate a new ability estimate, say z^* , from a normal distribution with mean the ability estimate of the response pattern under the fitted model (i.e., `object`), and standard deviation the standard error of the ability estimate, as returned by the `factor.scores` function.

Step 2: Simulate a new response pattern of dichotomous items under the assumed IRT model, using z^* and the maximum likelihood estimates under `object`.

Step 4: For the new response pattern and using z^* and the MLEs, compute the values of the person-fit statistic.

Denote by T_{obs} the value of the person-fit statistic for the original data-set. Then the p -value is approximated according to the formula

$$\left(1 + \sum_{b=1}^B I(T_b \leq T_{obs})\right) / (1 + B),$$

if `alternative = "less"`,

$$\left(1 + \sum_{b=1}^B I(T_b \geq T_{obs})\right) / (1 + B),$$

if `alternative = "greater"`, or

$$\left(1 + \sum_{b=1}^B I(|T_b| \geq |T_{obs}|)\right) / (1 + B),$$

if `alternative = "two.sided"`, where T_b denotes the value of the person-fit statistic in the b th simulated data-set, $I(\cdot)$ denotes the indicator function, and $|\cdot|$ denotes the absolute value. For the L_z statistic, negative values (i.e., `alternative = "less"`) indicate response patterns that are unlikely, given the measurement model and the ability estimate. Positive values (i.e., `alternative = "greater"`) indicate that the examinee's response pattern is more consistent than the probabilistic IRT model expected. Finally, when `alternative = "two.sided"` both the above settings are captured.

This simulation scheme explicitly accounts for the fact that ability values are estimated, by drawing from their large sample distribution. Strictly speaking, drawing z^* from a normal distribution is not theoretically appropriate, since the posterior distribution for the latent abilities is not normal. However, the normality assumption will work reasonably well, especially when a large number of items is considered.

Value

An object of class `persFit` is a list with components,

<code>resp.patterns</code>	the response patterns for which the fit statistics have been computed.
<code>Tobs</code>	a numeric matrix with person-fit statistics for each response pattern.
<code>p.values</code>	a numeric matrix with the corresponding p -values.
<code>statistic</code>	the value of the <code>statistic</code> argument.
<code>FUN</code>	the value of the <code>FUN</code> argument.
<code>alternative</code>	the value of the <code>alternative</code> argument.
<code>B</code>	the value of the <code>B</code> argument.
<code>call</code>	a copy of the matched call of <code>object</code> .

Author(s)

Dimitris Rizopoulos (d.rizopoulos@erasmusmc.nl)

References

- Drasgow, F., Levine, M. and Williams, E. (1985) Appropriateness measurement with polychotomous item response models and standardized indices. *British Journal of Mathematical and Statistical Psychology*, **38**, 67–86.
- Levine, M. and Rubin, D. (1979) Measuring the appropriateness of multiple-choice test scores. *Journal of Educational Statistics*, **4**, 269–290.
- Meijer, R. and Sijtsma, K. (2001) Methodology review: Evaluating person fit. *Applied Psychological Measurement*, **25**, 107–135.
- Reise, S. (1990) A comparison of item- and person-fit methods of assessing model-data fit in IRT. *Applied Psychological Measurement*, **14**, 127–137.

See Also

[item.fit](#), [margins](#), [GoF.gpcm](#), [GoF.rasch](#)

Examples

```
# person-fit statistics for the Rasch model
# for the Abortion data-set
person.fit(rasch(Abortion))

# person-fit statistics for the two-parameter logistic model
# for the LSAT data-set
person.fit(ltm(LSAT ~ z1), simulate.p.value = TRUE, B = 100)
```

plot descript	<i>Descriptive Statistics Plot method</i>
---------------	---

Description

The plot method for `descript` objects currently works for dichotomous response patterns, and produces the xy-plot of the total score versus the proportion of correct responses for each item.

Usage

```
## S3 method for class 'descript':
plot(x, items = NULL, includeFirstLast = FALSE, xlab, ylab, ...)
```

Arguments

<code>x</code>	an object inheriting from class <code>descript</code> .
<code>items</code>	a numeric vector indicating which items to plot.
<code>includeFirstLast</code>	logical; if <code>TRUE</code> the first and last total scores categories are included.
<code>xlab, ylab</code>	character string or an expression ; see title .
<code>...</code>	extra graphical parameters to be passed to <code>matplot()</code> .

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

See Also

[descript](#)

Examples

```
## Descriptives for WIRS data:
dsc <- descript(WIRS)
dsc
plot(dsc, includeFirstLast = TRUE, type = "b", lty = 1, pch = 1:6)
legend("topleft", names(WIRS), pch = 1:6, col = 1:6, lty = 1, bty = "n")
```

plot fscores *Factor Scores - Ability Estimates Plot method*

Description

Plots a Kernel Density Estimation of the distribution of the factor scores (i.e., person parameters). Provides also the option to include in the plot the item difficulty parameters (similar to the Item Person Maps).

Usage

```
## S3 method for class 'fscores':
plot(x, bw = "nrd0", adjust = 2, kernel = "gaussian",
     include.items = FALSE, tol = 0.2, xlab = "Ability", ylab = "Density",
     main = "Kernel Density Estimation for Ability Estimates",
     pch = 16, cex = 1.5, ...)
```

Arguments

`x` an object inheriting from class `fscores`.

`bw`, `adjust`, `kernel` arguments to `density()`.

`include.items` logical; if `TRUE` the item difficulty parameters are included in the plot.

`tol` the tolerance used to group the item difficulty parameters, i.e., when `include.items = TRUE` the values `round(betas / tol) * tol` are plotted, where `beta` is the numeric vector of item difficulty parameters.

`xlab`, `ylab`, `main` character string or an [expression](#); see [title](#).

`pch`, `cex` arguments to `stripchart()`; used when `include.items = TRUE`.

`...` extra graphical parameters to be passed to `plot.density()`.

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

See Also[factor.scores](#)**Examples**

```
## Factor Scores for LSAT data:
fsc <- factor.scores(rasch(LSAT))
plot(fsc, include.items = TRUE, main = "KDE for Person Parameters")
legend("left", "item parameters", pch = 16, cex = 1.5, bty = "n")
```

plot IRT

*Plot method for fitted IRT models***Description**

Produces the Item Characteristic or Item Information Curves for fitted IRT models.

Usage

```
## S3 method for class 'gpcm':
plot(x, type = c("ICC", "IIC", "OCCu", "OCCl"), items = NULL,
     category = NULL, zrange = c(-3.8, 3.8),
     z = seq(zrange[1], zrange[2], length = 100), annot,
     labels = NULL, legend = FALSE, cx = "top", cy = NULL, ncol = 1,
     bty = "n", col = palette(), lty = 1, pch, xlab, ylab, main,
     sub = NULL, cex = par("cex"), cex.lab = par("cex.lab"),
     cex.main = par("cex.main"), cex.sub = par("cex.sub"),
     cex.axis = par("cex.axis"), plot = TRUE, ...)

## S3 method for class 'grm':
plot(x, type = c("ICC", "IIC", "OCCu", "OCCl"), items = NULL,
     category = NULL, zrange = c(-3.8, 3.8),
     z = seq(zrange[1], zrange[2], length = 100), annot,
     labels = NULL, legend = FALSE, cx = "top", cy = NULL, ncol = 1,
     bty = "n", col = palette(), lty = 1, pch, xlab, ylab, main,
     sub = NULL, cex = par("cex"), cex.lab = par("cex.lab"),
     cex.main = par("cex.main"), cex.sub = par("cex.sub"),
     cex.axis = par("cex.axis"), plot = TRUE, ...)

## S3 method for class 'ltm':
plot(x, type = c("ICC", "IIC", "loadings"), items = NULL,
     zrange = c(-3.8, 3.8), z = seq(zrange[1], zrange[2], length = 100),
     annot, labels = NULL, legend = FALSE, cx = "topleft", cy = NULL,
     ncol = 1, bty = "n", col = palette(), lty = 1, pch, xlab, ylab,
     zlab, main, sub = NULL, cex = par("cex"), cex.lab = par("cex.lab"),
```

```

cex.main = par("cex.main"), cex.sub = par("cex.sub"),
cex.axis = par("cex.axis"), plot = TRUE, ...)

## S3 method for class 'rasch':
plot(x, type = c("ICC", "IIC"), items = NULL,
     zrange = c(-3.8, 3.8), z = seq(zrange[1], zrange[2], length = 100),
     annot, labels = NULL, legend = FALSE, cx = "topleft", cy = NULL,
     ncol = 1, bty = "n", col = palette(), lty = 1, pch, xlab, ylab,
     main, sub = NULL, cex = par("cex"), cex.lab = par("cex.lab"),
     cex.main = par("cex.main"), cex.sub = par("cex.sub"),
     cex.axis = par("cex.axis"), plot = TRUE, ...)

## S3 method for class 'tpm':
plot(x, type = c("ICC", "IIC"), items = NULL,
     zrange = c(-3.8, 3.8), z = seq(zrange[1], zrange[2], length = 100),
     annot, labels = NULL, legend = FALSE, cx = "topleft", cy = NULL,
     ncol = 1, bty = "n", col = palette(), lty = 1, pch, xlab, ylab, main,
     sub = NULL, cex = par("cex"), cex.lab = par("cex.lab"),
     cex.main = par("cex.main"), cex.sub = par("cex.sub"),
     cex.axis = par("cex.axis"), plot = TRUE, ...)

```

Arguments

<code>x</code>	an object inheriting either from class <code>gpcm</code> , class <code>grm</code> , class <code>ltm</code> , class <code>rasch</code> or class <code>tpm</code> .
<code>type</code>	the type of plot; "ICC" refers to Item Response Category Characteristic Curves whereas "IIC" to Item Information Curves. For <code>ltm</code> objects the option "loadings" is also available that produces the scatter plot of the standardized loadings. For <code>grm</code> objects the options "OCCu" and "OCCI" are also available that produces the item operation characteristic curves.
<code>items</code>	a numeric vector denoting which items to plot; if <code>NULL</code> all items are plotted; if 0 and <code>type = "IIC"</code> the Test Information Curve is plotted.
<code>category</code>	a scalar indicating the response category for which the curves should be plotted; if <code>NULL</code> all categories are considered. This argument is only relevant for <code>grm</code> objects.
<code>zrange</code>	a numeric vector of length 2 indicating the range for the latent variable values.
<code>z</code>	a numeric vector denoting the values for the latent variable(s) values to be used in the plots.
<code>annot</code>	logical; if <code>TRUE</code> the plotted lines are annotated.
<code>labels</code>	character vector; the labels to use in either the annotation or legend. If <code>NULL</code> adequate labels are produced.
<code>legend</code>	logical; if <code>TRUE</code> a legend is printed.
<code>cx, cy, ncol, bty</code>	arguments of <code>legend</code> ; <code>cx</code> and <code>cy</code> correspond to the <code>x</code> and <code>y</code> arguments of <code>legend</code> .
<code>col, lty, pch</code>	control values, see <code>par</code> ; recycling is used if necessary.

`xlab`, `ylab`, `zlab`, `main`, `sub`
character string or an [expression](#); see [title](#).

`cex`, `cex.lab`, `cex.main`, `cex.sub`, `cex.axis`
the cex family of argument; see [par](#).

`plot`
logical; if TRUE the plot(s) is(are) produced otherwise only the values used to create the plot(s) are returned.

`...`
extra graphical parameters to be passed to `plot()`, `lines()`, `legend()` and `text()`.

Details

Item response category characteristic curves show how the probability of responding in the k th category, in each item, changes with the values of the latent variable (ability).

The item information curves indicate the relative ability of an item to discriminate among contiguous trait scores at various locations along the trait continuum. The test information curve, which is the sum of item information curves, provides a visual depiction of where along the trait continuum a test is most discriminating (Reise and Waller, 2002).

Value

The values used to create the plot, i.e., the x-, y-coordinates. This is either a matrix or a list in which the first column or element provides the latent variable values used, and the remaining columns or elements correspond to either probabilities or information or loadings, depending on the value of the `type` argument.

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

References

Reise, S. and Waller, N. (2002) Item response theory for dichotomous assessment data. In Drasgow, F. and Schmitt, N., editors, *Measuring and Analyzing Behavior in Organizations*. San Francisco: Jossey-Bass.

See Also

[information](#), [gpcm](#), [grm](#), [ltm](#), [rasch](#), [tpm](#)

Examples

```
# Examples for plot.grm()

fit <- grm(Science[c(1,3,4,7)])

## Item Response Category Characteristic Curves for
## the Science data
op <- par(mfrow = c(2, 2))
plot(fit, lwd = 2, legend = TRUE, ncol = 2)
```

```

# re-set par()
par(op)

## Item Characteristic Curves for the 2nd category,
## and items 1 and 3
plot(fit, category = 2, items = c(1, 3), lwd = 2, legend = TRUE, cx = "right")

## Item Information Curves for the Science data;
plot(fit, type = "IIC", legend = TRUE, cx = "topright", lwd = 2, cex = 1.4)

## Test Information Function for the Science data;
plot(fit, type = "IIC", items = 0, lwd = 2)

#####

# Examples for plot.ltm()

## Item Characteristic Curves for the two-parameter logistic
## model; plot only items 1, 2, 4 and 6; take the range of the
## latent ability to be (-2.5, 2.5):
fit <- ltm(WIRS ~ z1)
plot(fit, items = c(1, 2, 4, 6), zrange = c(-2.5, 2.5), lwd = 3, cex = 1.4)

## Test Information Function under the two-parameter logistic
## model for the Lsat data
fit <- ltm(LSAT ~ z1)
plot(fit, type = "IIC", items = 0, lwd = 2, cex.lab = 1.2, cex.main = 1.3)
info <- information(fit, c(-3, 0))
text(x = 2, y = 0.5, labels = paste("Total Information:", round(info$InfoTotal, 3),
  "\n\nInformation in (-3, 0):", round(info$InfoRange, 3),
  paste("(", round(100 * info$PropRange, 2), "%)", sep = ")), cex = 1.2)

## Item Characteristic Surfaces for the interaction model:
fit <- ltm(WIRS ~ z1 * z2)
plot(fit, ticktype = "detailed", theta = 30, phi = 30, expand = 0.5, d = 2,
  cex = 0.7, col = "lightblue")

#####

# Examples for plot.rasch()

## Item Characteristic Curves for the WIRS data;
## plot only items 1, 3 and 5:
fit <- rasch(WIRS)
plot(fit, items = c(1, 3, 5), lwd = 3, cex = 1.4)
abline(v = -4:4, h = seq(0, 1, 0.2), col = "lightgray", lty = "dotted")

fit <- rasch(LSAT)

## Item Characteristic Curves for the LSAT data;
## plot all items plus a legend and use only black:
plot(fit, legend = TRUE, cx = "right", lwd = 3, cex = 1.4,
  cex.lab = 1.6, cex.main = 2, col = 1, lty = c(1, 1, 1, 2, 2),

```

```

    pch = c(16, 15, 17, 0, 1))
abline(v = -4:4, h = seq(0, 1, 0.2), col = "lightgray", lty = "dotted")

## Item Information Curves, for the first 3 items; include a legend
plot(fit, type = "IIC", items = 1:3, legend = TRUE, lwd = 2, cx = "topright")

## Test Information Function
plot(fit, type = "IIC", items = 0, lwd = 2, cex.lab = 1.1,
     sub = paste("Call: ", deparse(fit$call)))

## Total information in (-2, 0) based on all the items
info.Tot <- information(fit, c(-2, 0))$InfoRange
## Information in (-2, 0) based on items 2 and 4
info.24 <- information(fit, c(-2, 0), items = c(2, 4))$InfoRange
text(x = 2, y = 0.5, labels = paste("Total Information in (-2, 0):",
    round(info.Tot, 3),
    "\n\nInformation in (-2, 0) based on\n Items 2 and 4:", round(info.24, 3),
    paste("(", round(100 * info.24 / info.Tot, 2), "%)", sep = "")),
     cex = 1.2)

## The Standard Error of Measurement can be plotted by
vals <- plot(fit, type = "IIC", items = 0, plot = FALSE)
plot(vals[, "z"], 1 / sqrt(vals[, "info"]), type = "l", lwd = 2,
     xlab = "Ability", ylab = "Standard Error",
     main = "Standard Error of Measurement")

#####

# Examples for plot.tpm()

## Compare the Item Characteristic Curves for the LSAT data,
## under the constraint Rasch model, the unconstraint Rasch model,
## and the three parameter model assuming equal discrimination
## across items
par(mfrow = c(2, 2))
p11 <- plot(rasch(LSAT, constr = cbind(length(LSAT) + 1, 1)))
text(2, 0.35, "Rasch model\nDiscrimination = 1")
p12 <- plot(rasch(LSAT))
text(2, 0.35, "Rasch model")
p13 <- plot(tpm(LSAT, type = "rasch", max.guessing = 1))
text(2, 0.35, "Rasch model\nwith Guessing parameter")

## Compare the Item Characteristic Curves for Item 4
## (you have to run the above first)
plot(range(p11[, "z"]), c(0, 1), type = "n", xlab = "Ability",
     ylab = "Probability", main = "Item Characteristic Curves - Item 4")
lines(p11[, c("z", "Item 4")], lwd = 2, col = "black")
lines(p12[, c("z", "Item 4")], lwd = 2, col = "red")
lines(p13[, c("z", "Item 4")], lwd = 2, col = "blue")
legend("right", c("Rasch model Discrimination = 1", "Rasch model",
    "Rasch model with\nGuessing parameter"), lwd = 2, col = c("black",
    "red", "blue"), bty = "n")

```

 rasch

Rasch Model

Description

Fit the Rasch model under the Item Response Theory approach.

Usage

```
rasch(data, constraint = NULL, IRT.param = TRUE, start.val = NULL,
      na.action = NULL, control = list())
```

Arguments

<code>data</code>	a <code>data.frame</code> (that will be converted to a numeric matrix using <code>data.matrix()</code>) or a numeric matrix of manifest variables.
<code>constraint</code>	a two-column numeric matrix with at most p rows (where p is the number of items), specifying fixed-value constraints. The first column represents the item (i.e., 1 denotes the first item, 2 the second, etc., and $p + 1$ the discrimination parameter) and the second column the value at which the corresponding parameter should be fixed. See Examples for more info.
<code>IRT.param</code>	logical; if <code>TRUE</code> then the coefficients' estimates are reported under the usual IRT parameterization. See Details for more info.
<code>start.val</code>	the character string "random" or a numeric vector of $p + 1$ starting values, where the first p values correspond to the easiness parameters while the last value corresponds to the discrimination parameter. If "random", random starting values are used. If <code>NULL</code> starting values are automatically computed.
<code>na.action</code>	the <code>na.action</code> to be used on <code>data</code> . In case of missing data, if <code>na.action = NULL</code> the model uses the available cases, i.e., it takes into account the observed part of sample units with missing values (valid under MAR mechanisms if the model is correctly specified). If you want to apply a complete case analysis then use <code>na.action = na.exclude</code> .
<code>control</code>	a list of control values, <ul style="list-style-type: none"> iter.qN the number of quasi-Newton iterations. Default 150. GHk the number of Gauss-Hermite quadrature points. Default 21. method the optimization method to be used in <code>optim()</code>. Default "BFGS". verbose logical; if <code>TRUE</code> info about the optimization procedure are printed.

Details

The Rasch model is a special case of the unidimensional latent trait model when all the discrimination parameters are equal. This model was first discussed by Rasch (1960) and it is mainly used in educational testing where the aim is to study the abilities of a particular set of individuals.

The model is defined as follows

$$\log\left(\frac{\pi_i}{1 - \pi_i}\right) = \beta_i + \beta z,$$

where π_i denotes the conditional probability of responding correctly to the i th item given z , β_i is the easiness parameter for the i th item, β is the discrimination parameter (the same for all the items) and z denotes the latent ability.

If `IRT.param = TRUE`, then the parameters estimates are reported under the usual IRT parameterization, i.e.,

$$\log\left(\frac{\pi_i}{1 - \pi_i}\right) = \beta(z - \beta_i^*).$$

The fit of the model is based on approximate marginal Maximum Likelihood, using the Gauss-Hermite quadrature rule for the approximation of the required integrals.

Value

An object of class `rasch` with components,

<code>coefficients</code>	a matrix with the parameter values at convergence. These are always the estimates of β_i, β parameters, even if <code>IRT.param = TRUE</code> .
<code>log.Lik</code>	the log-likelihood value at convergence.
<code>convergence</code>	the convergence identifier returned by <code>optim()</code> .
<code>hessian</code>	the approximate Hessian matrix at convergence returned by <code>optim()</code> .
<code>counts</code>	the number of function and gradient evaluations used by the quasi-Newton algorithm.
<code>patterns</code>	a list with two components: (i) <code>X</code> : a numeric matrix that contains the observed response patterns, and (ii) <code>obs</code> : a numeric vector that contains the observed frequencies for each observed response pattern.
<code>GH</code>	a list with two components used in the Gauss-Hermite rule: (i) <code>Z</code> : a numeric matrix that contains the abscissas, and (ii) <code>GHw</code> : a numeric vector that contains the corresponding weights.
<code>max.sc</code>	the maximum absolute value of the score vector at convergence.
<code>constraint</code>	the value of the <code>constraint</code> argument.
<code>IRT.param</code>	the value of the <code>IRT.param</code> argument.
<code>X</code>	a copy of the response data matrix.
<code>control</code>	the values used in the <code>control</code> argument.
<code>na.action</code>	the value of the <code>na.action</code> argument.
<code>call</code>	the matched call.

Warning

In case the Hessian matrix at convergence is not positive definite, try to re-fit the model using `rasch(..., start.val = "random")`.

Note

Although the common formulation of the Rasch model assumes that the discrimination parameter is fixed to 1, `rasch()` estimates it. If you wish to fit the constrained version of the model, use the `constraint` argument accordingly. See **Examples** for more info.

The optimization algorithm works under the constraint that the discrimination parameter β is always positive.

When the coefficients' estimates are reported under the usual IRT parameterization (i.e., `IRT.param = TRUE`), their standard errors are calculated using the Delta method.

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

References

- Baker, F. and Kim, S-H. (2004) *Item Response Theory*, 2nd ed. New York: Marcel Dekker.
- Rasch, G. (1960) *Probabilistic Models for Some Intelligence and Attainment Tests*. Copenhagen: Paedagogiske Institute.
- Rizopoulos, D. (2006) **Itm**: An R package for latent variable modelling and item response theory analyses. *Journal of Statistical Software*, **17(5)**, 1–25. URL <http://www.jstatsoft.org/v17/i05/>

See Also

[coef.rasch](#), [fitted.rasch](#), [summary.rasch](#), [anova.rasch](#), [plot.rasch](#), [vcov.rasch](#), [GoF.rasch](#), [item.fit](#), [person.fit](#), [margins](#), [factor.scores](#)

Examples

```
## The common form of the Rasch model for the
## LSAT data, assuming that the discrimination
## parameter equals 1
rasch(LSAT, constraint = cbind(ncol(LSAT) + 1, 1))

## The Rasch model for the LSAT data under the
## normal ogive; to do that fix the discrimination
## parameter to 1.702
rasch(LSAT, constraint = cbind(ncol(LSAT) + 1, 1.702))

## The Rasch model for the LSAT data with
## unconstrained discrimination parameter
rasch(LSAT)

## The Rasch model with (artificially created)
## missing data
data <- LSAT
data[] <- lapply(data, function(x){
  x[sample(1:length(x), sample(15, 1))] <- NA
  x
})
```

```

    })
  rasch(data)

```

rcor.test

Pairwise Associations between Items using a Correlation Coefficient

Description

Computes and tests the pairwise associations between items using a correlation coefficient

Usage

```
rcor.test(mat, p.adjust = FALSE, p.adjust.method = "holm", ...)
```

Arguments

mat	a numeric matrix or a numeric data.frame containing the manifest variables as columns.
p.adjust	logical; if TRUE the p -values are adjusted for multiple comparisons.
p.adjust.method	the method argument of p.adjust().
...	extra arguments passed to cor() and cor.test().

Value

An object of class rcor.test with components,

cor.mat	the correlation matrix.
p.values	a three column numeric matrix containing the p -values for all the combinations of items.

The print method for class rcor.test returns a square matrix in which the upper diagonal part contains the estimates of the correlation coefficients, and the lower diagonal part contains the corresponding p -values.

Note

rcor.test() is more appropriate for informal testing of association between polytomous items.

Author(s)

Dimitris Rizopoulos (d.rizopoulos@erasmusmc.nl)

Examples

```
## pairwise associations for Environment data:
rcor.test(data.matrix(Environment), method = "kendall")

## pairwise associations for independent normal random variates:
mat <- matrix(rnorm(1000), 100, 10, dimnames = list(NULL, LETTERS[1:10]))
rcor.test(mat)
rcor.test(mat, method = "kendall")
rcor.test(mat, method = "spearman")
```

residuals

*Residuals for IRT models***Description**

Computes the residuals for vectors of response patterns.

Usage

```
## S3 method for class 'gpcm':
residuals(object, resp.patterns = NULL, order = TRUE, ...)

## S3 method for class 'grm':
residuals(object, resp.patterns = NULL, order = TRUE, ...)

## S3 method for class 'ltm':
residuals(object, resp.patterns = NULL, order = TRUE, ...)

## S3 method for class 'rasch':
residuals(object, resp.patterns = NULL, order = TRUE, ...)

## S3 method for class 'tpm':
residuals(object, resp.patterns = NULL, order = TRUE, ...)
```

Arguments

object	an object inheriting either from class <code>gpcm</code> , class <code>grm</code> , class <code>ltm</code> , class <code>rasch</code> or class <code>tpm</code> .
resp.patterns	a matrix or a <code>data.frame</code> of response patterns with columns denoting the items; if <code>NULL</code> the expected frequencies are computed for the observed response patterns.
order	logical; if <code>TRUE</code> the response patterns are sorted according to the residual estimates.
...	additional arguments; currently none is used.

Details

The following residuals are computed:

$$\frac{O_i - E_i}{\sqrt{E_i}},$$

where O_i and E_i denote the observed and expected frequencies for the i th response pattern.

Value

A numeric matrix containing the observed and expected frequencies as well as the residual value for each response pattern.

Author(s)

Dimitris Rizopoulos (d.rizopoulos@erasmusmc.nl)

See Also

[fitted.gpcm](#), [fitted.grm](#), [fitted.ltm](#), [fitted.rasch](#), [fitted.tpm](#)

Examples

```
fit <- ltm(LSAT ~ z1)
residuals(fit)
residuals(fit, order = FALSE)
```

rmvlogis

Generate Random Responses Patterns under Dichotomous and Polytomous IRT models

Description

Produces Bernoulli or Multinomial random variates under the Rasch, the two-parameter logistic, the three parameter, the graded response, and the generalized partial credit models.

Usage

```
rmvlogis(n, thetas, IRT = TRUE, link = c("logit", "probit"),
         distr = c("normal", "logistic", "log-normal", "uniform"),
         z.vals = NULL)
```

```
rmvordlogis(n, thetas, IRT = TRUE, model = c("gpcm", "grm"),
            link = c("logit", "probit"),
            distr = c("normal", "logistic", "log-normal", "uniform"),
            z.vals = NULL)
```

Arguments

<code>n</code>	a scalar indicating the number of response patterns to simulate.
<code>thetas</code>	for <code>rmvlogis()</code> a numeric matrix with rows representing the items and columns the parameters. For <code>rmvordlogis()</code> a list with numeric vector elements, with first the threshold parameters and last the discrimination parameter. See Details for more info.
<code>IRT</code>	logical; if <code>TRUE</code> <code>thetas</code> are under the IRT parameterization. See Details for more info.
<code>model</code>	from which model to simulate.
<code>link</code>	a character string indicating the link function to use. Options are <code>logit</code> and <code>probit</code> .
<code>distr</code>	a character string indicating the distribution of the latent variable. Options are <code>Normal</code> , <code>Logistic</code> , <code>log-Normal</code> , and <code>Uniform</code> .
<code>z.vals</code>	a numeric vector of length <code>n</code> providing the values of the latent variable (ability) to be used in the simulation of the dichotomous responses; if specified the value of <code>distr</code> is ignored.

Details

The binary variates can be simulated under the following parameterizations for the probability of correctly responding in the i th item. If `IRT = TRUE`

$$\pi_i = c_i + (1 - c_i)g(\beta_{2i}(z - \beta_{1i})),$$

whereas if `IRT = FALSE`

$$\pi_i = c_i + (1 - c_i)g(\beta_{1i} + \beta_{2i}z),$$

z denotes the latent variable, β_{1i} and β_{2i} are the first and second columns of `thetas`, respectively, and $g()$ is the link function. If `thetas` is a three-column matrix then the third column should contain the guessing parameters c_i 's.

The ordinal variates are simulated according to the generalized partial credit model or the graded response model depending on the value of the `model` argument. Check [gpcm](#) and [grm](#) to see how these models are defined, under both parameterizations.

Value

a numeric matrix with `n` rows and columns the number of items, containing the simulated binary or ordinal variates.

Note

For options `distr = "logistic"`, `distr = "log-normal"` and `distr = "uniform"` the simulated random variates for z simulated under the Logistic distribution with `location = 0` and `scale = 1`, the log-Normal distribution with `meanlog = 0` and `sdlog = 1` and the Uniform distribution with `min = -3.5` and `max = 3.5`, respectively. Then, the simulated z variates are standardized, using the theoretical mean and variance of the Logistic, log-Normal and Uniform distribution, respectively.

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

See Also

[gpcm](#), [grm](#), [ltm](#), [rasch](#), [tpm](#)

Examples

```
# 10 response patterns under a Rasch model
# with 5 items
rmvlogis(10, cbind(seq(-2, 2, 1), 1))

# 10 response patterns under a GPCM model
# with 5 items, with 3 categories each
thetas <- lapply(1:5, function(u) c(seq(-1, 1, len = 2), 1.2))
rmvordlogis(10, thetas)
```

Science

Attitude to Science and Technology

Description

This data set comes from the Consumer Protection and Perceptions of Science and Technology section of the 1992 Euro-Barometer Survey (Karlheinz and Melich, 1992) based on a sample from Great Britain. The questions asked are given below:

Format

All of the below items were measured on a four-group scale with response categories "strongly disagree", "disagree to some extent", "agree to some extent" and "strongly agree":

Comfort Science and technology are making our lives healthier, easier and more comfortable.

Environment Scientific and technological research cannot play an important role in protecting the environment and repairing it.

Work The application of science and new technology will make work more interesting.

Future Thanks to science and technology, there will be more opportunities for the future generations.

Technology New technology does not depend on basic scientific research.

Industry Scientific and technological research do not play an important role in industrial development.

Benefit The benefits of science are greater than any harmful effect it may have.

References

Bartholomew, D., Steel, F., Moustaki, I. and Galbraith, J. (2002) *The Analysis and Interpretation of Multivariate Data for Social Scientists*. London: Chapman and Hall.

Karlheinz, R. and Melich, A. (1992) Euro-Barometer 38.1: *Consumer Protection and Perceptions of Science and Technology*. INRA (Europe), Brussels. [computer file]

Examples

```
## Descriptive statistics for Science data
descript(Science)
```

summary

Summary method for fitted IRT models

Description

Summarizes the fit of either `gpc`, `lrm`, `rasch` or `tpm` objects.

Usage

```
## S3 method for class 'gpc':
summary(object, robust.se = FALSE, ...)
```

```
## S3 method for class 'lrm':
summary(object, ...)
```

```
## S3 method for class 'lrm':
summary(object, robust.se = FALSE, ...)
```

```
## S3 method for class 'rasch':
summary(object, robust.se = FALSE, ...)
```

```
## S3 method for class 'tpm':
summary(object, ...)
```

Arguments

<code>object</code>	an object inheriting from either class <code>gpc</code> , either class <code>lrm</code> , class <code>lrm</code> , class <code>rasch</code> or class <code>tpm</code> .
<code>robust.se</code>	logical; if TRUE robust estimation of standard errors is used, based on the sandwich estimator.
<code>...</code>	additional argument; currently none is used.

Value

An object of either class `summ.gpcm`, class `summ.grm`, class `summ.ltm` or class `summ.rasch` with components,

<code>coefficients</code>	the estimated coefficients' table.
<code>Var.betas</code>	the approximate covariance matrix for the estimated parameters; returned only in <code>summ.ltm</code> and <code>summ.rasch</code> .
<code>logLik</code>	the log-likelihood of object.
<code>AIC</code>	the AIC for object.
<code>BIC</code>	the BIC for object.
<code>max.sc</code>	the maximum absolute value of the score vector at convergence.
<code>conv</code>	the convergence identifier returned by <code>optim()</code> .
<code>counts</code>	the <code>counts</code> argument returned by <code>optim()</code> .
<code>call</code>	the matched call of object.
<code>ltm.struct</code>	a character vector describing the latent structure used in object; returned only in <code>summ.ltm</code> .
<code>control</code>	the values used in the <code>control</code> argument in the fit of object.
<code>nitems</code>	the number of items in the data set; returned only in <code>summ.ltm</code> and <code>summ.rasch</code> .

Note

For the parameters that have been constrained, the standard errors and z -values are printed as NA.

When the coefficients' estimates are reported under the usual IRT parameterization (i.e., `IRT.param = TRUE` in the call of either `grm`, `ltm` or `rasch`), their standard errors are calculated using the Delta method.

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

See Also

[gpcm](#), [grm](#), [ltm](#), [rasch](#), [tpm](#)

Examples

```
# use Hessian = TRUE if you want standard errors
fit <- grm(Science[c(1,3,4,7)], Hessian = TRUE)
summary(fit)

## One factor model using the WIRS data;
## results are reported under the IRT
## parameterization
fit <- ltm(WIRS ~ z1)
summary(fit)
```

testEquatingData *Prepares Data for Test Equating*

Description

Test equating by common items.

Usage

```
testEquatingData(DataList, AnchoringItems = NULL)
```

Arguments

`DataList` a list of `data.frames` or matrices containing common and unique items between several forms.

`AnchoringItems` a `data.frame` or a matrix containing anchoring items for across sample equating.

Details

The purpose of this function is to combine items from different forms. Two cases are considered. Alternate Form Equating (where common and unique items are analyzed simultaneously) and Across Sample Equating (where different sets of unique items are analyzed separately based on previously calibrated anchor items).

Value

A matrix containing the common and unique items.

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

References

Yu, C.-H. and Osborn Popp, S. (2005) Test equating by common items and common subjects: concepts and applications. *Practical Assessment Research and Evaluation*, **10(4)**, 1–19. URL <http://pareonline.net/getvn.asp?v=10&n=4>

Rizopoulos, D. (2006) **Itm**: An R package for latent variable modelling and item response theory analyses. *Journal of Statistical Software*, **17(5)**, 1–25. URL <http://www.jstatsoft.org/v17/i05/>

Examples

```
# Let two data-sets with common and unique items
dat1 <- as.data.frame(rmvlogis(20, cbind(c(-2, 1, 2, 1), 1)))
names(dat1) <- c("CIIt2", "CIIt3", "CIIt4", "W")

dat2 <- as.data.frame(rmvlogis(10, cbind(c(-2, -1, 1, 2, 0.95), 1)))
names(dat2) <- c("CIIt1", "CIIt2", "CIIt3", "CIIt4", "K")

# combine in one data-set by
lisForms <- list(dat1, dat2)
testEquatingData(lisForms)
```

 tpm

Birnbaum's Three Parameter Model

Description

Fit Birnbaum's three parameter model under the Item Response Theory approach.

Usage

```
tpm(data, type = c("latent.trait", "rasch"), constraint = NULL,
     max.guessing = 1, IRT.param = TRUE, start.val = NULL,
     na.action = NULL, control = list())
```

Arguments

<code>data</code>	a <code>data.frame</code> (that will be converted to a numeric matrix using <code>data.matrix()</code>) or a numeric matrix of manifest variables.
<code>type</code>	a character string indicating the type of model to fit. Available options are 'rasch' that assumes equal discrimination parameter among items, and 'latent.trait' (default) that assumes a different discrimination parameter per item.
<code>constraint</code>	a three-column numeric matrix specifying fixed-value constraints. The first column represents the item (i.e., 1 denotes the first item, 2 the second, etc.); the second column denotes the type of parameter to fix for the item specified in the first column (i.e., 1 denotes the guessing parameters, 2 the easiness parameters, and 3 the discrimination parameters); the third column specifies the value at which the corresponding parameter should be fixed. See Examples for more info.
<code>max.guessing</code>	a scalar between 0 and 1 denoting the upper bound for the guessing parameters.
<code>IRT.param</code>	logical; if <code>TRUE</code> then the coefficients' estimates are reported under the usual IRT parameterization. See Details for more info.

<code>start.val</code>	the character string "random" or a numeric matrix supplying starting values with p rows and 3 columns, with p denoting the number of items. If NULL starting values are automatically computed. If "random", random starting values are used. If a matrix, then the first column should contain the guessing parameter, the second β_{1i} , and the third β_{2i} (see Details). If <code>type == "rasch"</code> , then the third should contain the same number p times.
<code>na.action</code>	the <code>na.action</code> to be used on data. In case of missing data, if <code>na.action = NULL</code> the model uses the available cases, i.e., it takes into account the observed part of sample units with missing values (valid under MAR mechanisms if the model is correctly specified). If you want to apply a complete case analysis then use <code>na.action = na.exclude</code> .
<code>control</code>	a list of control values with elements, <ul style="list-style-type: none"> optimizer a character string denoting the optimizer to use, either "optim" (default) or "nlminb". iter.qN scalar denoting the number of iterations in the optimization procedure. For <code>optim()</code> this is passed to the control argument 'maxit', whereas for <code>nlminb()</code> this is passed to both control arguments 'iter.max' and 'eval.max'. Default 1000. GHk scalar denoting the number of Gauss-Hermite quadrature points. Default 21. method a character string denoting the optimization method to be used in <code>optim()</code>. Default "BFGS". verbose logical; if TRUE info about the optimization procedure are printed. eps.hessian the step-length to use in the central difference approximation that approximates the hessian. Default is $1e-03$. parscale a scaling numeric vector of length equal to the parameters to be estimated (taking into account any constraints). This is passed to either to the 'parscale' control argument of <code>optim()</code> or to the 'scale' argument of <code>nlminb()</code>. Default is 0.5 for the guessing parameters and 1 for the discrimination and easiness parameters.

Details

Birnbaum's three parameter model is usually employed to handle the phenomenon of non-random guessing in the case of difficult items.

The model is defined as follows

$$\pi_i = c_i + (1 - c_i) \frac{\exp(\beta_{1i} + \beta_{2i}z)}{1 + \exp(\beta_{1i} + \beta_{2i}z)},$$

where π_i denotes the conditional probability of responding correctly to the i th item given z , c_i denotes the guessing parameter, β_{1i} is the easiness parameter, β_{2i} is the discrimination parameter, and z denotes the latent ability. In case `type = "rasch"`, β_{2i} is assumed equal for all items.

If `IRT.param = TRUE`, then the parameters estimates are reported under the usual IRT parameterization, i.e.,

$$\pi_i = c_i + (1 - c_i) \frac{\exp[\beta_{2i}(z - \beta_{1i}^*)]}{1 + \exp[\beta_{2i}(z - \beta_{1i}^*)]}.$$

The fit of the model is based on approximate marginal Maximum Likelihood, using the Gauss-Hermite quadrature rule for the approximation of the required integrals.

Value

An object of class `tpm` with components,

<code>coefficients</code>	a matrix with the parameter values at convergence. These are always the estimates of β_i, β parameters, even if <code>IRT.param = TRUE</code> .
<code>log.Lik</code>	the log-likelihood value at convergence.
<code>convergence</code>	the convergence identifier returned by <code>optim()</code> .
<code>hessian</code>	the approximate Hessian matrix at convergence obtained using a central difference approximation.
<code>counts</code>	the number of function and gradient evaluations used by the optimization algorithm.
<code>patterns</code>	a list with two components: (i) <code>X</code> : a numeric matrix that contains the observed response patterns, and (ii) <code>obs</code> : a numeric vector that contains the observed frequencies for each observed response pattern.
<code>GH</code>	a list with two components used in the Gauss-Hermite rule: (i) <code>Z</code> : a numeric matrix that contains the abscissas, and (ii) <code>GHw</code> : a numeric vector that contains the corresponding weights.
<code>max.sc</code>	the maximum absolute value of the score vector at convergence.
<code>type</code>	the value of the <code>type</code> argument.
<code>constraint</code>	the value of the <code>constraint</code> argument.
<code>max.guessing</code>	the value of the <code>max.guessing</code> argument.
<code>IRT.param</code>	the value of the <code>IRT.param</code> argument.
<code>X</code>	a copy of the response data matrix.
<code>control</code>	the values used in the <code>control</code> argument.
<code>na.action</code>	the value of the <code>na.action</code> argument.
<code>call</code>	the matched call.

Warning

The three parameter model is known to have numerical problems like non-convergence or convergence on the boundary, especially for the guessing parameters. These problems usually result in a zero estimate for some guessing parameters and/or in a non positive definite Hessian matrix or in a high absolute value for the score vector (returned by the `summary` method) at convergence. In case of estimates on the boundary, the `constraint` argument can be used to set the guessing parameter(s) for the problematic item(s) to zero. In addition, `tpm()` has a number of control parameters that can be tuned in order to obtain successful convergence; the most important of these are the starting values, the parameter scaling vector and the optimizer.

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

References

- Baker, F. and Kim, S-H. (2004) *Item Response Theory*, 2nd ed. New York: Marcel Dekker.
- Birnbaum, A. (1968). Some latent trait models and their use in inferring an examinee's ability. In F. M. Lord and M. R. Novick (Eds.), *Statistical Theories of Mental Test Scores*, 397–479. Reading, MA: Addison-Wesley.
- Rizopoulos, D. (2006) **Itm**: An R package for latent variable modelling and item response theory analyses. *Journal of Statistical Software*, **17(5)**, 1–25. URL <http://www.jstatsoft.org/v17/i05/>

See Also

[coef.tpm](#), [fitted.tpm](#), [summary.tpm](#), [anova.tpm](#), [plot.tpm](#), [vcov.tpm](#), [item.fit](#), [person.fit](#), [margins](#), [factor.scores](#)

Examples

```
# the three parameter model
tpm(LSAT)

# use 'nlminb' as optimizer
tpm(LSAT, control = list(optimizer = "nlminb"))

# the three parameter model with equal
# discrimination parameter across items
# fix the guessing parameter for the third item to zero
tpm(LSAT, type = "rasch", constraint = cbind(3, 1, 0))

# the three parameter model for the Abortion data
fit <- tpm(Abortion)
fit

# the guessing parameter estimates for items 1, 3, and 4 seem to be on
# the boundary; update the fit by fixing them to zero
update(fit, constraint = cbind(c(1, 3, 4), 1, 0))
```

unidimTest

Unidimensionality Check using Modified Parallel Analysis

Description

An empirical check for the unidimensionality assumption for ltm, rasch and tpm models.

Usage

```
unidimTest(object, data, thetas, IRT = TRUE, z.vals = NULL,
           B = 100, ...)
```

Arguments

<code>object</code>	a model object inheriting either from class <code>ltm</code> , class <code>rasch</code> or class <code>tpm</code> . For <code>ltm()</code> it is assumed that the two-parameter logistic model has been fitted (i.e., one latent variable and no nonlinear terms); see Note for an extra option.
<code>data</code>	a matrix or a <code>data.frame</code> of response patterns with columns denoting the items; used if <code>object</code> is missing.
<code>thetas</code>	a numeric matrix with IRT model parameter values to be used in <code>rmvlogis</code> ; used if <code>object</code> is missing.
<code>IRT</code>	logical, if <code>TRUE</code> , then argument <code>thetas</code> contains the measurement model parameters under the usual IRT parameterization (see <code>rmvlogis</code>); used if <code>object</code> is missing.
<code>z.vals</code>	a numeric vector of length equal to the number of rows of <code>data</code> , providing ability estimates. If <code>object</code> is supplied then the abilities are estimated using <code>factor.scores</code> . If <code>NULL</code> , the abilities are simulated from a standard normal distribution.
<code>B</code>	the number of samples for the Monte Carlo procedure to approximate the distribution of the statistic under the null hypothesis.
<code>...</code>	extra arguments to <code>polycor()</code> .

Details

This function implements the procedure proposed by Drasgow and Lissak (1983) for examining the latent dimensionality of dichotomously scored item responses. The statistic used for testing unidimensionality is the second eigenvalue of the tetrachoric correlations matrix of the dichotomous items. The tetrachoric correlations between are computed using function `polycor()` from package ‘`polycor`’, and the largest one is taken as communality estimate.

A Monte Carlo procedure is used to approximate the distribution of this statistic under the null hypothesis. In particular, the following steps are replicated `B` times:

Step 1: If `object` is supplied, then simulate new ability estimates, say z^* , from a normal distribution with mean the ability estimates \hat{z} in the original data-set, and standard deviation the standard error of \hat{z} (in this case the `z.vals` argument is ignored). If `object` is not supplied and the `z.vals` argument has been specified, then set $z^* = z.vals$. Finally, if `object` is not supplied and the `z.vals` argument has not been specified, then simulate z^* from a standard normal distribution.

Step 2: Simulate a new data-set of dichotomous responses, using z^* , and parameters the estimated parameters extracted from `object` (if it is supplied) or the parameters given in the `thetas` argument.

Step 3: For the new data-set simulated in Step 2, compute the tetrachoric correlations matrix and take the largest correlations as communalities. For this matrix compute the eigenvalues.

Denote by T_{obs} the value of the statistic (i.e., the second eigenvalue) for the original data-set. Then the p -value is approximated according to the formula $(1 + \sum_{b=1}^B I(T_b \geq T_{obs})) / (1 + B)$, where $I(\cdot)$ denotes the indicator function, and T_b denotes the value of the statistic in the b th data-set.

Value

An object of class `unidimTest` is a list with components,

<code>Tobs</code>	a numeric vector of the eigenvalues for the observed data-set.
<code>Tboot</code>	a numeric matrix of the eigenvalues for each simulated data-set.
<code>p.value</code>	the p -value.
<code>call</code>	a copy of the matched call of <code>object</code> if that was supplied.

Note

For `ltm` objects you can also use a likelihood ratio test to check unidimensionality. In particular, `fit0 <- ltm(data ~ z1); fit1 <- ltm(data ~ z1 + z2); anova(fit0, fit1)`.

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

References

Drasgow, F. and Lissak, R. (1983) Modified parallel analysis: a procedure for examining the latent dimensionality of dichotomously scored item responses. *Journal of Applied Psychology*, **68**, 363–373.

See Also

[descript](#)

Examples

```
## Not run:
# Unidimensionality Check for the LSAT data-set
# under a Rasch model:
out <- unidimTest(rasch(LSAT))
out
plot(out, type = "b", pch = 1:2)
legend("topright", c("Real Data", "Average Simulated Data"), lty = 1,
      pch = 1:2, col = 1:2, bty = "n")
## End(Not run)
```

vcov

vcov method for fitted IRT models

Description

Extracts the asymptotic variance-covariance matrix of the MLEs from either `gpcm`, `grm`, `ltm`, `rasch` or `tpm` objects.

Usage

```
## S3 method for class 'gpcm':  
vcov(object, robust = FALSE, ...)  
  
## S3 method for class 'grm':  
vcov(object, ...)  
  
## S3 method for class 'ltm':  
vcov(object, robust = FALSE, ...)  
  
## S3 method for class 'rasch':  
vcov(object, robust = FALSE, ...)  
  
## S3 method for class 'tpm':  
vcov(object, ...)
```

Arguments

object	an object inheriting from either class <code>gpcm</code> , class <code>grm</code> , class <code>ltm</code> , class <code>rasch</code> or class <code>tpm</code> .
robust	logical; if TRUE the sandwich estimator is used.
...	additional arguments; currently none is used.

Value

a numeric matrix representing the estimated covariance matrix of the maximum likelihood estimates. Note that this covariance matrix is for the parameter estimates under the additive parameterization and not under the usual IRT parameterization; for more info check the **Details** section of [grm](#), [ltm](#), [rasch](#), and [tpm](#).

Author(s)

Dimitris Rizopoulos <d.rizopoulos@erasmusmc.nl>

See Also

[gpcm](#), [grm](#), [ltm](#), [rasch](#), [tpm](#)

Examples

```
fit <- rasch(WIRS)  
vcov(fit)  
sqrt(diag(vcov(fit))) # standard errors under additive parameterization
```

WIRS

Workplace Industrial Relation Survey Data

Description

These data were taken from a section of the 1990 Workplace Industrial Relation Survey (WIRS) dealing with management/worker consultation in firms. The questions asked are given below:

Format

Please consider the most recent change involving the introduction of the new plant, machinery and equipment. Were discussions or consultations of any of the type on this card held either about the introduction of the change or about the way it was to be implemented.

Item 1 Informal discussion with individual workers.

Item 2 Meeting with groups of workers.

Item 3 Discussions in established joint consultative committee.

Item 4 Discussions in specially constituted committee to consider the change.

Item 5 Discussions with the union representatives at the establishment.

Item 6 Discussions with paid union officials from outside.

Source

The WIRS surveys can be found at <http://www.niesr.ac.uk/niesr/wers98/>.

References

Bartholomew, D. (1998) Scaling unobservable constructs in social science. *Applied Statistics*, **47**, 1–13.

Bartholomew, D., Steel, F., Moustaki, I. and Galbraith, J. (2002) *The Analysis and Interpretation of Multivariate Data for Social Scientists*. London: Chapman and Hall.

Examples

```
## Descriptive statistics for Wirs data
descript(WIRS)
```

Index

*Topic **datasets**

Abortion, 3
Environment, 13
LSAT, 30
Mobility, 36
Science, 55
WIRS, 66

*Topic **methods**

anova, 4
coef, 7
factor.scores, 13
fitted, 17
margins, 34
plot.descript, 41
plot.fscores, 42
plot.IRT, 43
residuals, 52
summary, 56
vcov, 64

*Topic **multivariate**

biserial.cor, 6
cronbach.alpha, 9
descript, 11
GoF, 18
gpcm, 20
grm, 23
information, 26
item.fit, 27
ltm, 30
ltm-package, 1
mult.choice, 37
person.fit, 38
rasch, 48
rcor.test, 51
rmvlogis, 53
testEquatingData, 58
tpm, 59
unidimTest, 62

*Topic **package**

ltm-package, 1

*Topic **regression**

gpcm, 20
grm, 23
ltm, 30
rasch, 48
tpm, 59

Abortion, 3
anova, 4
anova.gpcm, 23
anova.grm, 26
anova.ltm, 34
anova.rasch, 50
anova.tpm, 62

biserial.cor, 6

coef, 7
coef.gpcm, 23
coef.grm, 26
coef.ltm, 34
coef.rasch, 50
coef.tpm, 62
cronbach.alpha, 9

descript, 11, 41, 64

Environment, 11, 13
expression, 41, 42, 45

factor.scores, 13, 23, 26, 34, 39, 43, 50,
62, 63

fitted, 17
fitted.gpcm, 23, 53
fitted.grm, 26, 53
fitted.ltm, 34, 53
fitted.rasch, 50, 53
fitted.tpm, 53, 62

GoF, 18

GoF.gpcm, 5, 23, 29, 40
 GoF.rasch, 5, 29, 36, 40, 50
 gpcm, 5, 9, 16, 20, 20, 45, 54, 55, 57, 65
 grm, 5, 9, 16, 23, 45, 54, 55, 57, 65

 information, 26, 45
 item.fit, 20, 27, 34, 36, 40, 50, 62

 legend, 44
 LSAT, 30
 ltm, 5, 9, 16, 22, 30, 45, 55, 57, 65
 ltm-package, 1

 margins, 20, 23, 26, 29, 34, 34, 40, 50, 62
 Mobility, 36
 mult.choice, 37

 par, 44, 45
 person.fit, 20, 29, 34, 36, 38, 50, 62
 plot.descript, 41
 plot.fscores, 42
 plot.IRT, 43
 plot.descript, 12
 plot.descript(plot.descript), 41
 plot.fscores, 16
 plot.fscores(plot.fscores), 42
 plot.gpcm, 23, 27
 plot.gpcm(plot.IRT), 43
 plot.grm, 26, 27
 plot.grm(plot.IRT), 43
 plot.ltm, 27, 34
 plot.ltm(plot.IRT), 43
 plot.rasch, 27, 50
 plot.rasch(plot.IRT), 43
 plot.tpm, 62
 plot.tpm(plot.IRT), 43

 rasch, 5, 9, 16, 20, 22, 45, 48, 55, 57, 65
 rcor.test, 51
 residuals, 52
 residuals.gpcm, 18
 residuals.grm, 18
 residuals.ltm, 18
 residuals.rasch, 18
 residuals.tpm, 18
 rmvlogis, 53, 63
 rmvordlogis(rmvlogis), 53

 Science, 55
 summary, 56

 summary.gpcm, 23
 summary.grm, 26
 summary.ltm, 34
 summary.rasch, 50
 summary.tpm, 62

 testEquatingData, 58
 title, 41, 42, 45
 tpm, 5, 9, 16, 45, 55, 57, 59, 65

 unidimTest, 12, 62

 vcov, 64
 vcov.gpcm, 23
 vcov.grm, 26
 vcov.ltm, 34
 vcov.rasch, 50
 vcov.tpm, 62

 WIRS, 66