Package 'equate'

February 24, 2026

Type Package

Version 2.0.9

Title Observed-Score Linking and Equating

URL https://github.com/talbano/equate

BugReports https://github.com/talbano/equate/issues

Depends R (>= 4.1.0)

Imports utils, grDevices, graphics, stats

Encoding UTF-8

Description Contains methods for observed-score linking and equating under the single-group, equivalent-groups, and nonequivalent-groups with anchor test(s) designs. Equating types include identity, mean, linear, general linear, equipercentile, circle-arc, and composites of these. Equating methods include synthetic, nominal weights, Tucker, Levine observed score, Levine true score, Braun/Holland, frequency estimation, and chained equating. Plotting and summary methods, and methods for multivariate presmoothing and bootstrap error estimation are also provided.

LazyLoad yes

LazyData yes

License GPL-3

RoxygenNote 7.3.3

Suggests knitr, rmarkdown, bookdown

VignetteBuilder knitr

NeedsCompilation no

Author Anthony Albano [aut, cre]

Maintainer Anthony Albano <tony.d.albano@gmail.com>

Repository CRAN

Date/Publication 2026-02-24 06:30:02 UTC

Contents

`ACTmath` *ACT Mathematics Test Scores*

Description

This dataset contains score distributions for two forms of the ACT mathematics test, as presented in table 2.5 of *Test Equating, Scaling, and Linking* (Kolen and Brennan, 2004; p. 50).

Usage

```
ACTmath
```

Format

A 41 by 3 `data.frame` containing the score scale, frequencies for form X, and frequencies for form Y.

Source

Kolen, M. J., and Brennan, R. L. (2004). *Test Equating, Scaling, and Linking*. (2nd ed.), New York: Springer.

The dataset is also provided with the equating software RAGE, available at the following link: https://education.uiowa.edu/casma/computer-programs

bootstrap *Bootstrap Equating Error*

Description

These functions return bootstrap standard errors, bias, and RMSE of equating. A summary method estimates mean and weighted mean errors over the score scale.

Usage

```
bootstrap(x, ...)

## Default S3 method:
bootstrap(x, y, ...)

## S3 method for class 'equate'
bootstrap(x, xp = x$x, yp = x$y, ...)

## S3 method for class 'freqtab'
bootstrap(
  x,
  y,
  xn = sum(x),
  yn = sum(y),
  reps = 100,
  crit,
  args,
  eqs = FALSE,
  sharesmooth = FALSE,
  ...
)

## S3 method for class 'bootstrap'
summary(object, weights, subset, ...)
```

Arguments

x	either an equating object, obtained with the equate function, or a score distribution of class "freqtab".
...	further arguments passed to or from other methods.
y	score distribution of class "freqtab".
xp, yp	optional frequency tables replacing those equated in x, used for parametric bootstrap resampling.
xn, yn	integers specifying the number of scores to sample from each distribution at each replication (default is the total number observed in each).

reps	number of bootstrap replications.
crit	vector of equated scores serving as the criterion equating function when calculating bootstrap bias and RMSE, both of which are returned when crit is specified.
args	named list of equating arguments, passed to equate, specifying, e.g., the equating type and method. See below for details.
eqs	logical, with default FALSE, indicating whether or not the matrices of equating functions (one column per replication, per equating) should be returned.
sharesmooth	logical, defaulting to FALSE, indicating whether or not loglinear presmoothing should be performed once per replication using arguments given in args. Ignored if smoothmethod = "loglinear" is not given in any args.
object	bootstrap output to be summarized.
weights	vector of weights to be used in calculating weighted average errors with summary, defaulting to the frequencies in margin(object$x).
subset	vector indicating a subset of the score scale for which errors should be summarized.

Details

Samples are drawn of size xn and yn, with replacement, from each score distribution. Form Y equivalents of each form X score are then obtained using either the arguments in the equating output or those provided. This process is repeated reps times. Standard errors are calculated as standard deviations over replications for each score point; bias is the mean equated score over replications, minus the criterion; and RMSE is the square root of the squared standard error and squared bias combined.

The bootstrap method for objects of class "equate" is designed to be called from within equate. It simply extracts the necessary arguments from the equating output before bootstrapping.

When each element in args is a named list of equating arguments, multiple equatings are performed at each replication in the bootstrapping.

The summary method returns a data.frame of mean standard errors, bias, and rmse, and weighted means, as applicable.

Value

With bootstrap, a list is returned, containing arguments supplied for x, y, reps, xn, yn, and args. For a single equating, the mean equating function over replications and a vector of standard errors se are included, along with vectors of bias and rmse, when crit is provided, and a matrix of equating functions eqs when eqs = TRUE. For multiple equatings, where each element of args is a list of equating arguments, matrices are returned for the mean functions, standard error, bias, and RMSE, and the equating functions will be returned as a list of matrices. The summary method returns a data frame of mean standard errors, bias, and rmse, and weighted means, as applicable.

Methods (by class)

- bootstrap(default): Default bootstrap method for "freqtab" objects.
- bootstrap(equate): Method for "equate" objects.
- bootstrap(freqtab): Bootstrap method for "freqtab" objects.

Author(s)

Anthony Albano <tony.d.albano@gmail.com>

See Also

plot.bootstrap

Examples

```
# Parametric bootstrapping using smoothed
# frequency distributions
set.seed(111213)
x <- freqtab(KBneat$x, scales = list(0:36, 0:12))
y <- freqtab(KBneat$y, scales = list(0:36, 0:12))
xp <- loglinear(x, asfreqtab = TRUE)
yp <- loglinear(y, asfreqtab = TRUE)
crit <- equate(xp, yp, "e", "c")$conc$yx
eqargs <- list(m.t = list(type = "m", method = "t"),
  l.t = list(type = "l", method = "t"))
bootout1 <- bootstrap(x = x, y = y, xn = 20, yn = 20,
  crit = crit, args = eqargs, reps = 30)
plot(bootout1, out = "rmse", legendplace = "top",
  addident = FALSE)

# Bootstraps for an existing equating
eq <- equate(x, y, type = "m", method = "t")
bootout2 <- bootstrap(eq, xn = 100, yn = 100,
  crit = crit, reps = 20)
summary(bootout2)
```

---

composite *Composite Linking and Equating*

Description

This function creates a composite linking or equating as a combination of two or more other linking or equating functions.

Usage

```
composite(x, ...)

## Default S3 method:
composite(x, wc, ...)

## S3 method for class 'equate.list'
composite(x, wc, name, symmetric = FALSE, p = 1, verbose = TRUE, ...)
```

```
## S3 method for class 'list'
composite(x, wc, name, symmetric = FALSE, p = 1, verbose = TRUE, ...)
```

Arguments

x	for the default method, x is a matrix of equating functions, with one function per column. Otherwise, x is a list of equatings, where each element is an object of class "equate".
...	further arguments passed to or from other functions.
wc	vector of weights for creating the composite. length(wc) should match either ncol(x) for the default method or length(x).
name	an optional name, used to label the output. If missing, a name will be created using x.
symmetric	logical, with default FALSE, indicating whether or not weights wc should be modified to create symmetric weights. Only supported for composites of linear functions.
p	integer specifying the type of circle used to define symmetry.
verbose	logical, with default TRUE, indicating whether or not full output should be returned. When FALSE, only the equated scores are returned.

Details

Composite linking and equating create a single linking or equating function as a weighted combination of two or more other linking or equating functions. See Holland and Strawderman (2011) for details.

Value

For the default method, and when verbose = FALSE, a vector of composite equated scores is returned. Otherwise, a list of equating output is returned, including output for the composite and each function being combined.

Methods (by class)

- composite(default): Default method for a matrix of equating functions, one per column.
- composite(equate.list): Create composite across functions in "equate.list" object.
- composite(list): Create composite across functions in "list" object.

Author(s)

Anthony Albano <tony.d.albano@gmail.com>

References

Holland, P. W., and Strawderman, W. E. (2011). How to average equating functions, if you must. In A. A. von Davier (Ed.), Statistical models for test equating, scaling, and linking (pp. 89-107). New York, NY: Springer.

See Also

equate

Examples

```
# See vignette("equatevignette") for additional examples

# Example from the equate help file, without the bootstrapping
# Random groups equating for (1) identity, (2) mean,
# (3) linear, (4) equipercentile with loglinear
# smoothing, and (5) a composite of mean and identity
rx <- as.freqtab(ACTmath[, 1:2])
ry <- as.freqtab(ACTmath[, c(1, 3)])
set.seed(2007)

req1 <- equate(rx, ry, type = "i")
req2 <- equate(rx, ry, type = "m")
req3 <- equate(rx, ry, type = "l")
req4 <- equate(rx, ry, type = "e", smooth = "loglin",
  degrees = 3)
req5 <- composite(list(req1, req2), wc = .5, symmetric = TRUE)

# Compare equating functions
plot(req1, req2, req3, req4, req5[[1]], addident = FALSE)
```

---

equate *Observed Score Linking and Equating*

Description

This function links the scale of x to the scale of y for the single-group, equivalent-groups, and nonequivalent-groups with anchor test designs. A summary method is also provided.

Usage

```
equate(x, ...)

## S3 method for class 'list'
equate(x, args, ...)

## S3 method for class 'freqtab'
equate(
  x,
  y,
  type = c("identity", "mean", "linear", "general linear", "circle-arc",
    "equipercentile"),
 method = c("none", "nominal weights", "tucker", "levine", "frequency estimation",
```

```
    "chained", "braun/holland"),
  name,
  lowp,
  highp,
  boot = FALSE,
  verbose = TRUE,
  ...
)

## Default S3 method:
equate(x, y, ...)

## S3 method for class 'equate'
summary(object, ...)

## S3 method for class 'equate.list'
summary(object, ...)
```

Arguments

x, y	for the default method, x must be a vector of scores and y an object of class "equate", the output of a previous equating. The standard usage is to provide x as a frequency table of class "freqtab", where y is also a frequency table, and x is equated to y; if y is missing, a single-group design is assumed. Finally, x may be a list of two or more frequency tables, in which case the required arguments for one or more equatings are listed in args. See below for details.
...	further arguments passed to or from other functions, including arguments specific to different equating methods. See below for details.
args	list of arguments passed to equate. See below for details.
type	the type of equating. See below for details.
method	the equating method, where "none" (default) indicates equating under the single-group or equivalent-groups design, and "nominal weights", "tucker", "levine", "frequency estimation", "braun/holland", and "chained" indicate the corresponding methods under the nonequivalent groups design.
name	an optional name, used to label the output. If missing, a name will be created using type and method.
lowp, highp	two vectors, each of length 2, specifying the coordinates for the low and high points of the X and Y scales. lowp defaults to the minimums and highp the maximums of the scales. Recycled if necessary. When lowp = "obs" or highp = "obs", minimum and maximum observed scores are used.
boot	logical indicating whether or not bootstrapping should be performed. Default is FALSE. See below and the bootstrap function for details.
verbose	logical, with default TRUE, indicating whether or not full output should be returned. When FALSE, only the equated scores are returned.
object	output from either an equating or list of equatings, produced by the equate function.

Details

Equating is typically performed on two frequency tables, x and y. In this case, the scores from both are used to define the equating function that links the scale of x to y. The equivalent-groups design is assumed when x and y are separate, univariate frequency tables. The nonequivalent-groups design is assumed when a method is specified, and x and y are separate multivariate frequency tables. Finally, a single-group design is assumed when x is a bivariate frequency table (containing scores on X and Y) and y is missing.

The single-group design currently only differs from the equivalent groups design in that presmoothing can be used to preserve bivariate moments for x and y in the single-group design, whereas in the equivalent-groups design, with x and y specified separately, presmoothing is performed separately. If presmoothing is not performed via equate, the single-group and equivalent-groups designs produce the same result.

When x is a vector of scores and equating output is supplied for y, no other arguments are required. Scores from x are converted directly to the scale indicated in y. If y is a composite equating, composite equated scores will be returned based on the weighted combination of equating functions included in y.

When x is a list of frequency tables, each element in args must be a named list of equating arguments. In this case, the length of args corresponds to the number of equatings that will be performed. The arguments for each equating are specified as they would be when x and y are frequency tables, except for x and y; the frequency tables to be equated are specified in args by referencing their names in the list of frequency tables. See below for examples.

Six equating types are currently supported: identity, mean, linear, and equipercentile, as described by Kolen and Brennan (2004); circle-arc equating, as described by Livingston and Kim (2009); and a general linear function that extends the traditional identity, mean, and linear types. Corresponding linking methods are also supported. The equating design is implied by the method argument, where "none" (default) indicates that no method is needed (because examinees taking forms X and Y are assumed to be the same or equivalent). The nominal weights, Tucker, Levine observed score, Levine true score, frequency estimation, Braun/Holland, and chained equating methods are supported for the nonequivalent-groups with anchor test design. All but the Levine true score and chained method rely on a "synthetic" distribution of scores (Braun and Holland, 1982), a weighted combination of x and y.

Depending on the equating method, the following additional arguments may be required:

midp coordinates for the midpoint of the equating line, used in general linear and circle-arc equating.

cx, cy, sx, sy parameters used in general linear equating. See below for details.

wax, way, wbx, wby weights used when finding the slope and intercept in general linear equating. See below.

ws value between 0 and 1 specifying the weight applied to form X scores (and implicitly specifying the form Y weight as 1 - ws) when estimating the synthetic population. When set to -1 (the default), proportional weights are calculated for X and Y based on sample size.

internal logical indicating whether or not the anchor item scores are included in the total scores. This applies only to the Levine method, as all other methods assume an internal anchor test. Default is TRUE.

lts logical indicating whether or not to use levine true score ("lts") equating. Default is FALSE.

smoothmethod string indicating one of four smoothing methods to be used in equipercentile equating: "none" (default), "average", "bump", and "loglinear" (see below).

chainmidp string specifying the type of chained linear equating used to obtain the midpoint in chained circle-arc equating, whether "mean" (default) or "linear".

simple logical, with default TRUE, indicating whether or not simplified circle-arc equating should be used (see below).

reps the number of replications to use in bootstrapping. Passed to bootstrap.

xp, yp optional parametric distributions, as frequency tables, replacing x and y when bootstrapping.

xn, yn sample sizes to be drawn from x and y, or xp and yp, at each bootstrap replication. These default to the observed sample sizes.

crit a vector of equated scores serving as the criterion equating function when calculating bootstrap bias and RMSE; both are returned when crit is specified.

General linear equating is a new approach to estimating a linear linking or equating function. The slope and intercept of the line are estimated based on multiple sources of information, including the means and standard deviations of X and Y, and other values supplied through cx and cy, representing the centrality of X and Y, and sx and sy, representing the scaling or variability of X and Y. The weights wax and way specify the proportional weighting given to the standard deviations of X and Y, and indirectly the weighting given to sx and sy, in determining the slope. wbx and wby specify the proportional weighting given to the means of X and Y, and indirectly the weighting given to cx and cy, in determining the intercept. Synthetic means and standard deviations will be used when appropriate. Chained general linear equating is not currently supported.

For equipercentile equating under the random groups design, three smoothing options are available: smoothmethod = "average" and smoothmethod = "bump" require the additional argument jmin, and loglinear smoothing (smoothmethod = "loglinear") requires either a score function or maximum polynomial terms. For frequency estimation and chained methods, only smoothing methods "bump" and "loglinear" are supported. See the presmoothing function for details and examples.

In equipercentile equating, the high point for y, i.e., highp[2], is used to obtain form Y equivalents of form X scores with percentile ranks of 100. Typically this is set to be the number of score points in the form Y scale, which assumes that scores are integers ranging from 1 (or 0) to the total number of items, and that each item is scored correct/incorrect. Scores on other scales (such as scales which include negative values, or which exclude zero) may also be used. In such cases highp[2] can be set to the highest possible score on form Y, or alternatively the highest observed score on Y.

lowp and highp are used to define the slope and intercept of the identity linking function. When the score scales for X and Y are equivalent, the identity function is simply the unequated X scale; however, when forms differ in their scales, e.g., because of changes in content or length, the identity linking function will map X onto Y based on the low and high coordinates.

The simplified approach to circle-arc equating, as demonstrated by Livingston and Kim (2009), involves combining a circle-arc with the identity function. When the low and high scores differ for the X and Y scales, this becomes the identity linking function. The linear component can be omitted, and symmetric circle-arc equating used, with simple = FALSE. The result is an equating function based only on the circle-arc that passes through the points lowp, highp, and the estimated midpoint.

Analytical standard errors are currently returned for linear equating under equivalent groups and chained, Tucker, and Levine equating with nonequivalent groups. Chained, Tucker, and Levine

standard errors are provided with and without assumptions of normality, as shown in Zu (2012). With boot = TRUE, bootstrap standard errors are estimated using a default of reps = 100 replications, sampling the maximum amount from each score distribution (controlled by the arguments xn and yn). See bootstrap for details and examples, including how to obtain bootstrap bias and RMSE.

Value

When y contains output from an equating, a vector of equated scores is returned. Otherwise, an object of class "equate" is returned, listing the following components, some of which are dependent on the equating type, method, and smoothing:

name	name for the equating
type	equating type
method	equating method
design	equating design, as specified in x
x, y	original frequency tables for X and Y
concordance	conversion table containing scores on X with their form Y equivalents, and standard errors, when available.
points	low and high points defining the identity line, and midpoints for general linear and circle-arc equating
weight	weights used in general linear equating
internal, lts, jmin, degree, xdegree, scorefun	additional arguments, as supplied in ...
coefficients	conversion coefficients intercept and slope; for circle-arc equating, circle center points and radius are also included; for general linear equating, slope and intercept components are included
ws	weight applied to X in synthetic estimation
synthstats	means and standard deviations for the synthetic distributions
xsynthetic, ysynthetic	frequency tables for the synthetic distributions
smoothmethod	smoothing method
xsmooth, ysmooth	smoothed frequency tables for X and Y
bootstraps	list containing bootstrap standard errors, and, optionally, other bootstrap output

The summary method returns a list with the name, type, method, design, and synthetic weight, along with frequency tables for the total, anchor, and equated score distributions and descriptive statistics for each.

Methods (by class)

- equate(list): Equating a list of frequency tables.
- equate(freqtab): Equating frequency distributions in x and y.
- equate(default): Default equating method for a vector of raw scores x and equating output in y.

Author(s)

Anthony Albano <tony.d.albano@gmail.com>

References

Albano, A. D. (2016). equate: An R package for observed-score linking and equating. *Journal of Statistical Software, 74*(8), 1–36.

Kolen, M. J., and Brennan, R. L. (2004). *Test Equating, Scaling, and Linking*. (2nd ed.), New York: Springer.

Livingston, S. A., and Kim, S. (2009). The circle-arc method for equating in small samples, *Journal of Educational Measurement, 46*, 330–343.

Zu, J., and Yuan, K. H. (2012). Standard error of linear observed-score equating for the NEAT design with nonnormally distributed data. *Journal of Educational Measurement, 49*, 190–213.

See Also

freqbump, freqavg, loglinear, bootstrap

Examples

```
# See vignette("equatevignette") and Albano (2016) for a
# description of methods and additional examples

# Random groups equating for (1) identity, (2) mean,
# (3) linear, (4) equipercentile with loglinear
# smoothing, and (5) a composite of mean and identity
rx <- as.freqtab(ACTmath[, 1:2])
ry <- as.freqtab(ACTmath[, c(1, 3)])
set.seed(2007)

req1 <- equate(rx, ry, type = "i", boot = TRUE, reps = 5)
req2 <- equate(rx, ry, type = "m", boot = TRUE, reps = 5)
req3 <- equate(rx, ry, type = "l", boot = TRUE, reps = 5)
req4 <- equate(rx, ry, type = "e", boot = TRUE, reps = 5,
  smooth = "loglin", degree = 3)
req5 <- composite(list(req1, req2), wc = .5, symmetric = TRUE)

# Compare equating functions
plot(req1, req2, req3, req4, req5[[1]], addident = FALSE)

# Compare boostrap standard errors
# Bootstrapping isn't supported for composite equating
plot(req1, req2, req3, req4, addident = FALSE, out = "se",
  legendplace = "topleft")

# Nonequivalent groups design for (1) Tucker linear,
# (2) frequency estimation , and (3) Braun/Holland linear
nx <- freqtab(KBneat$x, scales = list(0:36, 0:12))
ny <- freqtab(KBneat$y, scales = list(0:36, 0:12))
```

```
neq1 <- equate(nx, ny, type = "linear", method = "tuck", ws = 1)
neq2 <- equate(nx, ny, type = "equip", method = "freq", ws = 1)
neq3 <- equate(nx, ny, type = "linear", method = "braun", ws = 1)

# Compare equated scores
round(cbind(xscale = 0:36, tucker = neq1$conc$yx,
fe = neq2$conc$yx, braun = neq3$conc$yx), 2)

# Multiple linkings using PISA reading booklet 6
# clusters 3a, 5, 6, and 7
r3 <- freqtab(PISA$totals$b6$r3a, scales = 0:15)
r5 <- freqtab(PISA$totals$b6$r5, scales = 0:15)
r6 <- freqtab(PISA$totals$b6$r6, scales = 0:15)
r7 <- freqtab(PISA$totals$b6$r7, scales = 0:14)
eqargs <- list(r3r5 = list(type = "linear", x = "r3", y = "r5",
    name = "Linear Linking PISA r3 to r5"),
  r5r6 = list(type = "linear", x = "r5", y = "r6",
    name = "Linear Linking PISA r5 to r6"),
  r6r7 = list(type = "linear", x = "r6", y = "r7",
    name = "Linear Linking PISA r6 to r7"))
req <- equate(list(r3 = r3, r5 = r5, r6 = r6, r7 = r7), eqargs)

# Put PISA r3 on the scale of r7 using the linking chain
# Compare to a direct linking of r3 to r7
equate(equate(req$r3r5$conc$yx, req$r5r6), req$r6r7)
equate(r3, r7, "linear")$conc$yx

# Linking PISA cluster r3a to r5 with multiple anchors
m367 <- freqtab(PISA$totals$b6[1:198, c("r3a", "r6", "r7")],
scales = list(0:15, 0:16, 0:14))
m567 <- freqtab(PISA$totals$b6[199:396, c("r5", "r6", "r7")],
scales = list(0:15, 0:16, 0:14))
meq1 <- equate(m367, m567, type = "mean", method = "nom")
meq2 <- equate(m367, m567, type = "mean", method = "tuck")
meq3 <- equate(m367, m567, type = "lin", method = "tuck")
meq4 <- equate(m367, m567, type = "equip", method = "freq",
smooth = "log", show = FALSE)
meq <- equate(m367, m567, type = "mean", method = "nom")
plot(meq1, meq2, meq3, meq4, meq, req[[1]])
```

`freqtab` *Frequency Distribution Tables*

Description

Functions for creating and manipulating frequency tables of class "`freqtab`".

Usage

```
freqtab(x, ...)

## Default S3 method:
freqtab(x, scales, items, design, na.rm = TRUE, ...)

as.freqtab(x, scales, design, drop = FALSE, ...)

## S3 method for class 'table'
freqtab(x, design, ...)

## S3 method for class 'freqtab'
as.data.frame(x, row.names = NULL, optional = FALSE, drop = FALSE, ...)

## S3 method for class 'freqtab'
head(x, ...)

## S3 method for class 'freqtab'
tail(x, ...)

scales(x, margin = 1)

margin(x, margin = 1)

margins(x)

## S3 method for class 'freqtab'
droplevels(x, ...)
```

Arguments

x	either an object (vector or data.frame) containing total scores or item responses with which total scores will be calculated, or an object inheriting from class "freqtab". In the freqtab function, the first column in x must be the total test; any remaining columns may contain anchor scores. See below for details.
...	further arguments passed to or from other functions.
scales	list of vectors containing the score scales for each score scale in x.
items	list of vectors of column indices (numbers or column names) with which total scores will be computed for x when it contains item responses.
design	the equating design used in data collection. For univariate x, design = "eg" is assumed for equivalent groups. For multivariate x, design = "ng" is assumed for nonequivalent groups. Single-groups and counterbalanced designs must be specified with design = "sg" and design = "cb".
na.rm	logical with default TRUE specifying whether or not missing item responses should be ignored, i.e., treated as 0, when calculating total scores.
drop	logical, with default FALSE, indicating whether or not unused factor levels, or score values with zero counts, should be dropped. See below for details.

row.names, optional
: arguments passed to as.data.frame, currently ignored.

margin
: integer vector specifying the margin(s) over which frequencies should be summed.

Details

freqtab creates a frequency table from a vector or data.frame of scores. When the items argument is included, scores are assumed to be item responses, which are summed to create total scores. The scores are tabulated and stored as an array, with dimensions for each variable. Note that in previous versions of the "freqtab" class the frequency table was stored as a data.frame. This is no longer the case. Instead, the table is stored as an array and converted to a data.frame when printed or manipulated with the head and tail methods.

as.data.frame converts an object of class "freqtab" to "data.frame". droplevels returns x with any unused factor levels, or levels with zero counts, removed.

When x is an object of class "table", freqtab simply modifies the attributes and converts to class "freqtab". In this case, x must already be structured similar to a "freqtab" object, with the first dimension containing counts for total scores, and remaining dimensions containing counts for one or more anchor tests.

as.freqtab converts a 'flat' contingency table (see ftable) to class "freqtab" with the appropriate attributes. A flat contingency table is the data.frame version of a "freqtab" object, where the first column contains the total score scale, the last column contains counts, and the columns in between contain different anchor test score combinations. is.freqtab tests for class "freqtab".

scales extracts the measurement scales for the variables specified in margin, with margin = 1 referring to the total score scale, and subsequent margins referring to anchor tests. margin is a wrapper for margin.table, which itself is a simple wrapper for summing over marginal counts, i.e., apply(x, margin, sum). And margins returns the number of dimensions, i.e., score variables, in a frequency table.

design is used to set the dimnames of the frequency table, with total1 and total2 used with single and counterbalanced groups, and total and anchor(s) used otherwise. design also sets the design attribute, which is used in equate.

The main difference between the "freqtab" class and other tabulation classes, like "table" and "ftable", is that the dimnames, i.e., the score scales, are required to be numeric. This facilitates plotting with plot.freqtab, equating with the equate function, and descriptive statistics with the summary.freqtab and other methods.

Value

A table array with dimensions equal to the number of score scales. In most cases, this will be a univariate or bivariate distribution, but multivariate distributions are supported. scales and margins return numeric vectors.

Methods (by class)

- freqtab(default): Default method for a data.frame of item responses, a data.frame of total and anchor scores, or a vector of total scores.
- freqtab(table): Method for tables.

Author(s)

Anthony Albano <tony.d.albano@gmail.com>

See Also

table, ftable, summary.freqtab, plot.freqtab

Examples

```
# Univariate distribution with score scale
set.seed(2005)
x <- round(rnorm(1000, 100, 10))
head(freqtab(x, scales = 70:130))

# Existing frequency table converted to class "freqtab"
# The first score of zero, with zero counts, is dropped
head(as.freqtab(ACTmath[, 1:2], drop = TRUE))

# Bivariate distribution
# Reduce y to the anchor test margin (2)
ny <- freqtab(x = KBneat$y, scales = list(0:36, 0:12))
margin(ny, margin = 2)

# Summing scored item responses with the PISA data
attach(PISA)
r6items <- paste(items$itemid[items$clusterid == "r6"])
r7items <- paste(items$itemid[items$clusterid == "r7"])
pisa67 <- freqtab(students[students$book == 6, ],
items = list(r6items, r7items),
scales = list(0:16, 0:14))
detach(PISA)

# Scales for both margins
# Zero total score is unobserved
scales(pisa67, 1:2)
scales(droplevels(pisa67), 1:2)
```

KBneat *Test Scores under a NEAT design*

Description

This dataset contains scores for two forms of a 36-item test administered under a nonequivalent groups with anchor test design. The anchor test is internal and consists of twelve items taken by both groups of examinees. The data is distributed as part of the equating software CIPE (see link below), and additional examples using this dataset are presented in *Test Equating, Scaling, and Linking* (Kolen and Brennan, 2004).

Usage

```
KBneat
```

Format

A list of length two, where each element (`$x` and `$y`) contains two columns of data, one for the total score and another for the anchor test score for each examinee.

Source

Kolen, M. J., & Brennan, R. L. (2004). *Test Equating, Scaling, and Linking*. (2nd ed.), New York: Springer.

The dataset can be downloaded as part of the CIPE software, available at the following link: https://education.uiowa.edu/casma

---

PISA *Programme for International Student Assessment 2009 USA Data*

Description

This dataset contains scored cognitive item response data from the 2009 administration of the Programme for International Student Assessment (PISA), an international study of education systems. The data, and license under which they are released, are available online at https://www.oecd.org.

Usage

```
PISA
```

Format

`PISA` is a `list` containing four elements. The first, `PISA$students`, is a `data.frame` containing 233 variables across 5233 individuals, with one row per individual. All but one variable come from the USA PISA data file "INT_COG09_S_DEC11.txt". The remaining variable, language spoken at home, has been merged in from the student questionnaire file "INT_STQ09_DEC11.txt". Variable names match those found in the original files:

list("stidstd") Unique student ID (one for each of the 5233 cases);

list("schoolid") School ID (there are 165 different schools);

list("bookid") ID for the test booklet given to a particular student, of which there were 13;

list("langn") Student-reported language spoken at home, with 4466 students reporting English (indicated by code 313), 484 students reporting Spanish (with code 156) and 185 students reporting "another language" (code 859);

list("m033q01") Scored item-response data across the 189 items included in the general cognitive assessment, described below; and

to Scored item-response data across the 189 items included in the general cognitive assessment, described below; and

list("s527q04t") Scored item-response data across the 189 items included in the general cognitive assessment, described below; and

list("pv1math") PISA scale scores, referred to in the PISA technical documentation as "plausible values".

to PISA scale scores, referred to in the PISA technical documentation as "plausible values".

list("pv5read5") PISA scale scores, referred to in the PISA technical documentation as "plausible values".

Next, `PISA$booklets` is a `data.frame` containing 4 columns and 756 rows and describes the 13 general cognitive assessment booklets. Variables include:

list("bookid") The test booklet ID, as in `PISA$students`;

list("clusterid") ID for the cluster or item subset in which an item was placed; items were fully nested within clusters; however, each item cluster appeared in four different test booklets;

list("itemid") Item ID, matching the columns of `PISA$students`; each item appears in `PISA$booklets` four times, once for each booklet; and

list("order") The order in which the cluster was presented within a given booklet.

`PISA$items` is a `data.frame` containing 4 columns and 189 rows, with one row per item. Variables include:

list("itemid") Item ID, as in `PISA$booklets`

list("clusterid") Cluster ID, as in `PISA$booklets`

list("max") Maximum possible score value, either 1 or 2 points, with dichotomous scoring (max of 1) used for the majority of items; and

list("subject") The subject of an item, equivalent to the first character in `itemid` and `clusterid`.

list("format") Item format, abbreviated as `mc` for multiple choice, `cmc` for complex multiple choice, `ocr` for open constructed response, and `ccr` for closed constructed response.

list("noptions") Number of options, zero except for some multiple choice items.

Finally, `PISA$totals` is a list of 13 `data.frames`, one per booklet, where the columns correspond to total scores for all students on each cluster for the corresponding booklet. These total scores were calculated using `PISA$students` and `PISA$booklets`. Elements within the `PISA$totals` list are named by booklet, and the columns in the `data.frame` are named by cluster. For example, `PISA$totals$b1$m1` contains the total scores on cluster M1 for students taking booklet 1.

Source

OECD (2012). PISA 2009 Technical Report, PISA, OECD Publishing. http://dx.doi.org/10.1787/9789264167872-en

plot.bootstrap *Plotting Bootstrap Equating Results*

Description

This function plots bootstrap equating results for objects of class "bootstrap".

Usage

```
## S3 method for class 'bootstrap'
plot(
  x,
  add = FALSE,
  out = "mean",
  xpoints,
  ypoints,
  addident = TRUE,
  identy,
  identcol = 1,
  rescale = c(0, 1),
  xlab = "Total Score",
  ylab,
  col = rainbow(length(x$args)),
  pch,
  lty = 1,
  subset,
  morepars = NULL,
  addlegend = TRUE,
  legendtext,
  legendplace = "bottomright",
  ...
)
```

Arguments

x	output from the bootstrap function.
add	logical, with default FALSE, specifying whether to create a new plot or add to the current one.
out	character vector specifying the output to be plotted, either the mean equated scores ("mean"), standard errors ("se"), bias ("bias"), or RMSE ("rmse").
xpoints, ypoints	optional vectors of the same length containing raw scores on forms X and Y, assuming a single group or equivalent groups design.
addident	logical, with default TRUE, for plotting the identity function. The result depends on out.

identy	vector of y coordinates for plotting the identity line. Defaults to the identity function when out = "eqs", otherwise, a horizontal line with intercept 0.
identcol	color used for plotting the identity line.
rescale	intercept and slope, with default 0 and 1, used to rescale all lines before plotting.
xlab, ylab, col, pch, lty	graphical parameters passed to par, with the lengths of col and lty recycled as necessary.
subset	vector for subsetting the output when multiple equating functions are included in x.
morepars	list of additional graphical parameters, excluding xlab, ylab, col, pch, lty.
addlegend	logical, with default TRUE, indicating whether or not a legend should be added.
legendtext	character vector of text to be passed to the legend argument of the legend function, defaulting to a combination of the equating types and methods specified in each equating object.
legendplace	placement of the legend.
...	further arguments passed to or from other methods, excluding graphical parameters.

Details

Lines are plotted for the chosen output type, whether mean equated scores across replications (out = "mean"), standard errors (out = "se"), bias (out = "bias") or RMSE (out = "rmse"). The result is similar to that of plot.equate.

Author(s)

Anthony Albano <tony.d.albano@gmail.com>

See Also

bootstrap, plot.equate

Examples

```
set.seed(122713)
neat.x <- freqtab(KBneat$x, scales = list(0:36, 0:12))
neat.y <- freqtab(KBneat$y, scales = list(0:36, 0:12))
eqargs <- list(m.t = list(type = "mean", method = "t"),
  l.t = list(type = "lin", method = "t"),
c.t = list(type = "circ", method = "t"))
bootout <- bootstrap(x = neat.x, y = neat.y, args = eqargs,
reps = 20)
plot(bootout, out = "se", legendplace = "top")
```

plot.equate *Plotting Equating Results*

Description

Functions for plotting equating functions from one or more objects of class "equate" or "equate.list".

Usage

```
## S3 method for class 'equate'
plot(
  ...,
  elist = NULL,
  add = FALSE,
  out = "eqs",
  xpoints,
  ypoints,
  addident = TRUE,
  identy,
  identcol = 1,
  rescale = c(0, 1),
  xlab = "Total Score",
  ylab,
  col = rainbow(length(x)),
  pch,
  lty = 1,
  lwd = 1,
  subset,
  morepars = NULL,
  addlegend = TRUE,
  legendtext,
  legendplace = "bottomright"
)

## S3 method for class 'equate.list'
plot(x, ...)
```

Arguments

...	one or more equating objects, each containing results for equating the same two test forms.
elist	list of equatings to be plotted.
add	logical, with default FALSE, specifying whether to create a new plot or add to the current one.
out	character vector specifying the output to be plotted, either equating functions ("eqs"), standard errors ("se"), bias ("bias"), or RMSE ("rmse").

`xpoints`, `ypoints`
: optional vectors of the same length containing raw scores on forms X and Y, assuming a single group or equivalent groups design.

`addident`
: logical, with default `TRUE`, for plotting the identity function. The result depends on `out`.

`identy`
: vector of y coordinates for plotting the identity line. Defaults to the X scale when `out = "eqs"`, otherwise, a horizontal line with intercept 0.

`identcol`
: color used for plotting the identity line.

`rescale`
: intercept and slope, with default 0 and 1, used to rescale all lines before plotting.

`xlab`, `ylab`, `col`, `pch`, `lty`, `lwd`
: graphical parameters passed to `par`, with `col`, `pch`, `lty`, and `lwd` recycled as necessary.

`subset`
: vector for subsetting the output when multiple equating functions are included in `x`.

`morepars`
: list of additional graphical parameters, excluding `xlab`, `ylab`, `col`, `pch`, `lty`, and `lwd`.

`addlegend`
: logical, with default `TRUE`, indicating whether or not a legend should be added.

`legendtext`
: character vector of text to be passed to the `legend` argument of the `legend` function, defaulting to a combination of the equating types and methods specified in each equating object.

`legendplace`
: placement of the legend.

`x`
: "`equate.list`" object, containing output from multiple equatings.

Details

Equating functions (`out = "eqs"`) are plotted as lines based on the concordance table for each equating object that is supplied. Standard errors (`out = "se"`) default to bootstrap standard errors, if available, otherwise, analytical standard errors are plotted. Bias (`out = "bias"`) and RMSE (`out = "rmse"`) are also taken from bootstrapping output.

Examples

```
# See ?equate for additional examples

rx <- as.freqtab(ACTmath[, 1:2])
ry <- as.freqtab(ACTmath[, c(1, 3)])
set.seed(2007)

req1 <- equate(rx, ry, type = "i", boot = TRUE, reps = 5)
req2 <- equate(rx, ry, type = "m", boot = TRUE, reps = 5)
req3 <- equate(rx, ry, type = "l", boot = TRUE, reps = 5)
req4 <- equate(rx, ry, type = "e", boot = TRUE, reps = 5,
  smooth = "loglin", degree = 3)
req5 <- composite(list(req1, req2), wc = .5, symmetric = TRUE)

plot(req1, req2, req3, req4, req5[[1]], addident = FALSE)
plot(req5)
```

plot.freqtab *Plotting Frequency Distributions*

Description

This function plots univariate and bivariate frequency tables of class "freqtab".

Usage

```
## S3 method for class 'freqtab'
plot(
  x,
  y = NULL,
  xcol = 1,
  ycol,
  pch = 16,
  ylty = 1,
  xlab = names(dimnames(x))[1],
  addlegend = !missing(y),
  legendtext,
  ...
)

## S3 method for class 'freqtab'
points(x, xcol = 1, pch = 16, ds = 50, dm = 100, ...)
```

Arguments

x	univariate or bivariate score distribution of class "freqtab".
y	either an object of class "freqtab", where frequencies will be extracted, or a vector or matrix of frequencies, to be added to the plot of x. See below for details.
xcol, ycol	colors used in plotting x and y.
pch	plotting symbol used to plot bivariate points.
ylty	line type used to plot frequencies in y.
xlab	label for the x axis.
addlegend	logical indicating whether or not a legend should be added.
legendtext	character vector of text to be passed to the legend argument of the legend function, defaulting to column names used in y.
...	further arguments passed to or from other methods, such as graphical parameters besides col, type, and pch.
ds, dm	integers for the scaling and center of the RGB density values, with defaults of 50 and 100. These are used to convert the observed counts in x to the [0, 255] range of RGB values.

Details

For the points method, a scatterplot for x is added to the current opened plot.

For the plot method, when x is univariate, i.e, having 2 columns, a frequency plot is created for x. When x is bivariate, e.g., coming from a single group equating design or one form of a nonequivalent groups design, a scatterplot is produced with frequency plots for the marginal distributions.

y is used to superimpose lines, e.g., smoothed frequencies, over the (marginal) frequencies of x.

Colors must be specified using xcol and ycol. When ycol is missing, a vector of colors is created using rainbow(ncol(y)).

Value

The univariate option produces a single line plot of type = "h". Frequencies from y are then superimposed. The bivariate option produces a scatterplot with a marginal frequency plot for each distribution.

Author(s)

Anthony Albano <tony.d.albano@gmail.com>

See Also

plot.table, plot.equate, lines, points

Examples

```
x <- freqtab(KBneat$x, scales = list(0:36, 0:12))
plot(x)

xs <- loglinear(x, degrees = c(4, 1),
  stepup = TRUE, showWarnings = FALSE)
plot(x, xs, lwd = 2)
```

---

presmoothing *Frequency Distribution Presmoothing*

Description

These functions are used to smooth frequency distributions.

Usage

```
presmoothing(x, ...)

## Default S3 method:
presmoothing(
  x,
  smoothmethod = c("none", "average", "bump", "loglinear"),
  jmin,
  asfreqtab = TRUE,
  ...
)

## S3 method for class 'formula'
presmoothing(x, data, ...)

loglinear(
  x,
  scorefun,
  degrees = list(4, 2, 2),
  grid,
  rmimpossible,
  asfreqtab = TRUE,
  models,
  stepup = !missing(models),
  compare = FALSE,
  choose = FALSE,
  choosemethod = c("chi", "aic", "bic"),
  chip,
  verbose = FALSE,
  ...
)

freqbump(x, jmin = 1e-06, asfreqtab = FALSE, ...)

freqavg(x, jmin = 1, asfreqtab = FALSE, ...)
```

Arguments

x	either an object of class "freqtab" specifying a univariate or multivariate score distribution, or a "formula" object.
...	further arguments passed to other methods. For presmoothing, these are passed to loglinear and include those listed above.
smoothmethod	character string indicating the smoothing method to be used by presmoothing. "none" returns unsmoothed frequencies, "bump" adds a small frequency to each score value, "average" imputes small frequencies with average values, and "loglinear" fits loglinear models. See below for details.
jmin	for smoothmethod = "average", the minimum frequency, as an integer, below

	which frequencies will be replaced (default is 1). for `smoothmethod = "bump"`, the value to be added to each score point (as a probability, with default 1e-6).
`asfreqtab`	logical, with default `TRUE`, indicating whether or not a frequency table should be returned. For `smoothmethod = "average"` and `smoothmethod = "bump"`, the alternative is a vector of frequencies. For `loglinear`, there are other options.
`data`	an object of class "`freqtab`".
`scorefun`	matrix of score functions used in loglinear presmoothing, where each column includes a transformation of the score scale or interactions between score scales. If missing, `degrees` and `xdegree` will be used to construct polynomial score functions.
`degrees`	list of integer vectors, each one indicating the maximum polynomial score transformations to be computed for each variable at a given order of interactions. Defaults (`degrees = list(4, 2, 2)`) are provided for up to trivariate interactions. `degrees` are ignored if `scorefun` or `grid` are provided. See below for details.
`grid`	matrix with one column per margin in `x` and one row per term in the model. See below for details.
`rmimpossible`	integer vector indicating columns in `x` to be used in removing impossible scores before smoothing, assuming internal anchor variables. Impossible scores are kept by default. See below.
`models`	integer vector indicating which model terms should be grouped together when fitting multiple nested models. E.g., `models = c(1, 1,2, 3)` will compare three models, with the first two terms in model one, the third term added in model two, and the fourth in model three.
`stepup`	logical, with default `FALSE`, indicating whether or not multiple nested models should be automatically fit. If `TRUE` and `models` is missing, an attempt will be made to create it using `grid` and/or `degrees`. Otherwise, in the absence of `models`, each column in `scorefun` will define a new sequential model.
`compare`	logical, with default `FALSE`, indicating whether or not fit for nested models should be compared. If `TRUE`, `stepup` is also set to `TRUE` and only results from the model fit comparison are returned, that is, `verbose` is ignored.
`choose`	logical, with default `FALSE`, indicating whether or not the best-fitting model should be returned after comparing fit of nested models. Useful for automating model selection in simulations.
`choosemethod`	string, indicating the method for selecting a best-fitting model when `choose = TRUE`. `"chi"` selects the most complex model with chi-square p-value below the criterion in `chip`. Remaining methods choose the model with lowest value.
`chip`	proportion specifying the type-I error rate for model selection based on `choosemethod = "chi"`.
`verbose`	logical, with default `FALSE`, indicating whether or not full `glm` output should be returned.

Details

Loglinear smoothing is a flexible procedure for reducing irregularities in a frequency distribution prior to equating, where the degree of each polynomial term determines the specific moment of

the observed distribution that is preserved in the fitted distribution (see below for examples). The `loglinear` function is a wrapper for `glm`, and is used to simplify the creation of polynomial score functions and the fitting and comparing of multiple loglinear models.

`scorefun`, if supplied, must contain at least one score function of the scale score values. Specifying a list to `degrees` is an alternative to supplying `scorefun`. Each list element in `degrees` should be a vector equal in length to the number of variables contained in `x`; there should also be one such vector for each possible level of interaction between the variables in `x`.

For example, the default `degrees = list(4, 2, 2)` is recycled to produce `list(c(4, 4, 4), c(2, 2, 2), c(2, 2, 2))`, resulting in polynomials to the fourth power for each univariate distribution, to the second power for each two-way interaction, and to the second power for the three-way interaction.

Terms can also be specified with `grid`, which is a matrix with each row containing integers specifying the powers for each variable at each interaction term, including main effects. For example, the main effect to the first power for the total score in a bivariate distribution would be `c(1, 0)`; the interaction to the second power would be `c(2, 2)`.

`stepup` is used to run nested models based on subsets of the columns in `scorefun`. Output will correspond to models based on columns 1 and 2, 1 through 3, 1 through 4, to 1 through `ncol(scorefun)`. This list of polynomial terms is then used to create a `grid` using `expand.grid`. The `grid` can also be supplied directly, in which case `degrees` will be ignored.

`compare` returns output as an `anova` table, comparing model fit for all the models run with `stepup = TRUE`, or by specifying more than one model in `models`. When `choose = TRUE`, the arguments `choosemethod` and `chip` are used to automatically select the best-fitting model based on the `anova` table from running `compare`.

The remaining smoothing methods make adjustments to scores with low or zero frequencies. `smoothmethod = "bump"` adds the proportion `jmin` to each score point and then adjusts the probabilities to sum to 1. `smoothmethod = "average"` replaces frequencies falling below the minimum `jmin` with averages of adjacent values.

Value

When `smoothmethod = "average"` or `smoothmethod = "bump"`, either a smoothed frequency vector or table is returned. Otherwise, `loglinear` returns the following:

- when `compare = TRUE`, an anova table for model fit
- when `asfreqtab = TRUE`, a smoothed frequency table
- when `choose = TRUE`, a smoothed frequency table with attribute "anova" containing the model fit table for all models compared
- when `verbose = TRUE`, full `glm` output, for all nested models when `stepup = TRUE`
- when `stepup = TRUE` and `verbose = FALSE`, a `data.frame` of fitted frequencies, with one column per model

Methods (by class)

- `presmoothing(default)`: Default method for frequency tables.
- `presmoothing(formula)`: Method for "`formula`" objects.

Author(s)

Anthony Albano <tony.d.albano@gmail.com>

References

Holland, P. W., and Thayer, D. T. (1987). *Notes on the use of log-linear models for fitting discrete probability distributions* (PSR Technical Rep. No. 87-79; ETS RR-87-31). Princeton, NJ: ETS.

Holland, P. W., and Thayer, D. T. (2000). Univariate and bivariate loglinear models for discrete test score distributions. *Journal of Educational and Behavioral Statistics, 25*, 133–183.

Moses, T., and Holland, P. W. (2008). *Notes on a general framework for observed score equating* (ETS Research Rep. No. RR-08-59). Princeton, NJ: ETS.

Moses, T., and Holland, P. W. (2009). Selection strategies for univariate loglinear smoothing models and their effect on equating function accuracy. *Journal of Educational Measurement, 46*, 159–176. ETS.

Wang, T. (2009). Standard errors of equating for the percentile rank-based equipercentile equating with log-linear presmoothing. *Journal of Educational and Behavioral Statistics, 34*, 7–23.

See Also

glm, loglin

Examples

```
set.seed(2010)
x <- round(rnorm(1000, 100, 15))
xscale <- 50:150
xtab <- freqtab(x, scales = xscale)

# Adjust frequencies
plot(xtab, y = cbind(average = freqavg(xtab),
  bump = freqbump(xtab)))

# Smooth x up to 8 degrees and choose best fitting model
# based on aic minimization
xlog1 <- loglinear(xtab, degrees = 8,
  choose = TRUE, choosemethod = "aic")
plot(xtab, as.data.frame(xlog1)[, 2],
  legendtext = "degree = 3")

# Add "teeth" and "gaps" to x
# Smooth with formula interface
teeth <- c(.5, rep(c(1, 1, 1, 1, .5), 20))
xttab <- as.freqtab(cbind(xscale, c(xtab) * teeth))
xlog2 <- presmoothing(~ poly(total, 3, raw = TRUE),
  xttab, showWarnings = FALSE)

# Smooth xt using score functions that preserve
# the teeth structure (also 3 moments)
teeth2 <- c(1, rep(c(0, 0, 0, 0, 1), 20))
xt.fun <- cbind(xscale, xscale^2, xscale^3)
```

```
xt.fun <- cbind(xt.fun, teeth2, xt.fun * teeth2)
xlog3 <- loglinear(xttab, xt.fun, showWarnings = FALSE)

# Plot to compare teeth versus no teeth
op <- par(no.readonly = TRUE)
par(mfrow = c(3, 1))
plot(xttab, main = "unsmoothed", ylim = c(0, 30))
plot(xlog2, main = "ignoring teeth", ylim = c(0, 30))
plot(xlog3, main = "preserving teeth", ylim = c(0, 30))
par(op)

# Bivariate example, preserving first 3 moments of total
# and anchor for x and y, and the covariance
# between anchor and total
# see equated scores in Wang (2009), Table 4
xvtab <- freqtab(KBneat$x, scales = list(0:36, 0:12))
yvtab <- freqtab(KBneat$y, scales = list(0:36, 0:12))
Y <- as.data.frame(yvtab)[, 1]
V <- as.data.frame(yvtab)[, 2]
scorefun <- cbind(Y, Y^2, Y^3, V, V^2, V^3, V*Y)
wang09 <- equate(xvtab, yvtab, type = "equip",
  method = "chained", smooth = "loglin",
  scorefun = scorefun)
wang09$concordance

# Removing impossible scores has essentially no impact
xvlog1 <- loglinear(xvtab, scorefun, asfreqtab = FALSE)
xvlog2 <- loglinear(xvtab, scorefun, rmimpossible = 1:2)
plot(xvtab, cbind(xvlog1,
xvlog2 = as.data.frame(xvlog2)[, 3]))
```

---

px *Percentile Ranks and Cumulative Frequencies*

Description

These functions compute percentile ranks and cumulative frequency distributions for frequency tables.

Usage

```
px(x, ...)

## Default S3 method:
px(x, y, ys, ...)

## S3 method for class 'freqtab'
px(x, margin = 1, y, ymargin = 1, ...)
```

```
fx(x, ...)

## Default S3 method:
fx(x, ...)

## S3 method for class 'freqtab'
fx(x, margin = 1, ...)
```

Arguments

x	either a vector of counts, or an object of class "freqtab" from which counts will be taken.
...	further arguments passed to or from other methods.
y	an object of class "freqtab" when x is as well, otherwise, a vector or data.frame of counts. See below for details.
ys	vector specifying the y score scale, when it is not contained in the first column of y. If y can be converted to a data.frame, it is assumed to be univariate with the first column containing the score scale and the second containing the counts.
margin, ymargin	integers specifying the margins for which frequencies or percentile ranks will be returned. margin applies to x and ymargin to y.

Details

These functions compute percentile ranks and cumulative frequencies for a univariate distribution, and percentile ranks from one univariate distribution (x) corresponding to score values in another (y).

Value

A vector is returned containing either percentile ranks or cumulative frequencies with length equal to length(x).

Author(s)

Anthony Albano <tony.d.albano@gmail.com>

See Also

freqtab

Examples

```
x <- as.freqtab(ACTmath[, 1:2], drop = TRUE)
y <- as.freqtab(ACTmath[, c(1, 3)], drop = TRUE)

# Percentile ranks for the x scale
round(px(x), 3)
```

```
# Percentile ranks in y for x each score
round(px(x, y = y), 3)

# Cumulative frequency distribution for x
round(fx(x), 3)
```

`sample.freqtab` *Bootstrap Random Sampling from Frequency Tables*

Description

An extension of `sample` to objects of class "`freqtab`" for bootstrap sampling.

Usage

```
sample.freqtab(x, size = sum(x), replace = TRUE)
```

Arguments

`x`	object of class "`freqtab`", which is an array of counts across one or more numeric dimensions.
`size`	non-negative integer giving the sample size.
`replace`	logical with default `TRUE` indicating whether sampling should be with replacement.

Value

A table array, as a "`freqtab`" object, sampled from the original `x`.

Author(s)

Anthony Albano <tony.d.albano@gmail.com>

See Also

`table`, `ftable`, `summary.freqtab`, `plot.freqtab`

Examples

```
# Sample with replacement from ACT math and compare results
set.seed(2021)
rx <- as.freqtab(ACTmath[, 1:2])
rxs <- sample.freqtab(rx)
summary(rx)
summary(rxs)
```

summary.freqtab *Descriptive Statistics for Frequency Tables*

Description

These functions return descriptive statistics for a frequency table of class "freqtab".

Usage

```
## S3 method for class 'freqtab'
summary(object, margin = seq(margins(object)), ...)

## S3 method for class 'freqtab'
mean(x, margin = 1, ...)

sd.freqtab(x, margin = 1)

var.freqtab(x, margin = 1)

cov.freqtab(x, margin = seq(margins(x)))

cor.freqtab(x, margin = seq(margins(x)))

## S3 method for class 'freqtab'
min(x, margin = 1, ..., na.rm = FALSE)

## S3 method for class 'freqtab'
max(x, margin = 1, ..., na.rm = FALSE)

## S3 method for class 'freqtab'
range(x, margin = 1, ..., na.rm = FALSE)

skew.freqtab(x, margin = 1)

kurt.freqtab(x, margin = 1)
```

Arguments

object, x	object of class "freqtab".
margin	integer vector specifying the margin(s) for which summary statistics will be returned. This defaults to 1 for univariate statistics, and seq(margins(x)), i.e., all the margins, for multivariate statistics (covariance and correlation).
...	further arguments passed to or from other methods.
na.rm	logical indicating whether missing values should be removed, currently ignored since frequency tables cannot contain missing values.

Details

mean, sd.freqtab, var.freqtab, skew.freqtab, and kurt.freqtab return the mean, standard deviation, variance, skewness, and kurtosis. min and max return the minimum and maximum observed scores, and range returns both. cov.freqtab and cor.freqtab return the covariance and correlation matrices for one or more variables. summary returns univariate statistics across one or more margins.

Value

summary returns a data frame of summary statistics, including the mean, standard deviation, skewness, kurtosis, minimum, maximum, and number of observations for each variable in margin. Otherwise, a vector of length length(margin) is returned with the corresponding statistic for each variable.

Author(s)

Anthony Albano <tony.d.albano@gmail.com>

See Also

freqtab

Examples

```
summary(as.freqtab(ACTmath[, 1:2]))

ny <- freqtab(KBneat$y, scales = list(0:36, 0:12))
summary(ny)
cov.freqtab(ny)
```

Index