Package ‘SNSequate’

February 23, 2020

Version 1.3.3

Date 2020-02-23

Title Standard and Nonstandard Statistical Models and Methods for Test Equating

Maintainer Jorge Gonzalez <jorge.gonzalez@mat.uc.cl>

Depends R (>= 3.1.0), magic, stats

Imports methods, emdbook, plyr, statmod, knitr, progress

Description Contains functions to perform various models and methods for test equating. It currently implements the traditional mean, linear and equipercentile equating methods. Both IRT observed-score and true-score equating are also supported, as well as the mean-mean, mean-sigma, Haebara and Stocking-Lord IRT linking methods. It also supports newest methods such that local equating, kernel equating (using Gaussian, logistic, Epanechnikov, uniform and adaptive kernels) with presmoothing, and IRT parameter linking methods based on asymmetric item characteristic functions. Functions to obtain both standard error of equating (SEE) and standard error of equating differences between two equating functions (SEED) are also implemented for the kernel method of equating.

License GPL (>= 2)

URL http://www.mat.uc.cl/~jorge.gonzalez

Suggests testthat

RoxygenNote 7.0.2

NeedsCompilation no

Author Jorge Gonzalez [cre, aut],
Daniel Leon Acuna [ctb]

Repository CRAN

Date/Publication 2020-02-23 14:50:02 UTC

R topics documented:

`SNSequate-package` *Standard and Nonstandard Statistical Models and Methods for Test Equating*

Description

The package contains functions to perform various models and methods for test equating. It currently implements the traditional mean, linear and equipercentile equating methods. Both IRT observed-score and true-score equating are also supported, as well as the mean-mean, mean-sigma, Haebara and Stocking-Lord IRT linking methods. It also supports newest methods such that local equating, kernel equating (using Gaussian, logistic, Epanechnikov, uniform and adaptive kernels) with presmoothing, and IRT parameter linking methods based on asymmetric item characteristic functions. Functions to obtain both standard error of equating (SEE) and standard error of equating differences between two equating functions (SEED) are also implemented for the kernel method of equating.

Details

Package:	SNSequate
Type:	Package
Version:	1.3.3
Date:	2020-02-23
License:	GPL (>= 2)

Author(s)

Jorge Gonzalez

Maintainer: Jorge Gonzalez <jorge.gonzalez@mat.uc.cl>

References

Estay, G. (2012). *Characteristic Curves Scale Transformation Methods Using Asymmetric ICCs for IRT Equating*. Unpublished MSc. Thesis. Pontificia Universidad Catolica de Chile.

Gonzalez, J. (2013). Statistical Models and Inference for the True Equating Transformation in the Context of Local Equating. *Journal of Educational Measurement, 50(3),* 315-320.

Gonzalez, J. (2014). SNSequate: Standard and Nonstandard Statistical Models and Methods for Test Equating. *Journal of Statistical Software, 59(7),* 1-30.

Gonzalez, J. and Wiberg, M. (2017). Applying test equating methods using R. Springer.

Holland, P. and Thayer, D. (1989). The kernel method of equating score distributions. (Technical Report No 89-84). Princeton, NJ: Educational Testing Service.

Holland, P., King, B. and Thayer, D. (1989). The standard error of equating for the kernel method of equating score distributions (Tech. Rep. No. 89-83). Princeton, NJ: Educational Testing Service.

Kolen, M., and Brennan, R. (2004). *Test Equating, Scaling and Linking*. New York, NY: Springer-Verlag.

Lord, F. (1980). *Applications of Item Response Theory to Practical Testing Problems*. Lawrence Erlbaum Associates, Hillsdale, NJ.

Lord, F. and Wingersky, M. (1984). Comparison of IRT True-Score and Equipercentile Observed-Score Equatings. *Applied Psychological Measurement,8(4),* 453–461.

van der Linden, W. (2011). Local Observed-Score Equating. In A. von Davier (Ed.) *Statistical Models for Test Equating, Scaling, and Linking*. New York, NY: Springer-Verlag.

van der Linden, W. (2013). Some Conceptual Issues in Observed-Score Equating. *Journal of Educational Measurement, 50(3),* 249-285.

Von Davier, A., Holland, P., and Thayer, D. (2004). *The Kernel Method of Test Equating*. New York, NY: Springer-Verlag.

ACTmKB *Scores on two 40-items ACT mathematics test forms*

Description

The data set contains raw sample frequencies of number-right scores for two multiple choice 40-items mathematics tests forms. Form X was administered to 4329 examinees and form Y to 4152 examinees. This data has been described and analized by Kolen and Brennan (2004).

Usage

```
data(ACTmKB)
```

Format

A 41x2 matrix containing raw sample frequencies (raws) for two tests (columns).

Source

The data come with the distribution of the RAGE-RGEQUATE software which is freely available at https://education.uiowa.edu/centers/center-advanced-studies-measurement-and-assessment/computer-programs

References

Kolen, M., and Brennan, R. (2004). *Test Equating, Scaling and Linking*. New York, NY: Springer-Verlag.

Examples

```
data(ACTmKB)
## maybe str(ACTmKB) ; plot(ACTmKB) ...
```

bandwidth *Automatic selection of the bandwidth parameter* h

Description

This functions implements the minimization of the combined penalty function described by Holland and Thayer (1989); Von Davier et al, (2004). It returns the optimal value of h for kernel continuization, according to the above mentioned criteria. Different types of kernels (others than the gaussian) are accepted.

Usage

```
bandwidth(scores, kert, degree, design, Kp = 1, scores2, degreeXA, degreeYA,
J, K, L, wx, wy, w)
```

Arguments

Note that depending on the specified equating design, not all arguments are necessary as detailed below.

`scores` If the "EG" design is specified, a vector containing the raw sample frequencies coming from one group taking the test.

If the "SG" design is specified, a matrix containing the (joint) bivariate sample frequencies for X (raws) and Y (columns).

If the "CB" design is specified, a two column matrix containing the observed scores of the sample taking test X first, followed by test Y. The `scores2` argument is then used for the scores of the sample taking test Y first followed by test X.

If either the "NEAT_CB" or "NEAT_PSE" design is selected, a two column matrix containing the observed scores on test X (first column) and the observed scores on the anchor test A (second column). The `scores2` argument is then used for the observed scores on test Y.

`kert` A character string giving the type of kernel to be used for continuization. Current options include "`gauss`", "`logis`", and "`uniform`" for the gaussian, logistic and uniform kernels, respectively

`degree` Either a number or vector indicating the number of power moments to be fitted to the marginal distributions, or the number or cross moments to be fitted to the joint distributions, respectively. For the "EG" design it will be a number (see Details).

`design` A character string indicating the equating design (one of "EG", "SG", "CB", "NEAT_CE", "NEAT_PSE")

`Kp` A number which acts as a weight for the second term in the combined penalization function used to obtain `h` (see details).

`scores2` Only used for the "CB", "NEAT_CE" and "NEAT_PSE" designs. See the description of `scores`.

`degreeXA` A vector indicating the number of power moments to be fitted to the marginal distributions X and A, and the number or cross moments to be fitted to the joint distribution (X, A) (see details). Only used for the "NEAT_CE" and "NEAT_PSE" designs.

`degreeYA` Only used for the "NEAT_CE" and "NEAT_PSE" designs (see the description for `degreeXA`)

`J` The number of possible X scores. Only needed for "CB", "NEAT_CB" and "NEAT_PSE" designs

`K` The number of possible Y scores. Only needed for "CB", "NEAT_CB" and "NEAT_PSE" designs

`L` The number of possible A scores. Needed for "NEAT_CB" and "NEAT_PSE" designs

`wx` A number that satisfies $0 \leq w_X \leq 1$ indicating the weight put on the data that is not subject to order effects. Only used for the "CB" design.

`wy` A number that satisfies $0 \leq w_Y \leq 1$ indicating the weight put on the data that is not subject to order effects. Only used for the "CB" design.

w — A number that satisfies $0 \leq w \leq 1$ indicating the weight given to population P. Only used for the "NEAT" design.

Details

To automatically select h, the function minimizes

$$PEN_1(h) + K \times PEN_2(h)$$

where $PEN_1(h) = \sum_j (\hat{r}_j - \hat{f}_h(x_j))^2$, and $PEN_2(h) = \sum_j A_j(1 - B_j)$. The terms A and B are such that PEN_2 acts as a smoothness penalty term that avoids rapid fluctuations in the approximated density (see Chapter 10 in Von Davier, 2011 for more details). The K term corresponds to the Kp argument of the bandwidth function. The $\hat{r}$ values are assumed to be estimated by polynomial loglinear models of specific degree, which come from a call to loglin.smooth.

Value

A number which is the optimal value of h.

Author(s)

Jorge Gonzalez <jorge.gonzalez@mat.uc.cl>

References

Gonzalez, J. (2014). SNSequate: Standard and Nonstandard Statistical Models and Methods for Test Equating. *Journal of Statistical Software, 59(7),* 1-30.

Von Davier, A., Holland, P., and Thayer, D. (2004). *The Kernel Method of Test Equating*. New York, NY: Springer-Verlag.

A. von Davier (Ed.) (2011). *Statistical Models for Equating, Scaling, and Linking. New York: Springer*

See Also

loglin.smooth

Examples

```
#Example: The "Standard" column and firsts two rows of Table 10.1 in
#Chapter 10 of Von Davier 2011

data(Math20EG)

hx.logis<-bandwidth(scores=Math20EG[,1],kert="logis",degree=2,design="EG")$h
hx.unif<-bandwidth(scores=Math20EG[,1],kert="unif",degree=2,design="EG")$h
hx.gauss<-bandwidth(scores=Math20EG[,1],kert="gauss",degree=2,design="EG")$h

hy.logis<-bandwidth(scores=Math20EG[,2],kert="logis",degree=3,design="EG")$h
hy.unif<-bandwidth(scores=Math20EG[,2],kert="unif",degree=3,design="EG")$h
hy.gauss<-bandwidth(scores=Math20EG[,2],kert="gauss",degree=3,design="EG")$h
```

```
partialTable10.1<-rbind(c(hx.logis,hx.unif,hx.gauss),
c(hy.logis,hy.unif,hy.gauss))

dimnames(partialTable10.1)<-list(c("h.x","h.y"),c("Logistic","Uniform","Gaussian"))
partialTable10.1
```

BNP.eq *Bayesian non-parametric model for test equating*

Description

This function implements the Bayesian nonparametric approach for test equating as described in Gonzalez, Barrientos and Quintana (2015) <doi: 10.1016/j.csda.2015.03.012>. The main idea consists of introducing covariate dependent Bayesian nonparametric models for a collection of covariate-dependent equating transformations

$\{\varphi_{z_f,z_t}(\cdot) : z_f, z_t \in \mathcal{L}\}$

Usage

```
BNP.eq(scores_x, scores_y, range_scores = NULL, design = "EG",
  covariates = NULL, prior = NULL, mcmc = NULL, normalize = TRUE)
```

Arguments

scores_x	Vector. Scores of form X.
scores_y	Vector. Scores of form Y.
range_scores	Vector of length 2. Represent the minimum and maximum scores in the test.
design	Character. Only supports 'EG' design now.
covariates	Data.frame. A data frame with factors, containing covariates for test X and Y, stacked in that order.
prior	List. Prior information for BNP model. For more information see DPpackage.
mcmc	List. MCMC information for BNP model. For more information see DPpackage.
normalize	Logical. Whether normalize or not the response variable. This is due to Berstein's polynomials. Default is TRUE.

Details

The Bayesian nonparametric (BNP) approach starts by focusing on spaces of distribution functions, so that uncertainty is expressed on F itself. The prior distribution p(F) is defined on the space F of all distribution functions defined on X . If X is an infinite set then F is infinite-dimensional, and the corresponding prior model p(F) on F is termed nonparametric. The prior probability model is also referred to as a random probability measure (RPM), and it essentially corresponds to a distribution on the space of all distributions on the set X . Thus Bayesian nonparametric models are probability models defined on a function space.

Value

A 'BNP.eq' object, which is list containing the following items:

Y Response variable.

X Design Matrix.

fit DPpackage object. Fitted model with raw samples.

max_score Maximum score of test.

patterns A matrix describing the different patterns formed from the factors in the covariables.

patterns_freq The normalized frequency of each pattern.

Author(s)

Daniel Leon <dnacuna@uc.cl>, Felipe Barrientos <afb26@stat.duke.edu>.

References

Gonzalez, J., Barrientos, A., and Quintana, F. (2015). Bayesian Nonparametric Estimation of Test Equating Functions with Covariates. *Computational Statistics and Data Analysis, 89,* 222-244.

BNP.eq.predict — *Prediction step for Bayesian non-parametric model for test equating*

Description

This function implements the prediction step in the Bayesian non-parametric model for test equating

Usage

```
BNP.eq.predict(model, from = NULL, into = NULL, alpha = 0.05)
```

Arguments

`model`	A 'BNP.eq' object.
`from`	Numeric. A vector of indices indicating from which patterns equating should be performed. The covariates involved are integrated out.
`into`	Numeric. A vector of indices indicating into which patterns equating should be performed. The covariates involved are integrated out.
`alpha`	Numeric. Level of significance for credible bands.

Details

Predictions of the score probability distributions are obtained under the Bayesian nonparametric model and are used to compute the equating function.

Value

A 'BNP.eq.predict' object, which is a list containing the following items:

pdf A list of PDF's.

cdf A list of CDF's.

equ Numeric. Equated values.

grid Numeric. Grid used to evaluate pdf's and cdf's.

Author(s)

Daniel Leon <dnacuna@uc.cl>, Felipe Barrientos <afb26@stat.duke.edu>.

References

Gonzalez, J., Barrientos, A., and Quintana, F. (2015). Bayesian Nonparametric Estimation of Test Equating Functions with Covariates. *Computational Statistics and Data Analysis, 89,* 222-244.

CBdata *Observed (raw) score values for two different tests*

Description

The data set is from a small field study from an international testing program. It contains the observed scores for two tests X (with 75 items) and Y (with 76 items) administered to two independent, random samples of examinees from a single population P. For more details, see Chapter 9 in Von Davier et al, (2004) from where the data were obtained.

Usage

```
data(CBdata)
```

Format

A list with elements containing the observed scores of the sample taking test X first, followed by test Y (datX1Y2), and the scores of the sample taking test Y first followed by test X (datX2Y1).

References

Von Davier, A., Holland, P., and Thayer, D. (2004). *The Kernel Method of Test Equating*. New York, NY: Springer-Verlag.

Examples

```
data(CBdata)
## maybe str(CBdata) ; ...
```

`eqp.eq` *The equipercentile method of equating*

Description

This function implements the equipercentile method of test equating as described in Kolen and Brennan (2004).

Usage

```
eqp.eq(sx, sy, X, Ky = max(sy))
```

Arguments

`sx`	A vector containing the observed scores on test X
`sy`	A vector containing the observed scores on test Y
`X`	Either an integer or vector containing the values on the scale to be equated.
`Ky`	The total number of items in test form Y to which form X scores will be equated

Details

The function implements the equipercentile method of equating as described in Kolen and Brennan (2004). Given observed scores `sx` and `sy`, the functions calculates

$$\varphi(x) = G^{-1}(F(x))$$

where F and G are the cdf of scores on test forms X and Y, respectively.

Value

A two column matrix with the values of $\varphi()$ (second column) for each scale value `x` (first column)

Author(s)

Jorge Gonzalez <jorge.gonzalez@mat.uc.cl>

References

Gonzalez, J. (2014). SNSequate: Standard and Nonstandard Statistical Models and Methods for Test Equating. *Journal of Statistical Software, 59(7),* 1-30.

Kolen, M., and Brennan, R. (2004). *Test Equating, Scaling and Linking*. New York, NY: Springer-Verlag.

See Also

`mea.eq`, `lin.eq`, `ker.eq`

Examples

```
### Example from Kolen and Brennan (2004), pages 41-42:
### (scores distributions have been transformed to vectors of scores)

sx<-c(0,0,1,1,1,2,2,3,3,4)
sy<-c(0,1,1,2,2,3,3,3,4,4)
x<-2
eqp.eq(sx,sy,2)

# Whole scale range (Table 2.3 in KB)
eqp.eq(sx,sy,0:4)
```

---

gof *Functions to assess model fitting.*

Description

This function contains various measures to assess the model's goodness of fit.

Usage

```
gof(obs, fit, methods=c("FT"), p.out=FALSE)
```

Arguments

`obs`	A vector containing the observed values.
`fit`	A vector containing the fitted values.
`methods`	A character vector containing one or many of the following methods: **"FT"** Freeman-Tukey Residuals. This is the default test. **"Chisq"** Pearson's Chi-squared test. **"KL"** Symmetrised Kullback-Leibler divergence.
`p.out`	Boolean. Decides whether or not to display plots (on corresponding methods).

Author(s)

Daniel Leon Acuna. `<dnacuna@uc.cl>`

References

Gonzalez, J. (2014). SNSequate: Standard and Nonstandard Statistical Models and Methods for Test Equating. *Journal of Statistical Software, 59(7),* 1-30.

Kolen, M., and Brennan, R. (2004). *Test Equating, Scaling and Linking*. New York, NY: Springer-Verlag.

Johnson, D. H., and Sinanovic, S. (2000). Symmetrizing the Kullback-Leibler distance (Technical report). *IEEE Transactions on Information Theory*.

Examples

```
data(Math20EG)
mod <- ker.eq(scores=Math20EG,kert="gauss",degree=c(2,3),design="EG")

gof(Math20EG[,1], mod$rj*mod$nx, method=c("FT", "KL"))
```

`irt.eq` *IRT methods for Test Equating*

Description

Implements methods to perform Test Equating over IRT models.

Usage

```
irt.eq(n_items, param_x, param_y, theta_points=NULL, weights=NULL, n_points=10, w=1,
    A=NULL, B=NULL, link=NULL, method_link=NULL, common=NULL,  method="TS", D=1.7)
```

Arguments

`n_items`	Number of items of the test
`param_x`	Estimated parameters for IRT model on test X. This list must have the following structure: list(a, b, c), where each parameter is a vector with the respective estimate for each subject. If you want to perform other models (i.e. Rasch), replace according with a vector of zeros.
`param_y`	Estimated parameters for IRT model on test Y. This list must have the following structure: list(a, b, c), where each parameter is a vector with the respective estimate for each subject. If you want to perform other models (i.e. Rasch), replace according with a vector of zeros.
`method`	A string, either "TS" or "OS". Each one stands for "True Score Equating" and "Observed score equating". Notice that OS requires the additional arguments "theta_points" and "weigths".
`theta_points`	For "OS" only. Points over a grid of possible values of θ to integrate out the ability term.
`weights`	For "OS" only. Weigths for integrate out the ability term. If is NULL, the method assumes the distribution of ability is characterized by a finite number of abilities (Kolen and Brennan 2013, pg 199).
`n_points`	In case theta_ponints is not provided, is the length of the grid for the gaussian quadrature.
`A, B`	Scaling parameters. In the case they are not provided, they will be calculated depending on the next described inputs.
`link`	An irt.link object.
`method_link`	Method used to estimate A and B. Default is "mean/sigma". Others are "mean/mean", "Haebara" and "Stocklord". For more information see irt.link

common	Common items to estimate A and B. Default asume all items are common.
w	Weight of the synthetic population.
D	Sclaing constant

Details

This function implements two methods to perform Test Equating over Item Response Theory models (Kolen and Brennan 2013).

"True Score Equating" relate number-correct scores on Form X and Form Y. Assumes that the true score associated with each θ is equivalent to the true score on another form associated with that θ.

"Observed Score Equating" uses the IRT model to produce an estimated distribution of observed number-correct scores on each form. Using the compound binomial distribution (Lord and Wingersky 1984) to find the conditional distributions $f(x \mid \theta)$, and then integrate out the θ parameter. Afterwards, an Equipercentile Equating process is done over the estimated distributions.

Value

An object of the clas `irt.eq` is returned. Depending on the method used, the outputs are:

True Score Equating A list(n_items, theta_equivalent, tau_y) containing the number of items, the theta equivalent values on Form X to Form Y and the equivalent scores.

Observed Score Equating A list(n_items, f_hat, g_hat, e_Y_x) containing the number of items, the estimated distributions and the equated values.

Author(s)

Daniel Acuna Leon. <dnacuna@uc.cl>

References

Kolen, M. J., and Brennan, R. L. (2014). *Test Equating, Scaling, and Linking: Methods and Practices, Third Edition*. Springer Science & Business Media.

See Also

irt.link

Examples

```
data(KB36_t)
dfo <- KB36_t

param_x <- list(a=dfo[,3],b=dfo[,4],c=dfo[,5])
param_y <- list(a=dfo[,7],b=dfo[,8],c=dfo[,9])

theta_points=c(-5.2086,-4.163,-3.1175,-2.072,-1.0269,0.0184,
               1.0635,2.109,3.1546,4.2001)
weights=c(0.000101,0.00276,0.03021,0.142,0.3149,0.3158,
         0.1542,0.03596,0.003925,0.000186)
```

```
irt.eq(36, param_x, param_y, method="TS", A=1, B=0)
irt.eq(36, param_x, param_y, theta_points, weights, method="OS", A=1, B=0)
```

`irt.link` *IRT parameter linking methods*

Description

The function implements parameter linking methods to transform IRT scales. Mean-mean, mean-sigma, Haebara, and Stocking and Lord methods are available (see details).

Usage

```
irt.link(parm, common, model, icc, D)
```

Arguments

`parm`	A 6 column matrix containing item parameter estimates from an IRT model. The first three columns contains the parameters for the form `Y` fit, and the last three those of form `X`. The order for item paramters in the matrix is discrimination, difficulty, and guessing. See details.
`common`	A vector indicating the position where common items are located
`model`	A character string indicating the underlying IRT model: "1PL", "2PL", "3PL".
`icc`	A character string indicating the type of `icc` used in the characteristic curve methods (see details). Available options are "logistic" and "cloglog".
`D`	A number indicating the value of the constant `D` (see details)

Details

The function implments various methods of IRT parameter linking (a.k.a, scale transformation methods). It calculates the linking constants `A` and `B` to tranform parameter estimates. When assuming a 1PL model, the matrix `parm` should contain a column of ones and a column of zeroes in the places where discrimination and guessing parameters are located, respectively.

The characteristic curve methods (Haebara and Stocking and Lord) rely on the item characteristic curve p_{ij}assumed for the probability of a correct answer

$$p_{ij} = P(Y_{ij} = 1 \mid \theta_i) = c_j + (1 - c_j)\frac{\exp[Da_j(\theta_i - \beta_j)]}{1 + \exp[Da_j(\theta_i - \beta_j)]}$$

Besides the traditional logistic model, the `irt.link()` function allows the use of an asymetric cloglog ICC. See the help for `KB36.1PL` data set, where some details on how to fit a 1PL model with cloglog link in `lmer` are given.

For more details on characteristic curve methods see Kolen and Brennan (2004).

Value

A list with the constants `A` and `B` calculated using the four different methods

Note

Currently, the cloglog ICC is only implmented for the 1PL model. A 1PL model with asymetric cloglog link can be fitted in R using the `lmer()` function in package `lme4`

Author(s)

Jorge Gonzalez <jorge.gonzalez@mat.uc.cl>

References

Gonzalez, J. (2014). SNSequate: Standard and Nonstandard Statistical Models and Methods for Test Equating. *Journal of Statistical Software, 59(7),* 1-30.

Kolen, M., and Brennan, R. (2004). *Test Equating, Scaling and Linking*. New York, NY: Springer-Verlag.

Estay, G. (2012). *Characteristic Curves Scale Transformation Methods Using Asymmetric ICCs for IRT Equating*. Unpublished MSc. Thesis. Pontificia Universidad Catolica de Chile

See Also

`mea.eq`, `lin.eq`, `ker.eq`

Examples

```
#### Example. KB, Table 6.6
data(KB36)
parm.x = KB36$KBformX_par
parm.y = KB36$KBformY_par
comitems = seq(3,36,3)
parm = as.data.frame(cbind(parm.y, parm.x))

# Table 6.6 KB
irt.link(parm,comitems,model="3PL",icc="logistic",D=1.7)


# Same data but assuming a 1PL model. The parameter estimates are load from
# the KB36.1PL data set. See the help for KB36.1PL data for details on how these
# estimates were obtained using \code{lmer()} (see also Table 6.13 in KB)

data(KB36.1PL)

#preparing the input data matrices for irt.link() function
b.log.y<-KB36.1PL$b.logistic[,2]
b.log.x<-KB36.1PL$b.logistic[,1]
b.clog.y<-KB36.1PL$b.cloglog[,2]
b.clog.x<-KB36.1PL$b.cloglog[,1]
```

```
parm2 = as.data.frame(cbind(1,b.log.y,0, 1,b.log.x, 0))
parm3 = as.data.frame(cbind(1,b.clog.y,0, 1,b.clog.x,0))

#vector indicating common items
comitems = seq(3,36,3)

#Calculating the B constant under the logistic-link model
irt.link(parm2,comitems,model="1PL",icc="logistic",D=1.7)

#Calculating the B constant under the cloglog-link model
irt.link(parm3,comitems,model="1PL",icc="cloglog",D=1.7)
```

KB36 *Data on two 36-items test forms*

Description

The data set contains both response patterns and item parameters estimates following a 3PL model for two 36-items tests forms. Form X was administered to 1655 examinees and form Y to 1638 examinees. Also, 12 out of the 36 items are common between both test forms (items 3, 6, 9, 12, 15, 18, 21, 24, 27, 30, 33, 36). This data has been described and analized by Kolen and Brennan (2004).

Usage

```
data(KB36)
```

Format

A list with four elements containing binary data matrices of responses (KBformX and KBformY) and the corresponding parameter estimates which result from a 3PL fit to both data matrices (KBformX_par and KBformY_par).

Source

The data come with the distribution of the CIPE software which is freely available at https://education.uiowa.edu/centers/center-advanced-studies-measurement-and-assessment/computer-programs. The list of item parameters estimates can be found in Table 6.5 of Kolen and Brennan (2004).

References

Kolen, M., and Brennan, R. (2004). *Test Equating, Scaling and Linking*. New York, NY: Springer-Verlag.

Examples

```
data(KB36)
## maybe str(KB36) ; plot(KB36) ...
```

KB36.1PL *Difficulty parameter estimates for KB36 data under a 1PL model*

Description

This data set contains the estimated item difficuty parameters for the KB36 data, assuming a 1PL model. Two sets of parameters estimates for test forms X and Y are available: one that results from a fit assuming the traditional logistic link, and one which comes from the fit using a cloglog (asymmetric) link.

Usage

```
data(KB36.1PL)
```

Format

A list of 2 elements containing item (difficulty) parameters estimates for test forms X and Y under the logistic-link model (`b.logistic`), and under the cloglog-link model (`b.cloglog`)

Details

This data set is used to illustrate the characteristic curve methods (Haebara and Stocking-Lord) which can use an asymmetric cloglog ICC for the calculations, as described in Estay (2012).

A 1PL model using both logistic and cloglog link can be fitted using the `lmer()` function in the `lme4` R package (see De Boeck et. al, 2011 for details).

Source

The item parameter estimates for the 1PL model with logistic link are also shown in Table 6.13 of Kolen and Brennan (2004).

References

De Boeck, P., Bakker, M., Zwitser, R., Nivard, M., Hofman, A.,Tuerlinckx, F., Partchev, I. (2011). The Estimation of Item Response Models with the `lmer` Function from the **lme4** Package in R. *Journal of Statistical Software, 39(12),* 1-28.

Kolen, M., and Brennan, R. (2004). *Test Equating, Scaling and Linking*. New York, NY: Springer-Verlag.

Estay, G. (2012). *Characteristic Curves Scale Transformation Methods Using Asymmetric ICCs for IRT Equating*. Unpublished MSc. Thesis. Pontificia Universidad Catolica de Chile

Examples

```
data(KB36.1PL)
## maybe str(KB36.1PL) ; plot(KB36.1PL) ...
```

KB36_t *Data on two 36-items test forms*

Description

The data set contains item parameters estimates following a 3PL model for two 36-items tests forms, rescaled using mean-sigma method's A and B using all common items except item 27. This data has been described and analized by Kolen and Brennan (2004), Table 6.8.

Usage

```
data(KB36_t)
```

Format

A dataframe where each column represent item parameter estimates of forms `X` and `Y`, with their respective p-values.

References

Kolen, M., and Brennan, R. (2004). *Test Equating, Scaling and Linking*. New York, NY: Springer-Verlag.

See Also

KB36

Examples

```
data(KB36_t)
```

ker.eq *The Kernel method of test equating*

Description

This function implements the kernel method of test equating as described in Holland and Thayer (1989), and Von Davier et al. (2004). Nonstandard kernels others than the gaussian are available. Associated standard error of equating are also provided.

Usage

```
ker.eq(scores, kert, hx = NULL, hy = NULL, degree, design, Kp = 1, scores2,
degreeXA, degreeYA, J, K, L, wx, wy, w, gapsX, gapsY, gapsA, lumpX, lumpY,
lumpA, alpha, h.adap)
```

Arguments

Note that depending on the specified equating design, not all arguments are necessary as detailed below.

`scores`
: If the "EG" design is specified, a two column matrix containing the raw sample frequencies coming from the two groups of scores to be equated. It is assumed that the data in the first and second columns come from tests X and Y, respectively.

  If the "SG" design is specified, a matrix containing the (joint) bivariate sample frequencies for X (raws) and Y (columns).

  If the "CB" design is specified, a two column matrix containing the observed scores of the sample taking test X first, followed by test Y. The `scores2` argument is then used for the scores of the sample taking test Y first followed by test X.

  If either the "NEAT_CB" or "NEAT_PSE" design is selected, a two column matrix containing the observed scores on test X (first column) and the observed scores on the anchor test A (second column). The `scores2` argument is then used for the observed scores on test Y.

`kert`
: A character string giving the type of kernel to be used for continuization. Current options include `"gauss"`, `"logis"`, `"uniform"`, `"epan"` and `"adap"` for the gaussian, logistic, uniform, Epanechnikov and Adaptative kernels, respectively

`hx`
: An integer indicating the value of the bandwidth parameter to be used for kernel continuization of $F(x)$. If not provided (Default), this value is automatically calculated (see details).

`hy`
: An integer indicating the value of the bandwidth parameter to be used for kernel continuization of $G(y)$. If not provided (Default), this value is automatically calculated (see details).

`degree`
: A vector indicating the number of power moments to be fitted to the marginal distributions ("EG" design), and/or the number or cross moments to be fitted to the joint distributions (see Details).

`design`
: A character string indicating the equating design (one of "EG", "SG", "CB", "NEAT_CE", "NEAT_PSE")

`Kp`
: A number which acts as a weight for the second term in the combined penalization function used to obtain `h` (see details).

`scores2`
: Only used for the "CB", "NEAT_CE" and "NEAT_PSE" designs. See the description of `scores`.

`degreeXA`
: A vector indicating the number of power moments to be fitted to the marginal distributions X and A, and the number or cross moments to be fitted to the joint distribution (X, A) (see details). Only used for the "NEAT_CE" and "NEAT_PSE" designs.

`degreeYA`
: Only used for the "NEAT_CE" and "NEAT_PSE" designs (see the description for `degreeXA`)

`J`
: The number of possible X scores. Only needed for "CB", "NEAT_CB" and "NEAT_PSE" designs

K
: The number of possible Y scores. Only needed for "CB", "NEAT_CB" and "NEAT_PSE" designs

L
: The number of possible A scores. Needed for "NEAT_CB" and "NEAT_PSE" designs

wx
: A number that satisfies $0 \leq w_X \leq 1$ indicating the weight put on the data that is not subject to order effects. Only used for the "CB" design.

wy
: A number that satisfies $0 \leq w_Y \leq 1$ indicating the weight put on the data that is not subject to order effects. Only used for the "CB" design.

w
: A number that satisfies $0 \leq w \leq 1$ indicating the weight given to population P. Only used for the "NEAT" design.

gapsX
: A *list* object containing:

  `index` A vector of indices between 0 and J to smooth "gaps", usually ocurring at regular intervals due to scores rounded to integer values and other methodological factors.

  `degree` An integer indicating the maximum degree of the moments fitted by the log-linear model.

  Only used for the "NEAT" design.

gapsY
: A *list* object containing:

  `index` A vector of indices between 0 and K.

  `degree` An integer indicating the maximum degree of the moments fitted.

  Only used for the "NEAT" design.

gapsA
: A *list* object containing:

  `index` A vector of indices between 0 and L.

  `degree` An integer indicating the maximum degree of the moments fitted.

  Only used for the "NEAT" design.

lumpX
: An integer to represent the index where an artificial "lump" is created in the marginal distribution of frecuencies for X due to recording of negative rounded formulas or any other methodological artifact.

lumpY
: An integer to represent the index where an artificial "lump" is created in the marginal distribution of frecuencies for Y.

lumpA
: An integer to represent the index where an artificial "lump" is created in the marginal distribution of frecuencies for A.

alpha
: Only for Adaptative Kernel. Sensitivity parameter.

h.adap
: Only for Adaptative Kernel. A list(hx, hy) containing bandwidths for Adaptative kernel for each Form.

Details

This is a generic function that implements the kernel method of test equating as described in Von Davier et al. (2004). Given test scores X and Y, the functions calculates

$$\hat{e}_Y(x) = G_{h_Y}^{-1}(F_{h_X}(x;\hat{r}), \hat{s})$$

where $\hat{r}$ and $\hat{s}$ are estimated score probabilities obtained via loglinear smoothing (see loglin.smooth). The value of h_X and h_Y can either be specified by the user or left unspecified (default) in which case they are automatically calculated. For instance, one can specifies large values of h_X and h_Y, so that the $\hat{e}_Y(x)$ tends to the linear equating function (see Theorem 4.5 in Von Davier et al, 2004 for more details).

Value

An object of class ker.eq representing the kernel equating process. Generic functions such as print, and summary have methods to show the results of the equating. The results include summary statistics, equated values, standard errors of equating, and others.

The function SEED can be used to obtain standard error of equating differences (SEED) of two objects of class ker.eq. The function PREp can be used on a ker.eq object to obtain the percentage relative error measure (see Von Davier et al, 2004).

Scores	The possible values of x_j and y_k
eqYx	The equated values of test X in test Y scale
eqXy	The equated values of test Y in test X scale
SEEYx	The standard error of equating for equating X to Y
SEEXy	The standard error of equating for equating Y to X

Author(s)

Jorge Gonzalez <jorge.gonzalez@mat.uc.cl>

References

Gonzalez, J. (2014). SNSequate: Standard and Nonstandard Statistical Models and Methods for Test Equating. *Journal of Statistical Software, 59(7),* 1-30.

Holland, P. and Thayer, D. (1989). The kernel method of equating score distributions. (Technical Report No 89-84). Princeton, NJ: Educational Testing Service.

Holland, P., King, B. and Thayer, D. (1989). The standard error of equating for the kernel method of equating score distributions (Tech. Rep. No. 89-83). Princeton, NJ: Educational Testing Service.

Von Davier, A., Holland, P., and Thayer, D. (2004). *The Kernel Method of Test Equating*. New York, NY: Springer-Verlag.

See Also

loglin.smooth, SEED, PREp

Examples

```
#Kernel equating under the "EG" design
data(Math20EG)
mod<-ker.eq(scores=Math20EG,kert="gauss",hx=NULL,hy=NULL,degree=c(2,3),design="EG")

summary(mod)
```

```
#Reproducing Table 7.6 in Von Davier et al, (2004)

scores<-0:20
SEEXy<-mod$SEEXy
SEEYx<-mod$SEEYx

Table7.6<-cbind(scores,SEEXy,SEEYx)
Table7.6

#Other nonstandard kernels. Table 10.3 in Von Davier (2011).

mod.logis<-ker.eq(scores=Math20EG,kert="logis",hx=NULL,hy=NULL,degree=c(2,3),design="EG")
mod.unif<-ker.eq(scores=Math20EG,kert="unif",hx=NULL,hy=NULL,degree=c(2,3),design="EG")
mod.gauss<-ker.eq(scores=Math20EG,kert="gauss",hx=NULL,hy=NULL,degree=c(2,3),design="EG")

XtoY<-cbind(mod.logis$eqYx,mod.unif$eqYx,mod.gauss$eqYx)
YtoX<-cbind(mod.logis$eqXy,mod.unif$eqXy,mod.gauss$eqXy)

Table10.3<-cbind(XtoY,YtoX)
Table10.3

## Examples using Adaptive and Epanechnikov kernels
x_sim = c(1,2,3,4,5,6,7,8,9,10,11,10,9,8,7,6,5,4,3,2,1)
prob_sim = x_sim/sum(x_sim)
set.seed(1)
sim = rmultinom(1, p = prob_sim, size = 1000)

x_asimD = c(1,7,13,18,22,24,25,24,20,18,16,15,13,9,5,3,2.5,1.5,1.5,1,1)
probas_asimD = x_asimD/sum(x_asimD)
set.seed(1)
asim = rmultinom(1, p = probas_asimD, size = 1000)

scores = cbind(asim,sim)

mod.adap  = ker.eq(scores,degree=c(2,2),design="EG",kert="adap")
mod.epan  = ker.eq(scores,degree=c(2,2),design="EG",kert="epan")
```

`le.eq` *Local equating methods*

Description

This function implements the local method of equating as descibed in van der Linden (2011).

Usage

```
le.eq(S.X, It.X, It.Y, Theta)
```

Arguments

`S.X`	A vector containing the observed scores of the sample taking test X.
`It.X`	A matrix of item parameter estimates coming from an IRT model for test form X (difficulty, discrimation and guessing parameters are located in the first, second and third column, respectively).
`It.Y`	A matrix of item parameter estimates coming from an IRT model for test form Y.
`Theta`	Either a number or vector of values representing the value of `theta` where to condition on (see details)

Details

The function implements the local equating method as described in van der Linden (2011). Based on Lord (1980) principle of equity, local equating methods utilizes the conditional on abilities distributions of scores to obtain the transformation φ. The method leads to a family of transformations of the form

$$\varphi(x;\theta) = G^{-1}_{Y|\theta}(F_{X|\theta}(x)), \quad \theta \in \mathcal{R}$$

The conditional distributions of X and Y are obtained using the algorithm described by Lord and Wingersky (1984). Among other possibilities, a value for θ can be a EAP, ML or MAP estimation of it, for and underlying IRT model (for example, using the `ltm` R package (Rizopoulos, 2006)).

Value

A list containing the observed scores to be equated, the corresponding ability estimates where to condition on, and the equated values

Author(s)

Jorge Gonzalez `<jorge.gonzalez@mat.uc.cl>`

References

Gonzalez, J. (2014). SNSequate: Standard and Nonstandard Statistical Models and Methods for Test Equating. *Journal of Statistical Software, 59(7),* 1-30.

Lord, F. (1980). *Applications of Item Response Theory to Practical Testing Problems*. Lawrence Erlbaum Associates, Hillsdale, NJ.

Lord, F. and Wingersky, M. (1984). Comparison of IRT True-Score and Equipercentile Observed-Score Equatings. *Applied Psychological Measurement,8(4),* 453–461.

Rizopoulos, D. (2006). ltm: An R package for latent variable modeling and item response theory analyses. *Journal of Statistical Software, 17(5),* 1–25.

van der Linden, W. (2011). Local Observed-Score Equating. In A. von Davier (Ed.) *Statistical Models for Test Equating, Scaling, and Linking*. New York, NY: Springer-Verlag.

See Also

`mea.eq`, `eqp.eq`, `lin.eq` `ker.eq`

Examples

```
## Artificial data for two 5-items tests forms. Both forms are assumed
## being fitted by a 3PL model.

## Create (artificial) item parameters matrices for test form X and Y
ai<-c(1,0.8,1.2,1.1,0.9)
bi<-c(-2,-1,0,1,2)
ci<-c(0.1,0.15,0.05,0.1,0.2)
itx<-rbind(bi,ai,ci)
ai<-c(0.5,1.4,1.2,0.8,1)
bi<-c(-1,-0.5,1,1.5,0)
ci<-c(0.1,0.2,0.1,0.15,0.1)
ity<-rbind(bi,ai,ci)

#Two individuals with different ability (1 and 2) obtain the same score 2.
#Their corresponding equated scores values are:
le.eq(c(2,2),itx,ity,c(1,2))
```

`lin.eq` *The linear method of equating*

Description

This function implements the linear method of test equating as described in Kolen and Brennan (2004).

Usage

```
lin.eq(sx, sy, scale)
```

Arguments

`sx`	A vector containing the observed scores of the sample taking test X.
`sy`	A vector containing the observed scores of the sample taking test Y.
`scale`	Either an integer or vector containing the values on the scale to be equated.

Details

The function implements the linear method of equating as described in Kolen and Brennan (2004). Given observed scores sx and sy, the functions calculates

$$\varphi(x;\mu_x,\mu_y,\sigma_x,\sigma_y) = \frac{\sigma_x}{\sigma_y}(x-\mu_x)+\mu_y$$

where $\mu_x, \mu_y, \sigma_x, \sigma_y$ are the score means and standard deviations on test X and Y, respectively.

Value

A two column matrix with the values of $\varphi()$ (second column) for each scale value `x` (first column)

Author(s)

Jorge Gonzalez <jorge.gonzalez@mat.uc.cl>

References

Gonzalez, J. (2014). SNSequate: Standard and Nonstandard Statistical Models and Methods for Test Equating. *Journal of Statistical Software, 59(7),* 1-30.

Kolen, M., and Brennan, R. (2004). *Test Equating, Scaling and Linking*. New York, NY: Springer-Verlag.

See Also

mea.eq, eqp.eq, ker.eq

Examples

```
#Artificial data for two two 100 item tests forms and 5 individuals in each group
x1<-c(67,70,77,79,65,74)
y1<-c(77,75,73,89,68,80)

#Score means and sd
mean(x1); mean(y1)
sd(x1); sd(y1)

#An equivalent form y1 score of 72 on form x1
lin.eq(x1,y1,72)

#Equivalent form y1 score for the whole scale range
lin.eq(x1,y1,0:100)

#A plot comparing mean, linear and identity equating
plot(0:100,0:100, type='l', xlim=c(-20,100),ylim=c(0,100),lwd=2.0,lty=1,
ylab="Form Y raw score",xlab="Form X raw score")
abline(a=5,b=1,lwd=2,lty=2)
abline(a=mean(y1)-(sd(y1)/sd(x1))*mean(x1),b=sd(y1)/sd(x1),,lwd=2,lty=3)
arrows(72, 0, 72, 77,length = 0.15,code=2,angle=20)
arrows(72, 77, -20, 77,length = 0.15,code=2,angle=20)
abline(v=0,lty=2)
legend("bottomright",lty=c(1,2,3), c("Identity","Mean","Linear"),lwd=c(2,2,2))
```

---

loglin.smooth *Pre-smoothing using log-linear models.*

Description

This function fits log-linear models to score data and provides estimates of the (vector of) score probabilities as well as the C matrix decomposition of their covariance matrix, according to the specified equating design (see Details).

Usage

```
loglin.smooth(scores, degree, design, scores2, degreeXA, degreeYA,
J, K, L, wx, wy, w, gapsX, gapsY, gapsA, lumpX, lumpY, lumpA,...)
```

Arguments

Note that depending on the specified equating design, not all arguments are necessary as detailed below.

`scores` If the "EG" design is specified, a vector containing the raw sample frequencies coming from one group taking the test.

If the "SG" design is specified, a matrix containing the (joint) bivariate sample frequencies for X (raws) and Y (columns).

If the "CB" design is specified, a two column matrix containing the observed scores of the sample taking test X first, followed by test Y. The `scores2` argument is then used for the scores of the sample taking test Y first followed by test X.

If either the "NEAT_CB" or "NEAT_PSE" design is selected, a two column matrix containing the observed scores on test X (first column) and the observed scores on the anchor test A (second column). The `scores2` argument is then used for the observed scores on test Y.

`degree` Either a number or vector indicating the number of power moments to be fitted to the marginal distributions, or the number or cross moments to be fitted to the joint distributions, respectively. For the "EG" design it will be a number (see Details).

`design` A character string indicating the equating design (one of "EG", "SG", "CB", "NEAT_CE", "NEAT_PSE")

`scores2` Only used for the "CB", "NEAT_CE" and "NEAT_PSE" designs. See the description of `scores`.

`degreeXA` A vector indicating the number of power moments to be fitted to the marginal distributions X and A, and the number or cross moments to be fitted to the joint distribution (X, A) (see details). Only used for the "NEAT_CE" and "NEAT_PSE" designs.

`degreeYA` Only used for the "NEAT_CE" and "NEAT_PSE" designs (see the description for `degreeXA`)

`J` The number of possible X scores. Only needed for "CB", "NEAT_CB" and "NEAT_PSE" designs

`K` The number of possible Y scores. Only needed for "CB", "NEAT_CB" and "NEAT_PSE" designs

`L` The number of possible A scores. Needed for "NEAT_CB" and "NEAT_PSE" designs

`wx` A number that satisfies $0 \leq w_X \leq 1$ indicating the weight put on the data that is not subject to order effects. Only used for the "CB" design.

`wy` A number that satisfies $0 \leq w_Y \leq 1$ indicating the weight put on the data that is not subject to order effects. Only used for the "CB" design.

`w` A number that satisfies $0 \leq w \leq 1$ indicating the weight given to population P. Only used for the "NEAT" design.

`gapsX` A *list* object containing:

- `index` A vector of indices between 0 and J to smooth "gaps", usually ocurring at regular intervals due to scores rounded to integer values and other methodological factors.
- `degree` An integer indicating the maximum degree of the moments fitted by the log-linear model.

Only used for the "NEAT" design.

`gapsY` A *list* object containing:

- `index` A vector of indices between 0 and K.
- `degree` An integer indicating the maximum degree of the moments fitted.

Only used for the "NEAT" design.

`gapsA` A *list* object containing:

- `index` A vector of indices between 0 and L.
- `degree` An integer indicating the maximum degree of the moments fitted.

Only used for the "NEAT" design.

`lumpX` An integer to represent the index where an artificial "lump" is created in the marginal distribution of frecuencies for X due to recording of negative rounded formulas or any other methodological artifact.

`lumpY` An integer to represent the index where an artificial "lump" is created in the marginal distribution of frecuencies for Y.

`lumpA` An integer to represent the index where an artificial "lump" is created in the marginal distribution of frecuencies for A.

`...` Further arguments to be passed.

Details

This function fits loglinear models as described in Holland and Thayer (1987), and Von Davier et al. (2004). The following general equation can be used to represent the models according to the different designs used, in which the vector o (or matrix) of (marginal or bivariate) score probabilities satisfies the log-linear model:

$$\log(o_{gh}) = \alpha_m + Z_m(z_g) + W_m(w_h) + ZW_m(z_g, w_h)$$

where $Z_m(z_g) = \sum_{i=1}^{T_{Zm}} \beta_{zmi}(z_g)^i$, $W_m(w_h) = \sum_{i=1}^{T_{Wm}} \beta_{Wmi}(w_h)^i$, and, $ZW_m(z_g, w_h) = \sum_{i=1}^{I_{Zm}} \sum_{i'=1}^{I_{Wm}} \beta_{ZWmii'}(z_g)^i (w_h)^{i'}$.

The symbols will vary according to the different equating designs specified. Possible values are: $o = p_{(12)}, p_{(21)}, p, q$; $Z = X, Y$; $W = Y, A$; $z = x, y$; $w = y, a$; $m = (12), (21), P, Q$; $g = j, k$; $h = l, k$.

Particular cases of this general equation for each of the equating designs can be found in Von Davier et al (2004) (e.g., Equations (7.1) and (7.2) for the "EG" design, Equation (8.1) for the "SG" design, Equations (9,1) and (9.2) for the "CB" design).

Value

`sp.est` The estimated score probabilities

`C` The C matrix which is so that $\Sigma = CC^t$

Author(s)

Jorge Gonzalez <jorge.gonzalez@mat.uc.cl>

References

Gonzalez, J. (2014). SNSequate: Standard and Nonstandard Statistical Models and Methods for Test Equating. *Journal of Statistical Software, 59(7),* 1-30.

Holland, P. and Thayer, D. (1987). Notes on the use of loglinear models for fitting discrete probability distributions. Research Report 87-31, Princeton NJ: Educational Testing Service.

Von Davier, A., Holland, P., and Thayer, D. (2004). *The Kernel Method of Test Equating*. New York, NY: Springer-Verlag.

[1] Moses, T. "Paper SA06_05 Using PROC GENMOD for Loglinear Smoothing Tim Moses and Alina A. von Davier, Educational Testing Service, Princeton, NJ".

See Also

glm, ker.eq

Examples

```
#Table 7.4 from Von Davier et al. (2004)
data(Math20EG)
rj<-loglin.smooth(scores=Math20EG[,1],degree=2,design="EG")$sp.est
sk<-loglin.smooth(scores=Math20EG[,2],degree=3,design="EG")$sp.est
score<-0:20
Table7.4<-cbind(score,rj,sk)
Table7.4

## Example taken from [1]
score <- 0:20
freq <- c(10, 2, 5, 8, 7, 9, 8, 7, 8, 5, 5, 4, 3, 0, 2, 0, 1, 0, 2, 1, 0)
ldata <- data.frame(score, freq)

plot(ldata, pch=16, main="Data w Lump at 0")
m1 = loglin.smooth(scores=ldata$freq,kert="gauss",degree=c(3),design="EG")
m2 = loglin.smooth(scores=ldata$freq,kert="gauss",degree=c(3),design="EG",lumpX=0)
Ns = sum(ldata$freq)
points(m1$sp.est*Ns, col=2, pch=16)
points(m2$sp.est*Ns, col=3, pch=16) # Preserves the lump
```

`Math20EG` *Scores on two 20-items mathematics tests.*

Description

The data set contains raw sample frequencies of number-right scores for two parallel 20-items mathematics tests given to two samples from a national population of examinees. This data has been described and analized by Holland and Thayer (1989); Von Davier et al, (2004) (see also Von Davier, 2011 where other applications using these data set are shown).

Usage

```
data(Math20EG)
```

Format

A 21x2 matrix containing raw sample frequencies (raws) for two parallel tests (columns)

References

Holland, P. and Thayer, D. (1989). The kernel method of equating score distributions. (Technical Report No 89-84). Princeton, NJ: Educational Testing Service.

Von Davier, A., Holland, P., and Thayer, D. (2004). *The Kernel Method of Test Equating*. New York, NY: Springer-Verlag.

Examples

```
data(Math20EG)
## maybe str(Math20EG) ; ...
```

`Math20SG` *Bivariate score frequencies on two 20-items mathematics tests.*

Description

The data set contains the bivariate sample frequencies of number-right scores for two parallel 20-items mathematics tests given to a sample from a national population of examinees. This data has been described and analized by Holland and Thayer (1989); Von Davier et al, (2004).

Usage

```
data(Math20SG)
```

Format

A 21x21 matrix containing the bivariate sample frequencies for X (raws) and Y (columns)

References

Holland, P. and Thayer, D. (1989). The kernel method of equating score distributions. (Technical Report No 89-84). Princeton, NJ: Educational Testing Service.

Von Davier, A., Holland, P., and Thayer, D. (2004). *The Kernel Method of Test Equating*. New York, NY: Springer-Verlag.

Examples

```
data(Math20SG)
## maybe str(Math20SG) ; ...
```

`mea.eq` *The mean method of equating*

Description

This function implements the mean method of test equating as described in Kolen and Brennan (2004).

Usage

```
mea.eq(sx, sy, scale)
```

Arguments

`sx`	A vector containing the observed scores of the sample taking test X.
`sy`	A vector containing the observed scores of the sample taking test Y.
`scale`	Either an integer or vector containing the values on the scale to be equated.

Details

The function implements the mean method of equating as described in Kolen and Brennan (2004). Given observed scores sx and sy, the functions calculates

$$\varphi(x; \mu_x, \mu_y) = x - \mu_x + \mu_y$$

where μ_x and μ_y are the score means on test X and Y, respectively.

Value

A two column matrix with the values of $\varphi()$ (second column) for each scale value `x` (first column)

Author(s)

Jorge Gonzalez <jorge.gonzalez@mat.uc.cl>

References

Gonzalez, J. (2014). SNSequate: Standard and Nonstandard Statistical Models and Methods for Test Equating. *Journal of Statistical Software, 59(7),* 1-30.

Kolen, M., and Brennan, R. (2004). *Test Equating, Scaling and Linking*. New York, NY: Springer-Verlag.

See Also

lin.eq, eqp.eq, ker.eq, le.eq

Examples

```
#Artificial data for two two 100 item tests forms and 5 individuals in each group
x1<-c(67,70,77,79,65,74)
y1<-c(77,75,73,89,68,80)

#Score means
mean(x1); mean(y1)

#An equivalent form y1 score of 72 on form x1
mea.eq(x1,y1,72)

#Equivalent form y1 score for the whole scale range
mea.eq(x1,y1,0:100)
```

PREp *Percent relative error*

Description

This function calculates the percent relative error as described in Von Davier et al. (2004).

Usage

```
PREp(eq, p)
```

Arguments

eq	An object of class ker.eq previously obtained using ker.eq.
p	The number of moments to be calculated.

Details

PREp (when equating form X to Y) is calculated as

$$\text{PREp} = 100\frac{\mu_p(e_Y(X)) - \mu_p(Y)}{\mu_p(Y)}$$

where $\mu_p(Y) = \sum_k (y_k)^p s_k$ and $\mu_p(e_Y(X)) = \sum_j (e_Y(x_j))^p r_j$. Similar formulas can be found when equating from Y to X.

Value

A matrix containing the PREp for both `X` to `Y` (first column) and `Y` to `X` (second column) cases.

Author(s)

Jorge Gonzalez <jorge.gonzalez@mat.uc.cl>

References

Gonzalez, J. (2014). SNSequate: Standard and Nonstandard Statistical Models and Methods for Test Equating. *Journal of Statistical Software, 59(7),* 1-30.

Von Davier, A., Holland, P., and Thayer, D. (2004). *The Kernel Method of Test Equating*. New York, NY: Springer-Verlag.

See Also

`ker.eq`

Examples

```
#Example: Table 7.5 in Von Davier et al. (2004)

data(Math20EG)
mod.gauss<-ker.eq(scores=Math20EG,kert="gauss", hx = NULL, hy = NULL,degree=c(2, 3),design="EG")
PREp(mod.gauss,10)
```

`rowBlockSum` *Take a matrix and sum blocks of rows*

Description

This function implements a method to sum blocks of rows in a matrix

Usage

```
rowBlockSum(mat, blocksize, w = NULL)
```

Arguments

`mat`	Input matrix
`blocksize`	Size of the row blocks
`w`	(Optional) Vector for weighted sum

Details

The original data set contains very long column headers. This function does a keyword search over the headers to find those column headers that match a particular keyword, e.g., mean, median, etc.

Value

A matrix.

Author(s)

Daniel Acuna Leon. <dnacuna@uc.cl>

SEED *Standard error of equating difference*

Description

This function calculates the standard error of equating diference (SEED) as described in Von Davier et al. (2004).

Usage

```
SEED(eq1, eq2)
```

Arguments

eq1	An object of class `ker.eq` which contains one of the two estimated equated functions to be used for the SEED.
eq2	An object of class `ker.eq` which contains one of the two estimated equated functions to be used for the SEED.

Details

The SEED can be used as a measure to choose whether to support or not a certain equating function on another another one. For instance, when h_X and h_Y tends to infinity, then the (gaussian kernel) $\hat{e}_Y(x)$ equating function tends to the linear equating function (see Theorem 4.5 in Von Davier et al, 2004 for more details). Thus, one can calculate the measure

$$SEED_Y(x) = \sqrt{Var(\hat{e}_Y(x) - \widehat{Lin}_Y(x))}$$

to decide between $\hat{e}_Y(x)$ and $\widehat{Lin}_Y(x)$.

Value

A two column matrix with the values of `SEEYx` for each `x` in the first column and the values of `SEEXy` for each `y` in the second column

Author(s)

Jorge Gonzalez <jorge.gonzalez@mat.uc.cl>

References

Gonzalez, J. (2014). SNSequate: Standard and Nonstandard Statistical Models and Methods for Test Equating. *Journal of Statistical Software, 59(7),* 1-30.

Von Davier, A., Holland, P., and Thayer, D. (2004). *The Kernel Method of Test Equating*. New York, NY: Springer-Verlag.

See Also

ker.eq

Examples

```
#Example: Figure7.7 in Von Davier et al, (2004)
data(Math20EG)

mod.gauss<-ker.eq(scores=Math20EG,kert="gauss", hx = NULL, hy = NULL,degree=c(2, 3),design="EG")
mod.linear<-ker.eq(scores=Math20EG,kert="gauss", hx = 20, hy = 20,degree=c(2, 3),design="EG")

Rx<-mod.gauss$eqYx-mod.linear$eqYx
seed<-SEED(mod.gauss,mod.linear)$SEEDYx

plot(0:20,Rx,ylim=c(-0.8,0.8),pch=15)
abline(h=0)
points(0:20,2*seed,pch=0)
points(0:20,-2*seed,pch=0)

#Example Figure 10.4 in Von Davier (2011)
mod.unif<-ker.eq(scores=Math20EG,kert="unif", hx = NULL, hy = NULL,degree=c(2, 3),design="EG")
mod.logis<-ker.eq(scores=Math20EG,kert="logis", hx = NULL, hy = NULL,degree=c(2, 3),design="EG")

Rx1<-mod.logis$eqYx-mod.gauss$eqYx
Rx2<-mod.unif$eqYx-mod.gauss$eqYx

seed1<-SEED(mod.logis,mod.gauss)$SEEDYx
seed2<-SEED(mod.unif,mod.gauss)$SEEDYx

plot(0:20,Rx1,ylim=c(-0.2,0.2),pch=15,main="LK vs GK",ylab="",xlab="Scores")
abline(h=0)
points(0:20,2*seed1,pch=0)
points(0:20,-2*seed1,pch=0)

plot(0:20,Rx2,ylim=c(-0.2,0.2),pch=15,main="UK vs GK",ylab="",xlab="Scores")
abline(h=0)
points(0:20,2*seed2,pch=0)
points(0:20,-2*seed2,pch=0)
```

`SEPA` *A sample of observed score values for two different forms of the SEPA test.*

Description

The data set is from a private national evaluation system called SEPA. It contains two test forms X and Y both composed of 50 items. The SEPA data is a list containing two samples with 1,458 test takers who took test form X and 2,619 test takers who took test form Y.

Usage

```
data(SEPA)
```

Format

A list with elements containing the observed scores in test forms X and Y.

References

Gonzalez, J. and Wiberg, M. (2017). *Applying test equating methods using R*. Springer.

Examples

```
data(SEPA)
## maybe str(SEPA) ; ...
```

`sim_unimodal` *Simulate test scores.*

Description

Simulate test scores from a negative-hypergeometric (beta-binomial) distribution, according to Keats & Lord (1962).

Usage

```
sim_unimodal(n, x_mean, x_var, N_item, seed = NULL, name = NULL)
```

Arguments

n	Size of the resulting sample.
x_mean	Mean of the target distribution.
x_var	Variance of the target distribution.
N_item	Number of items in the test.
seed	Optional. Seed for the random number generator.
name	Optional. Generate X and Y scores from the data according 5 of the proposed distributions in Keats & Lord (1967). Overrides any other previous parameter input set.

Details

Simulate test scores from a negative-hypergeometric (beta-binomial) distribution, according to Keats & Lord (1962).

Value

Simulated values.

Author(s)

Daniel Leon Acuna, <dnacuna@uc.cl>

References

Keats, J. A., & Lord, F. M. (1962). A theoretical distribution for mental test scores. Psychometrika, 27(1), 59-72.

Examples

```
sim_unimodal(2354, 27.06, 8.19^2, 40)  # GANA
sim_unimodal(name="TQS8")
```

Index