Package 'weights'

February 12, 2020

Title Weighting and Weighted Statistics

Version 1.0.1

Date 2020-02-10

Author Josh Pasek [aut, cre], with some assistance from Alex Tahk and some code modified from R-core; Additional contributions by Gene Culter and Marcus Schwemmle.

Maintainer Josh Pasek <josh@joshpasek.com>

Depends Hmisc, gdata, mice

Description Provides a variety of functions for producing simple weighted statistics, such as weighted Pearson's correlations, partial correlations, Chi-Squared statistics, histograms, and t-tests. Also now includes some software for quickly recoding survey data and plotting point estimates from interaction terms in regressions (and multiply imputed regressions). NOTE: Weighted partial correlation calculations pulled to address a bug.

License GPL (>= 2)

LazyLoad yes

NeedsCompilation yes

Repository CRAN

Date/Publication 2020-02-12 15:20:02 UTC

R topics documented:

---

anes04 *Demographic Data From 2004 American National Election Studies (ANES)*

Description

A dataset containing demographic data from the 2004 American National Election Studies. The data include 5 variables: "female" (A Logical Variable Indicating Sex), "age" (Numerically Coded, Ranging From 18 to a Topcode of 90), "educats" (5 Education Categories corresponding to 1-Less than A High School Degree, 2-High School Gradutate, 3-Some College, 4-College Graduate, 5-Post College Education), "racecats" (6 Racial Categories), and "married" (A Logical Variable Indicating the Respondent's Marital Status, with one point of missing data). Dataset is designed show how production of survey weights works in practice.

Usage

```
data(anes04)
```

Format

The format is: chr "anes04"

Source

http://www.electionstudies.org

---

dummify *Separate a factor into separate dummy variables for each level.*

Description

`dummify` creates a matrix with columns signifying separate dummy variables for each level of a factor. The column names are the former levels of the factor.

Usage

```
dummify(x, show.na=FALSE, keep.na=FALSE)
```

Arguments

`x`	`x` is a factor the researcher desires to split into separate dummy variables.
`show.na`	If `show.na` is 'TRUE', output will include a column idicating the cases that are missing.
`keep.na`	If `keep.na` is 'TRUE', output vectors will have "NA"s for cases that were originally missing.

Value

dummify returns a matrix with a number of rows equal to the length of x and a number of columns equal to the number of levels of x.

Author(s)

Josh Pasek, Assistant Professor of Communication Studies at the University of Michigan (www.joshpasek.com).

Examples

```
data("anes04")

anes04$agecats <- cut(anes04$age, c(17, 25,35,45,55,65, 99))
levels(anes04$agecats) <- c("age1824", "age2534", "age3544",
          "age4554", "age5564", "age6599")

agedums <- dummify(anes04$agecats)
table(anes04$agecats)
summary(agedums)
```

---

nalevs *Recode variables to 0-1 scale*

Description

nalevs takes as an input any vector and recodes it to range from 0 to 1, to treat specified levels as missing, to treat specified levels as 0, 1, .5, or the mean (weighted or unweighted) of the levels present after coding.

Usage

```
nalevs(x, naset=NULL, setmid=NULL, set1=NULL, set0=NULL,
setmean=NULL, weight=NULL)
```

Arguments

x	A vector to be recoded to range from 0 to 1.
naset	A vector of values of x to be coded as NA.
setmid	A vector of values of x to be recoded to .5.
set1	A vector of values of x to be recoded to 1.
set0	A vector of values of x to be recoded to 0.
setmean	A vector of values of x to be recoded to the mean (if no weight is specified) or weighted mean (if a weight is specified) of values of x after all recoding.
weight	A vector of weights for x if weighted means are desired for values listed for setmean.

Value

A vector of length equal to that of x of class numeric.

Author(s)

Josh Pasek, Assistant Professor of Communication Studies at the University of Michigan (www.joshpasek.com).

Examples

```
data(anes04)
summary(anes04$age)
summary(nalevs(anes04$age))
table(anes04$educcats)
table(nalevs(anes04$educcats, naset=c(2, 4)))
```

---

plotwtdinteraction *Functions to Identify and Plot Predicted Probabilities As Well As Two- and Three-Way Interactions From Regressions With or Without Weights and Standard Errors*

Description

plotwtdinteraction produces a plot from a regression object to illustrate a two- or three-way interaction for a prototypical individual holding constant all other variables. Prototypical individual is identified as the mean (numeric), median (ordinal), and/or modal (factors and logical variables) values for all measures. Standard errors are illustrated with polygons by default.

findwtdinteraction generates a table of point estimates from a regression object to illustrate a two- or three-way interaction for a prototypical individual holding constant all other variables. Prototypical individual is identified as the mean (numeric), median (ordinal), and/or modal (factors and logical variables) values for all measures. Standard errors are illustrated with polygons by default.

plotinteractpreds plots an object from findwtdinteraction.

These functions are known to be compatible with lm, glm, as well as multiply imputed lm and glm data generated with the mice package.

Note, this set of functions is still in beta, please let me know if you run into any bugs when using it.

Important: If you are using a regression output from a multiply imputed dataset with a continuous variable as an interacting term, you should always specify the levels (acrosslevs, bylevs, or atlevs) for the variable, as imputations can change the set of levels that are available and thus can make the point estimates across imputed datasets incompatible with one-another.

ordinal regressions (polr) and multinomial regressions (multinom) do not currently support standard errors

Usage

```
plotwtdinteraction(x, across, by=NULL, at=NULL, acrosslevs=NULL, bylevs=NULL,
atlevs=NULL, weight=NULL, dvname=NULL, acclevnames=NULL, bylevnames=NULL,
atlevnames=NULL, stdzacross=FALSE, stdzby=FALSE, stdzat=FALSE, limitlevs=20,
type="response", seplot=TRUE, ylim=NULL, main=NULL, xlab=NULL, ylab=NULL,
legend=TRUE, placement="bottomright", lwd=3, add=FALSE, addat=FALSE, addby=TRUE,
mfrow=NULL, linecol=NULL, secol=NULL, showbynamelegend=FALSE, showatnamelegend=FALSE,
lty=NULL, density=30, startangle=45, approach="prototypical", data=NULL, ...)

findwtdinteraction(x, across, by=NULL, at=NULL, acrosslevs=NULL, bylevs=NULL,
atlevs=NULL, weight=NULL, dvname=NULL, acclevnames=NULL, bylevnames=NULL,
atlevnames=NULL, stdzacross=FALSE, stdzby=FALSE, stdzat=FALSE, limitlevs=20,
type="response", approach="prototypical", data=NULL)

plotinteractpreds(out, seplot=TRUE, ylim=NULL, main=NULL, xlab=NULL, ylab=NULL,
legend=TRUE, placement="bottomright", lwd=3, add=FALSE, addat=FALSE, mfrow=NULL,
linecol=NULL, secol=NULL, showbynamelegend=FALSE, showatnamelegend=FALSE, lty=NULL,
density=30, startangle=45, ...)
```

Arguments

x	x is a regression object in lm, glm, or mira (multiply imputed) format that includes the variables to be plotted.
out	out is an object estimate using findwtdinteraction that should be plotted.
across	across specifies the name of the variable, in quotation marks, that was used in the regression that should be plotted on the X axis.
by	by specifies the name of the variable, in quotation marks, that was used in the regression that should form each of the separate lines in the regression.
at	at (optional) specifies the name of the variable, in quotation marks, that represents the third-way of a 3-way interaction. Depending on specifications, this can either be plotted as additional lines or as separate graphs.
acrosslevs	acrosslevs (optional) specifies the unique levels of the variable across that should be estimated across the x axis. If this is not specified, each unique level of the across variable will be used.
bylevs	bylevs (optional) specifies the unique levels of the variable by that should yield separate lines. If this is not specified, each unique level of the by variable will be used.
atlevs	atlevs (optional) specifies the unique levels of the variable at that should yield separate figures or lines. If this is not specified, each unique level of the at variable will be used.
weight	weight (optional) allows the user to introduce a separate weight that was not used in the original regression. If the regression was run using weights, those weights will always be used to generate estimates of the prototypical individual to be used.
dvname	dvname (optional) allows the user to relabel the dependent variable for printouts.

acclevnames	dvname (optional) allows the user to specify the names for the specified levels of the accross variable.
bylevnames	dvname (optional) allows the user to specify the names for the specified levels of the by variable.
atlevnames	dvname (optional) allows the user to specify the names for the specified levels of the at variable.
stdzacross	dvname (optional) shows levels of across variable in (weighted) standard deviation units. This defaults to showing 1SD below mean and 1SD above mean; specifying acrosslevs to other values will provide results in SD units instead of variable units.
stdzby	dvname (optional) shows levels of by variable in (weighted) standard deviation units. This defaults to showing 1SD below mean and 1SD above mean; specifying bylevs to other values will provide results in SD units instead of variable units.
stdzat	dvname (optional) shows levels of at variable in (weighted) standard deviation units. This defaults to showing 1SD below mean and 1SD above mean; specifying atlevs to other values will provide results in SD units instead of variable units.
limitlevs	limitlevs sets the number of different levels that any given interacting variable can have. This is meant to prevent inadvertent generation and plotting of tons of point estimates for continuous variables. The default is set to 20.
type	type sets the type of prediction to be used for generation of the estimates. This defaults to "response" but can be used with any type of model prediction for which only one numeric estimate is given. (Not currently compatible with estimates derived from polr regression).
seplot	seplot (optional) if set to TRUE, plots will include polygons illustrating standard errors.
ylim	ylim (optional) passes on y-axis limits to plot function.
main	main (optional) passes on title to plot function.
xlab	xlab (optional) passes on x-axis labels to plot function.
ylab	ylab (optional) passes on y-axis labels to plot function.
legend	legend (optional) if TRUE will produce a legend on the interaction figure.
placement	placement (optional) passes to legend function a location for the legend. Can be set to "bottomright", "bottomleft", "topright", and "topleft".
lwd	lwd (optional) specifies the line strength for plots, this passes on to the plot command.
add	add (optional) logical statement to add the results to an existing plot (at=TRUE) rather than generating a new one (at=FALSE is the default).
addat	addat (optional) logical statement specifying whether the levels of at should be different plots (addat=TRUE) or if each level of at should generate a new plot (addat=FALSE is the default)
addby	addby (optional) logical statement specifying whether the levels of by should be different plots (addby=TRUE) or if each level of by should generate a new plot (addby=TRUE is the default)

`mfrow` — `mfrow` (optional) temporarily changes the number of plots per page in `par` for the purpose of generating current plots. This should generally only be used for 3-way interactions. It takes commands of the form `c(2,3)`, specifying the number of rows and columns in the graphics interface. The algorithm defaults to putting all 3-way interactions on a single page with a width of 2.

`linecol` — `linecol` (optional) Specifies the colors of lines in the figure(s). For two-way interactions, this should be a vector of the same length as `bylevs`. For 3-way interactions, the colors demarcate the levels of `at` instead and should be the same length as `atlevs`.

`secol` — `secol` (optional) Specifies the colors of standard error in the figure(s). For two-way interactions, this should be a vector of the same length as `bylevs`. For 3-way interactions, the colors demarcate the levels of `at` instead and should be the same length as `atlevs`.

`showbynamelegend` — `showbynamelegend` (optional) adds name of `by` variable to names of value levels in legend.

`showatnamelegend` — `showatnamelegend` (optional) adds name of `at` variable to names of value levels in legend.

`lty` — `lty` (optional) line type to pass on to plot.

`density` — `density` (optional) line density for standard error plots.

`startangle` — `startangle` (optional) line angle for standard error plots.

`approach` — `approach` determines whether you want to estimate counterfactuals for a prototypical individual `approach="prototypical"` (the default), for the entire population `approach="population"`, or for individuals in the subgroups specified in the `by` and `at` categories `approach="at"`, `approach="by"`, `approach="atby"`.

`data` — `data` (optional) allows you to replace the dataset used in the regression to produce other prototypical values.

`...` — `...` (optional) Additional arguments to be passed on to plot command (or future methods of `findwtdinteraction`).

Value

A table or figure illustrating the predicted values of the dependent variable across levels of the independent variables for a prototypical respondent.

Author(s)

Josh Pasek, Assistant Professor of Communication Studies at the University of Michigan (www.joshpasek.com).

rd *Round Numbers To Text With No Leading Zero*

Description

Rounds numbers to text and drops leading zeros in the process.

Usage

```
rd(x, digits=2, add=TRUE, max=(digits+3))
```

Arguments

x	A vector of values to be rounded (must be numeric).
digits	The number of digits to round to (must be an integer).
add	An optional dichotomous indicator for whether additional digits should be added if no numbers appear in pre-set digit level.
max	Maximum number of digits to be shown if add=TRUE.

Value

A vector of length equal to that of x of class character.

Author(s)

Josh Pasek, Assistant Professor of Communication Studies at the University of Michigan (www.joshpasek.com).

Examples

```
rd(seq(0, 1, by=.1))
```

starmaker *Produce stars from p values for tables.*

Description

Recodes p values to stars for use in tables.

Usage

```
starmaker(x, p.levels=c(.001, .01, .05, .1), symbols=c("***", "**", "*", "+"))
```

Arguments

x	A vector of p values to be turned into stars (must be numeric).
p.levels	A vector of the maximum p value for each symbol used (p<p.level).
symbols	A vector of the symbols to be displayed for each p value.

Value

A vector of length equal to that of x of class character.

Author(s)

Josh Pasek, Assistant Professor of Communication Studies at the University of Michigan (www.joshpasek.com).

Examples

```
starmaker(seq(0, .15, by=.01))
cbind(p=seq(0, .15, by=.01), star=starmaker(seq(0, .15, by=.01)))
```

---

stdz *Standardizes any numerical vector, with weights.*

Description

stdz produces a standardized copy of any input variable. It can also standardize a weighted variable to produce a copy of the original variable standardized around its weighted mean and variance.

Usage

```
stdz(x, weight=NULL)
```

Arguments

x	x should be a numerical vector which the researcher wishes to standardize.
weight	weight is an optional vector of weights to be used to determining the weighted mean and variance for standardization.

Value

A vector of length equal to x with a (weighted) mean of zero and a (weighted) standard deviation of 1.

Author(s)

Josh Pasek, Assistant Professor of Communication Studies at the University of Michigan (www.joshpasek.com).

See Also

wtd.cor wtd.chi.sq wtd.t.test

Examples

```
test <- c(1,1,1,1,1,1,2,2,2,3,3,3,4,4)
weight <- c(.5,.5,.5,.5,.5,1,1,1,1,2,2,2,2,2)

summary(stdz(test))
summary(stdz(test, weight))
wtd.mean(stdz(test, weight), weight)
wtd.var(stdz(test, weight), weight)
```

---

wpct *Provides a weighted table of percentages for any variable.*

Description

wpct produces a weighted table of the proportion of data in each category for any variable. This is simply a weighted frequency table divided by its sum.

Usage

```
wpct(x, weight=NULL, na.rm=TRUE, ...)
```

Arguments

x	x should be a vector for which a set of proportions is desired.
weight	weight is a vector of weights to be used to determining the weighted proportion in each category of x.
na.rm	If na.rm is true, missing data will be dropped. If na.rm is false, missing data will return an error.
...	... (optional) Additional arguments to be passed on to wtd.table.

Value

A table object of length equal to the number of separate values of x.

Author(s)

Josh Pasek, Assistant Professor of Communication Studies at the University of Michigan (www.joshpasek.com).

Examples

```
test <- c(1,1,1,1,1,1,2,2,2,3,3,3,4,4)
weight <- c(.5,.5,.5,.5,.5,1,1,1,1,2,2,2,2,2)

wpct(test)
wpct(test, weight)
```

wtd.chi.sq *Produces weighted chi-squared tests.*

Description

wtd.chi.sq produces weighted chi-squared tests for two- and three-variable contingency tables. Decomposes parts of three-variable contingency tables as well. Note that weights run with the default parameters here treat the weights as an estimate of the precision of the information. A prior version of this software was set to default to mean1=FALSE.

Usage

```
wtd.chi.sq(var1, var2, var3=NULL, weight=NULL, na.rm=TRUE,
drop.missing.levels=TRUE, mean1=TRUE)
```

Arguments

var1	var1 is a vector of values which the researcher would like to use to divide any data set.
var2	var2 is a vector of values which the researcher would like to use to divide any data set.
var3	var3 is an optional additional vector of values which the researcher would like to use to divide any data set.
weight	weight is an optional vector of weights to be used to determine the weighted chi-squared for all analyses.
na.rm	na.rm removes missing data from analyses.
drop.missing.levels	drop.missing.levels drops missing levels from variables.
mean1	mean1 is an optional parameter for determining whether the weights should be forced to have an average value of 1. If this is set as false, the weighted correlations will be produced with the assumption that the true N of the data is equivalent to the sum of the weights.

Value

A two-way chi-squared produces a vector including a single chi-squared value, degrees of freedom measure, and p-value for each analysis.

A three-way chi-squared produces a matrix with a single chi-squared value, degrees of freedom measure, and p-value for each of seven analyses. These include: (1) the values using a three-way contingency table, (2) the values for a two-way contingency table with each pair of variables, and (3) assessments for whether the relations between each pair of variables are significantly different across levels of the third variable.

Author(s)

Josh Pasek, Assistant Professor of Communication Studies at the University of Michigan (www.joshpasek.com).

See Also

wtd.cor wtd.t.test

Examples

```
var1 <- c(1,1,1,1,1,2,2,2,2,2,3,3,3,3,3)
var2 <- c(1,1,2,2,3,3,1,1,2,2,3,3,1,1,2)
var3 <- c(1,2,3,1,2,3,1,2,3,1,2,3,1,2,3)
weight <- c(.5,.5,.5,.5,.5,1,1,1,1,1,2,2,2,2,2)

wtd.chi.sq(var1, var2)
wtd.chi.sq(var1, var2, weight=weight)

wtd.chi.sq(var1, var2, var3)
wtd.chi.sq(var1, var2, var3, weight=weight)
```

---

wtd.cor *Produces weighted correlations with standard errors and significance. For a faster version without standard errors and p values, use the* wtd.cors *function.*

Description

wtd.cor produces a Pearsons correlation coefficient comparing two variables or matrices. Note that weights run with the default parameters here treat the weights as an estimate of the precision of the information. For survey data, users should run this code with bootstrapped standard errors bootse=TRUE, which are robust to heteroskadesticity, although these will vary slightly each time the weights are run. A prior version of this software was set to default to mean1=FALSE and bootse=FALSE.

Usage

```
wtd.cor(x, y=NULL, weight=NULL, mean1=TRUE, collapse=TRUE, bootse=FALSE,
bootp=FALSE, bootn=1000)
```

Arguments

x	x should be a matrix or vector which the researcher wishes to correlate with y.
y	y should be a numerical vector or matrix which the researcher wishes to correlate with x. If y is NULL, x will be used instead
weight	weight is an optional vector of weights to be used to determining the weighted mean and variance for calculation of the correlations.
mean1	mean1 is an optional parameter for determining whether the weights should be forced to have an average value of 1. If this is set as false, the weighted correlations will be produced with the assumption that the true N of the data is equivalent to the sum of the weights.

collapse	collapse is an indicator for whether the data should be collapsed to a simpler form if either x or y is a vector instead of a matrix.
bootse	bootse is an optional parameter that produces bootstrapped standard errors. This should be used to address heteroskadesticity issues when weights indicate probabilities of selection rather than the precision of estimates.
bootp	bootp is an optional parameter that produces bootstrapped p values instead of estimating p values from the standard errors. This parameter only operates when bootse=TRUE.
bootn	bootn is an optional parameter that is used to indicate the number of bootstraps that should be run for bootse and bootp.

Value

A list with matrices for the estimated correlation coefficient, the standard error on that correlation coefficient, the t-value for that correlation coefficient, and the p value for the significance of the correlation. If the list can be simplified, simplification will be done.

Author(s)

Josh Pasek, Assistant Professor of Communication Studies at the University of Michigan (www.joshpasek.com).

See Also

wtd.cors stdz wtd.t.test wtd.chi.sq

Examples

```
test <- c(1,1,1,1,1,1,2,2,2,3,3,3,4,4)
t2 <- rev(test)
weight <- c(.5,.5,.5,.5,1,1,1,1,2,2,2,2,2)

wtd.cor(test, t2)
wtd.cor(test, t2, weight)
wtd.cor(test, t2, weight, bootse=TRUE)
```

---

wtd.cors *Produces weighted correlations quickly using C.*

Description

wtd.cors produces a Pearsons correlation coefficient comparing two variables or matrices.

Usage

```
wtd.cors(x, y=NULL, weight=NULL)
```

Arguments

x	x should be a matrix or vector which the researcher wishes to correlate with y.
y	y should be a numerical vector or matrix which the researcher wishes to correlate with x. If y is NULL, x will be used instead
weight	weight is an optional vector of weights to be used to determining the weighted mean and variance for calculation of the correlations.

Value

A matrix of the estimated correlation coefficients.

Author(s)

Marcus Schwemmle at GfK programmed the C code, R wrapper by Josh Pasek, Assistant Professor of Communication Studies at the University of Michigan (www.joshpasek.com).

See Also

wtd.cor stdz wtd.t.test wtd.chi.sq

Examples

```
test <- c(1,1,1,1,1,1,2,2,2,3,3,3,4,4)
t2 <- rev(test)
weight <- c(.5,.5,.5,.5,.5,1,1,1,1,2,2,2,2,2)

wtd.cors(test, t2)
wtd.cors(test, t2, weight)
```

---

wtd.hist *Weighted Histograms*

Description

Produces weighted histograms by adding a "weight" option to the his.default function from the graphics package (Copyright R-core). The code here was copied from that function and modified slightly to allow for weighted histograms as well as unweighted histograms. The generic function hist computes a histogram of the given data values. If plot=TRUE, the resulting object of class "histogram" is plotted by plot.histogram, before it is returned.

Usage

```
wtd.hist(x, breaks = "Sturges",
     freq = NULL, probability = !freq,
     include.lowest = TRUE, right = TRUE,
     density = NULL, angle = 45, col = NULL, border = NULL,
     main = paste("Histogram of" , xname),
```

```
     xlim = range(breaks), ylim = NULL,
     xlab = xname, ylab,
     axes = TRUE, plot = TRUE, labels = FALSE,
     nclass = NULL, weight = NULL, ...)
```

Arguments

x	a vector of values for which the histogram is desired.
breaks	one of: • a vector giving the breakpoints between histogram cells, • a single number giving the number of cells for the histogram, • a character string naming an algorithm to compute the number of cells (see 'Details'), • a function to compute the number of cells. In the last three cases the number is a suggestion only.
freq	logical; if TRUE, the histogram graphic is a representation of frequencies, the counts component of the result; if FALSE, probability densities, component density, are plotted (so that the histogram has a total area of one). Defaults to TRUE *if and only if* breaks are equidistant (and probability is not specified).
probability	an *alias* for !freq, for S compatibility.
include.lowest	logical; if TRUE, an x[i] equal to the breaks value will be included in the first (or last, for right = FALSE) bar. This will be ignored (with a warning) unless breaks is a vector.
right	logical; if TRUE, the histogram cells are right-closed (left open) intervals.
density	the density of shading lines, in lines per inch. The default value of NULL means that no shading lines are drawn. Non-positive values of density also inhibit the drawing of shading lines.
angle	the slope of shading lines, given as an angle in degrees (counter-clockwise).
col	a colour to be used to fill the bars. The default of NULL yields unfilled bars.
border	the color of the border around the bars. The default is to use the standard foreground color.
main, xlab, ylab	these arguments to title have useful defaults here.
xlim, ylim	the range of x and y values with sensible defaults. Note that xlim is *not* used to define the histogram (breaks), but only for plotting (when plot = TRUE).
axes	logical. If TRUE (default), axes are draw if the plot is drawn.
plot	logical. If TRUE (default), a histogram is plotted, otherwise a list of breaks and counts is returned. In the latter case, a warning is used if (typically graphical) arguments are specified that only apply to the plot = TRUE case.
labels	logical or character. Additionally draw labels on top of bars, if not FALSE; see plot.histogram in the graphics package.
nclass	numeric (integer). For S(-PLUS) compatibility only, nclass is equivalent to breaks for a scalar or character argument.

weight	numeric. Defines a set of weights to produce a weighted histogram. Will default to 1 for each case if no other weight is defined.
...	further arguments and graphical parameters passed to plot.histogram and thence to title and axis (if plot=TRUE).

Details

The definition of *histogram* differs by source (with country-specific biases). R's default with equi-spaced breaks (also the default) is to plot the (weighted) counts in the cells defined by breaks. Thus the height of a rectangle is proportional to the (weighted) number of points falling into the cell, as is the area *provided* the breaks are equally-spaced.

The default with non-equi-spaced breaks is to give a plot of area one, in which the *area* of the rectangles is the fraction of the data points falling in the cells.

If right = TRUE (default), the histogram cells are intervals of the form (a,b], i.e., they include their right-hand endpoint, but not their left one, with the exception of the first cell when include.lowest is TRUE.

For right = FALSE, the intervals are of the form [a,b), and include.lowest means '*include highest*'.

The default for breaks is "Sturges": see nclass.Sturges. Other names for which algorithms are supplied are "Scott" and "FD" / "Freedman-Diaconis" (with corresponding functions nclass.scott and nclass.FD). Case is ignored and partial matching is used. Alternatively, a function can be supplied which will compute the intended number of breaks as a function of x.

Value

an object of class "histogram" which is a list with components:

breaks	the $n+1$ cell boundaries (= breaks if that was a vector). These are the nominal breaks, not with the boundary fuzz.
counts	n values; for each cell, the number of x[] inside.
density	values for each bin such that the area under the histogram totals 1. $\hat{f}(x_i\omega_i)$ / $f^(x[i]\omega[i])$, as estimated density values. If all(diff(breaks) == 1), they are the relative frequencies counts/n and in general satisfy $\sum_i \hat{f}(x_i\omega_i)(b_{i+1} - b_i) = 1$ / $sum[i; f^(x[i]\omega[i])(b[i+1] - b[i])] = 1$, where b_i = breaks[i].
intensities	same as density. Deprecated, but retained for compatibility.
mids	the n cell midpoints.
xname	a character string with the actual x argument name.
equidist	logical, indicating if the distances between breaks are all the same.

Author(s)

Josh Pasek, Assistant Professor of Communication Studies at the University of Michigan (www.joshpasek.com) was responsible for the updates to the hist function necessary to implement weighted counts. The hist.default code from the graphics package on which the current function was based was written by R-core. All modifications are noted in code and the copyright for all original code remains with R-core.

Examples

```
var1 <- c(1:100)
wgt <- var1/mean(var1)
par(mfrow=c(2, 2))
wtd.hist(var1)
wtd.hist(var1, weight=wgt)
wtd.hist(var1, weight=var1)
```

`wtd.t.test` *Produces weighted Student's t-tests with standard errors and significance.*

Description

`wtd.t.test` produces either one- or two-sample t-tests comparing weighted data streams to one another. Note that weights run with the default parameters here treat the weights as an estimate of the precision of the information. For survey data, users should run this code with bootstrapped standard errors `bootse=TRUE`, which are robust to heteroskadesticity, although these will vary slightly each time the weights are run. A prior version of this software was set to default to `mean1=FALSE` and `bootse=FALSE`.

Usage

```
wtd.t.test(x, y=0, weight=NULL, weighty=NULL, samedata=TRUE,
alternative="two.tailed", mean1=TRUE, bootse=FALSE, bootp=FALSE,
bootn=1000, drops="pairwise")
```

Arguments

`x`	`x` is a numerical vector which the researcher wishes to test against `y`.
`y`	`y` can be either a single number representing an alternative hypothesis or a second numerical vector which the researcher wishes to compare against `x`.
`weight`	`weight` is an optional vector of weights to be used to determine the weighted mean and variance for the `x` vector for all t-tests. If `weighty` is unspecified and `samedata` is TRUE, this weight will be assumed to apply to both `x` and `y`.
`weighty`	`weighty` is an optional vector of weights to be used to determine the weighted mean and variance for the `y` vector for two-sample t-tests. If `weighty` is unspecified and `samedata` is TRUE, this weight will be assumed to equal `weightx`. If `weighty` is unspecified and `samedata` is FALSE, this weight will be assumed to equal 1 for all cases.
`samedata`	`samedata` is an optional identifier for whether the `x` and `y` data come from the same data stream for a two-sample test. If true, `wtd.t.test` assumes that `weighty` should equal `weightx` if (1) `weighty` is unspecified, and (2) the lengths of the two vectors are identical.

alternative	alternative is an optional marker for whether one or two-tailed p-values shoould be returned. By default, two-tailed values will be returned (type="two.tailed"). To set to one-tailed values, alternative can be set to type="greater" to test x>y or type="less" to test x<y.
mean1	mean1 is an optional parameter for determining whether the weights should be forced to have an average value of 1. If this is set as false, the weighted correlations will be produced with the assumption that the true N of the data is equivalent to the sum of the weights.
bootse	bootse is an optional parameter that produces bootstrapped standard errors. This should be used to address heteroskadesticity issues when weights indicate probabilities of selection rather than the precision of estimates.
bootp	bootp is an optional parameter that produces bootstrapped p values instead of estimating p values from the standard errors. This parameter only operates when bootse=TRUE.
bootn	bootn is an optional parameter that is used to indicate the number of bootstraps that should be run for bootse and bootp.
drops	drops is set to limit a t-test on the same data to cases with nonmissing data for x, y, and weights (if specified). If drops is anything other than "pairwise", means for x and y are calculated on all available data rather than data that are available for both x and y. This parameter does nothing if x and y are not from the same dataset.

Value

A list element with an identifier for the test; coefficients for the t value, degrees of freedom, and p value of the t-test; and additional statistics of potential interest.

Author(s)

Josh Pasek, Assistant Professor of Communication Studies at the University of Michigan (www.joshpasek.com). Gene Culter added code for a one-tailed version of the test.

See Also

stdz wtd.cor wtd.chi.sq

Examples

```
test <- c(1,1,1,1,1,1,2,2,2,3,3,3,4,4)
t2 <- rev(test)+1
weight <- c(.5,.5,.5,.5,1,1,1,1,2,2,2,2,2)

wtd.t.test(test, t2)
wtd.t.test(test, t2, weight)
wtd.t.test(test, t2, weight, bootse=TRUE)
```

Index