Package 'lessR'

September 23, 2014

Version 3.1.1

Date 2014-09-22

Title Less Code, More Results

Author David W. Gerbing, School of Business Administration, Portland State University

Maintainer David W. Gerbing <gerbing@pdx.edu>

Depends R (>= 2.15.0)

Imports car, leaps, MBESS, foreign, gdata, triangle

Description Each function accomplishes the work of several or more standard R functions. For example, two function calls, Read() and CountAll(), read the data and generate descriptive statistics for all variables in the data frame, plus histograms and bar charts as appropriate. Other functions provide for descriptive statistics, a comprehensive regression analysis, analysis of variance and t-test, plotting, bar chart, histogram, box plot, density curves, calibrated power curve, reading multiple data formats with the same function and color themes. The function Help provides a help system that suggests specific analyses and functions. Variable labels are available. A confirmatory factor analysis of multiple indicator measurement models is also available as is a pedagogical routines for data simulation such as for the Central Limit Theorem.

License GPL (>= 2)

LazyLoad yes

NeedsCompilation yes

Repository CRAN

Date/Publication 2014-09-23 05:48:55

R topics documented:

ANOVA *Analysis of Variance*

Description

Abbreviation: av, av.brief

Analysis of variance from the R aov function plus graphics and effect sizes. Permitted designs are one-way between groups, two-way between groups and randomized blocks with one treatment factor with one observation for each treatment and block combination.

Usage

```
ANOVA(my.formula, data=mydata, brief=getOption("brief"), digits.d=NULL,
         rb.points=TRUE, res.rows=NULL, res.sort=c("zresid", "fitted", "off"),
         pdf=FALSE, pdf.width=5, pdf.height=5, ...)

av(...)

av.brief(..., brief=TRUE)
```

Arguments

my.formula	Standard R formula for specifying a model.
data	The default name of the data frame that contains the data for analysis is mydata, otherwise explicitly specify.
brief	If set to TRUE, reduced text output. Can change system default with set function.
digits.d	For the Basic Analysis, it provides the number of decimal digits. For the rest of the output, it is a suggestion only.
rb.points	For a randomized block design, a plot of the fitted value for each cell is obtained as well as the individual data values. Set to FALSE to suppress the data values.
res.rows	Default is 20, which lists the first 20 rows of data and residuals sorted by the specified sort criterion. To disable residuals, specify a value of 0. To see the residuals output for all observations, specify a value of "all".
res.sort	Default is "zresid", for specifying standardized residuals as the sort criterion for the display of the rows of data and associated residuals. Other values are "fitted" for the fitted values and "off" to not sort the rows of data.

pdf	Indicator as to if the graphic files should be saved as pdf files instead of directed to the standard graphics windows.
pdf.width	Width of the pdf file in inches.
pdf.height	Height of the pdf file in inches.
...	Other parameter values for R function lm which provides the core computations.

Details

OVERVIEW
The one-way ANOVA with Tukey HSD and plot based on the R functions aov, TukeyHSD, and provides summary statistics for each level. Two-factor ANOVA also provides an interaction plot of the means with interaction.plot as well as a table of means and other summary statistics. The two-factor analysis can be between groups or a randomized blocked design. Residuals are displayed by default. Tukey HSD comparisons and residuals are not displayed if brief=TRUE.

MODEL SPECIFICATION
In the following specifications, Y is the response variable, X is a treatment variable and Blocks is the blocking variable. The distinction between the one-way randomized blocks and the two-way between groups models is not the variable names, but rather the delimiter between the variable names. Use * to indicate a two-way crossed between groups design and + for a randomized blocks design.
one-way between groups: ANOVA(Y ~ X)
one-way randomized blocks: ANOVA(Y ~ X + Blocks)
two-way between groups: ANOVA(Y ~ X1 * X2)
For more complex designs, use the standard R function aov upon which ANOVA depends.

BALANCED DESIGN
The design for the two-factor analyses must be balanced. A check is performed and processing ceases if not balanced. For unbalanced designs, consider the function lmer in the lme4 package.

DECIMAL DIGITS
The number of decimal digits displayed on the output is, by default, the maximum number of decimal digits for all the data values of the response variable. Or, this value can be explicitly specified with the digits.d parameter.

Value

Following the standard R function aov, invisibly returns an object of class c("aov", "lm").

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

References

Gerbing, D. W. (2013). R Data Analysis without Programming, Chapter 7, NY: Routledge.

See Also

aov, TukeyHSD, interaction.plot

Examples

```
# create simulated data, no population mean difference
# X has two values only, Y is numeric
# put into a data frame
n <- 12
X <- sample(c("Group1","Group2","Group3"), size=n, replace=TRUE)
Y <- round(rnorm(n=n, mean=50, sd=10), 2)
mydata <- data.frame(X,Y)

# analyze data with formula version
# variable names and levels of X are automatically obtained from data
# although data frame not attached, reference variable names directly
ANOVA(Y ~ X)
# short name
av(Y ~ X)

# variable of interest is in a data frame which is not the default mydata
# access the data frame in the lessR dat.Learn data set
# although data not attached, access the variables directly by their name
data(dataLearn)
ANOVA(Score ~ StudyType, data=dataLearn)

# two-factor between-groups ANOVA with replications and interaction
ANOVA(breaks ~ wool * tension, data=warpbreaks)

# randomized blocks design with the second term the blocking factor
ANOVA(breaks ~ wool + tension, data=warpbreaks)
```

BarChart *Bar Chart of One or Two Variables*

Description

Abbreviation: bc

Plots a bar chart of one or two variables with default colors, including background color and grid lines, from a variety of different types of data. For two variables a legend is also provided. Also displays the frequency table for one or two variables and optionally provides the corresponding chi-square inferential analysis. For two variables, the frequencies include the joint and marginal frequencies.

For the analysis of a single variable, the cell proportions are also provided unless there are more than 10 unique values. If all values are unique, then only the value names are listed. For the analysis of two variables, also provided are the proportions within each cell, within each column and within each row.

Unlike the standard R function, barplot, the variable(s) can be entered directly into the function call without first converting to a table. If two variables are plotted, a legend is automatically provided in the right margin.

If the provided object to analyze is a set of multiple variables, including an entire data frame, then a bar chart is calculated for each non-numeric variable in the data frame and the results written to a

pdf file in the current working directory. The name of each output pdf file that contains a bar chart and its path are specified in the output.

Usage

```
BarChart(x=NULL, by=NULL, data=mydata, n.cat=getOption("n.cat"),

         col.fill=NULL, col.stroke=getOption("col.stroke.bar"),
         col.bg=getOption("col.bg"),
         col.grid=getOption("col.grid"),
         random.col=FALSE,
         colors=c("rainbow", "terrain", "heat"),

         horiz=FALSE, over.grid=FALSE, addtop=1,
         gap=NULL, prop=FALSE,

         xlab=NULL, ylab=NULL, main=NULL,
         cex.axis=.85, col.axis="gray30", col.ticks="gray30",

         beside=FALSE, col.low=NULL, col.hi=NULL, count.levels=NULL,

         legend.title=NULL, legend.loc="right.margin", legend.labels=NULL,
         legend.horiz=FALSE,

         quiet=getOption("quiet"),
         pdf.file=NULL, pdf.width=5, pdf.height=5, ...)

bc(...)
```

Arguments

x	Variable(s) to analyze. Can be a single variable, either within a data frame or as a vector in the user's workspace, or multiple variables in a data frame such as designated with the c function, or an entire data frame. If not specified, then defaults to all non-numerical variables in the specified data frame, mydata by default.
by	For each level of the first variable, x, display the frequencies at each level of this second variable, y.
data	Optional data frame that contains the variables of interest, default is mydata.
n.cat	When analyzing all the variables in a data frame, specifies the largest number of unique values of variable of a numeric data type for which the variable will be analyzed as a categorical. Set to 0 to turn off.
col.fill	Specified bar colors.
col.stroke	Color of the border of the bars. Black by default unless the background is dark. Specify NA to remove the border
col.bg	Color of the plot background.
col.grid	Color of the grid lines.

colors	Optional palettes that provide more options, which include values of "heat", "rainbow" and "terrain".
random.col	For two variable plots, when TRUE, randomizes the order of the colors within the chosen color palette, when the second variable is not ordered, otherwise is ignored. Each analysis generally yields different colors of the bars.
horiz	By default bars are vertical, but can set this option to TRUE.
over.grid	If TRUE, plot the grid lines over the histogram.
addtop	When horiz=FALSE, in the same scale as the vertical axis, puts more space between the bars and the top of the plot area, usually to accommodate the legend when plotting two variables.
gap	Gap between bars. Provides the value of the space option from the standard R barplot function with a default of 0.2 unless two variables are plotted and beside=TRUE, in which case the default is c(.1,1).
prop	Display proportions instead raw frequencies.
xlab	Label for x-axis, more generally the label for the values which could be on the vertical axis for a two variable plot if horiz=TRUE. Defaults to variable name.
ylab	Label for y-axis, more generally the frequency axis. Defaults to Frequency.
main	Title of graph.
cex.axis	Scale magnification factor, which by default displays the axis values to be smaller than the axis labels.
col.axis	Color of the font used to label the axis values.
col.ticks	Color of the ticks used to label the axis values.
beside	For a two variable plot, set to TRUE for the levels of the first variable to be plotted as adjacent bars instead of stacked on each other.
col.low	Only when the variable is an ordered factor, sets the color for the lowest level of the factor in the resulting ordered progression of colors.
col.hi	Only when the variable is an ordered factor, sets the color for the highest level of the factor in the resulting ordered progression of colors.
count.levels	If the name of a variable, this signals that the primary variable x has values that are counts, already tabulated, and that the specified variable here contains the names of the levels of x.
legend.title	Title of the legend, which is usually set by default except when raw counts are entered as a matrix. Then a title must be specified to generate a legend.
legend.loc	When plotting two variables, location of the legend, with the default in the right margin. Additional options from standard R are "topleft", "top", "topright" and others as shown in the help for the legend function.
legend.labels	When plotting two variables, labels for the legend, which by default are the levels for the second or by variable.
legend.horiz	By default the legend is vertical, but can be changed to horizontal.
quiet	If set to TRUE, no text output. Can change system default with set function.
pdf.file	Name of the pdf file to which graphics are redirected.

pdf.width	Width of the pdf file in inches.
pdf.height	Height of the pdf file in inches.
...	Other parameter values for graphics as defined by barplot, legend, and par including space for one variable only, and cex.lab, col.main, etc.

Details

OVERVIEW

Plot a bar chart with default colors for one or two variables, presumably with a relatively small number of values for each variable. By default, colors are selected for the bars, background and grid lines, all of which can be customized. The basic computations of the chart are provided with the standard R functions barplot, chisq.test and, for two variables, legend. Horizontal bar charts, specified by horiz=TRUE, list the value labels horizontally and automatically extend the left margin to accommodate both the value labels and the variable label.

The form of the entered data, the first variable x and optionally a second variable, y, is flexible. The data may be entered as factors, numeric values, characters, or a matrix. The data may be entered and the resulting frequencies computed, or the frequencies can be entered directly. The most natural type of data to enter, when entering the variables, is to enter factors. Plus, the bar colors for a second variable which is an ordered factor are also ordered in a corresponding progression.

DATA

The data may either be a vector from the global environment, the user's workspace, as illustrated in the examples below, or a variable in a data frame. The default input data frame is mydata. Specify a different data frame name with the data option. Regardless of its name, the variables in the data frame are referenced directly by their names, that is, no need to invoke the standard R mechanisms of the mydata$name notation, the with function or the attach function.

If the name of vector in the global environment and of a variable in the input data frame are the same, the vector is analyzed. If two variables are specified, both variables should be in the data frame, or one of the variables is in the data frame and the other in the global environment.

To obtain a bar chart of each numerical variable in the mydata data frame, use bar chart(). Or, for a data frame with a different name, insert the name between the parentheses. To analyze a subset of the variables in a data frame, specify the list with either a : or the c function, such as m01:m03 or c(m01,m02,m03).

COLORS

For a one variable plot, the color of the bars is set by the current color theme according to the function set, which includes the default color theme activated when a lessR session is initiated. When two variables are plotted, a set of arbitrary colors represent the various levels of the second variable. Transparency effects are only incorporated for a one variable bar plot in a single color.

There are three ways to override the default colors.

1. There are two pre-defined color palettes, each with 7 colors. Three more built-in R color palettes are also available by setting colors to one of "rainbow", "heat" and "terrain". The most vivid of all the palettes is "rainbow".

2. The order of the colors within the chosen palette can be randomized with the random.col="TRUE" option. For example, when this option is activated each of the seven colors in a palette has a 1/7 chance of appearing as the first color, the only color used in the plot of a single variable, and the color used for the first bar in each group for a plot of two variables. When invoked for a color="gray", the order from light to dark will generally be lost, which may be desirable if the categories do not represent an ordered factor.

3. The desired colors can be explicitly specified with the `col.bars` option, which overrides any other bar color options. When plotting one variable, include one color in this color list, the color used for all of the bars. When plotting two variables, usually the list of colors includes the same number of elements as the number of levels for the second variable. As always with R, if the list includes more than once color, the `c` function must be used to generate the list, as in `col.bars=c("coral3","seagreen3")` for a second variable with two levels. When two variables are plotted, if there are fewer specified colors than the levels of the second variable, the remaining colors are selected from the remaining colors in the activated palette.

The default colors in one of the two provided color palettes can be viewed, in the order in which they are displayed, by running the corresponding two lines of R code, for the default colors:

```
clr <- c("slategray3", "bisque3", "darksalmon", "darkolivegreen3", "thistle", "azure3", "moccasin")
barplot(rep(1,7), names=clr, col=clr, border=NA, space=.1)
```

When plotting one ordered factor, or when plotting two variables, and the second variable is an ordered factor, then neither of the two standard color palettes are used. Instead, the resulting bar colors for each level of the ordered factor are also ordered in a progression of colors. The default progression is based on the first color of either the regular or gray color palettes, but this can be changed with the `col.low` and `col.hi` options, or individually specify the color of each bar with the `col.bars` option. A specified palette can, for example, be from light to dark of the same hue, or from a light color of one hue to a dark color of another hue. Each color value can be specified with a color name, or with a specification with the `rgb` function. See the examples below.

Use the `showColors` function in this package to get, for each color: name, sample color swatch, and corresponding rgb specification. For a very small number of levels, such as two, it is may be desirable to specify the low and high values to not be closer to each other than the default values.

LEGEND

When two variables are plotted, a legend is produced, with values for each level of the second or `by` variable. By default, the location is placed in the right margin of the plot. This position can be changed with the `legend.loc` option, which, in addition to the `lessR` option of `right.margin`, accepts any valid value consistent with the standard R `legend` function, used to generate the legend.

If the default right margin is retained, variable labels are also accommodated for the legend title. To conserve horizontal space, the variable label is listed in multiple lines if needed. The legend title is constructed by forming lines of maximum length of 12 characters, with multiple words per line if possible. Any single word in the label of more than 12 characters is abbreviated to 12 characters with the R `abbreviate` function. Also, any value labels are abbreviated to a maximum of 6 characters.

If the legend is not in the right margin, sometimes bars from the graph may intrude into the legend. One response is to re-run the analysis with the legend in a new location. Another response is to invoke the `addtop` option to place more space between the top bar in the graph and the top of the graph. This option only applies for the default vertical bars. Also, the legend is displayed vertically by default, but can be changed to horizontal with the `legend.horiz` option.

ENTER COUNTS DIRECTLY

Instead of calculating the counts from the data, the counts can be entered directly. For two variables, enter the counts as a matrix and include the `xlab` option to label the horizontal axis, such as with the name of the variable. Also include the `legend.title` option to provide a legend. See the examples below.

Or, include the tabulated counts as the data which is read into R. If `count.levels` is not `NULL`,

then it is presumed to be a valid variable name. As such, it indicates that the primary variable, x consists of values already tabulated, that is, counts, and is ready to be plotted directly. The value for count.levels specifies the label for each level of x.

STATISTICS
In addition to the barchart, descriptive and optional inferential statistics are also presented. First, the frequency table for one variable or the joint frequency table for two variables is displayed. Second, the corresponding chi-square test is also displayed by default.

VARIABLE LABELS
If variable labels exist, then the corresponding variable label is listed as the label for the horizontal axis unless xlab is specified in the function call. If there are two variables to plot, the title of the resulting plot is based on the two variable labels, unless a specific title is listed with the main option. The variable label is also listed in the text output, next to the variable name. If the analysis is for two variables, then labels for both variables are included.

PDF OUTPUT
Because of the customized graphic windowing system that maintains a unique graphic window for the Help function, the standard graphic output functions such as pdf do not work with the lessR graphics functions. Instead, to obtain pdf output, use the pdf.file option, perhaps with the optional pdf.width and pdf.height options. These files are written to the default working directory, which can be explicitly specified with the R setwd function.

Value

If the analysis is of a single categorical variable, a list is invisibly returned with two tables, the frequencies and the proportions, respectively named freq and prop. If two categorical variables are analyzed, then nothing is returned.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

References

Gerbing, D. W. (2013). R Data Analysis without Programming, Chapter 4, NY: Routledge.

See Also

barplot, table, legend.

Examples

```
# ------------------------------------------------------
# generate some data in data frame mydata for two variables
# ------------------------------------------------------

# Pain is an ordered factor, Gender is an unordered factor
# Place in data frame mydata to simulate reading with rad
Pain <- sample(c("None", "Some", "Much", "Massive"), size=25, replace=TRUE)
Pain <- factor(Pain, levels=c("None", "Some", "Much", "Massive"), ordered=TRUE)
Gender <- sample(c("Male", "Female"), size=25, replace=TRUE)
```

```
Gender <- factor(Gender)
mydata <- data.frame(Pain, Gender)
rm(Pain); rm(Gender)


# ------------------------------------------
# barchart from the data for a single variable
# ------------------------------------------

# for each level of Pain, display the frequencies
# Pain is an ordered factor, so the bar colors are ordered
BarChart(Pain)
# short name
bc(Pain)
# save and then display the frequencies for later analysis
myCount <- BarChart(Pain)
myCount
# compare to standard R bar plot, which requires mydata$ reference
barplot(table(mydata$Pain))

# Gender is unordered, so one color used for all the bars
BarChart(Gender)

# specify a unique bar color for each of the two bars
BarChart(Gender, col.fill=c("palegreen3","tan"))

# automatically provide horizontal value labels
#   and adjust left margin as needed
BarChart(Pain, horiz=TRUE)


# ---------------------------------------
# barchart from the data for two variables
# ---------------------------------------

# at each level of Pain, show the frequencies of the Gender levels
BarChart(Pain, by=Gender)

# at each level of Pain, show the frequencies of the Gender levels
# set for all subsequent analyses for all lessR graphic functions
set(colors="gray")
BarChart(Pain, by=Gender)

# at each level of Gender, show the frequencies of the Pain levels
# Pain levels are ordered, so the corresponding colors are also ordered
# color theme set to red with ghost effects
set(colors="red", ghost=TRUE)
BarChart(Gender, by=Pain)
set(colors="sienna")

# specify an ordered blue palette of colors for the ordered levels of Pain
# only works when the variable is an ordered factor
# colors can be named or customized with rgb function
```

```
BarChart(Gender, by=Pain, col.low="azure", col.hi=rgb(100,110,200,max=255))

# define custom color gradient within each group of bars
# applies to both ordered and unordered factors
BarChart(Gender, by=Pain, col.fill=c("thistle1","thistle2","thistle3","thistle4"))

# display bars beside each other instead of stacked, Female and Male
# the levels of Pain are included within each respective bar
BarChart(Gender, by=Pain, beside=TRUE, legend.horiz=TRUE)

# horizontal bar chart of two variables, put legend on the top
BarChart(Gender, by=Pain, horiz=TRUE, legend.loc="top")

# many options, including those from par: col.main, col.axis, col.lab, cex.lab
# for more info on these graphic options, enter:  help(par)
BarChart(Pain, by=Gender, col.fill=c("coral3","seagreen3"),
  legend.loc="topleft", legend.labels=c("Girls", "Boys"),
  xlab="Pain Level", main="Gender for Different Pain Levels",
  col.bg="wheat1", col.grid="wheat3", col.main="wheat4",
  col.axis="wheat4", col.lab="wheat4", cex.lab=1.2)


# ------------------------------------------
# multiple bar charts across multiple variables
# ------------------------------------------

# bar charts for all non-numeric variables in the data frame called mydata
#   and all numeric variables with a small number of values, < n.cat
BarChart()

# specify a list of variables
mydata <- Read("Employee", format="lessR", quiet=TRUE)
BarChart(c(Dept, Gender))


# ---------------------------
# can enter many types of data
# ---------------------------

# generate and enter integer data
X1 <- sample(1:4, size=100, replace=TRUE)
X2 <- sample(1:4, size=100, replace=TRUE)
BarChart(X1)
BarChart(X1,X2)

# generate and enter type double data
X1 <- sample(c(1,2,3,4), size=100, replace=TRUE)
X2 <- sample(c(1,2,3,4), size=100, replace=TRUE)
BarChart(X1)
BarChart(X1, by=X2)

# generate and enter character string data
# that is, without first converting to a factor
```

```
Travel <- sample(c("Bike", "Bus", "Car", "Motorcycle"), size=25, replace=TRUE)
BarChart(Travel, horiz=TRUE)


# ------------------------------
# bar chart directly from counts
# ------------------------------

# barchart of one variable with three levels
# enter counts as a vector with the combine function, c
# must supply the level names and variable name
City <- c(206, 94, 382)
names(City) <- c("LA","Chicago","NY")
BarChart(City, main="Employees in Each City", addtop=10)

# barchart of two variables
# two Quality levels, the rows
# three Supplier levels, the columns
# enter counts row by row, then form the table with rbind function
# must supply the level (value) names and the variable names
# chart presented as Row Variable, analyzed at each level of Column Variable
row1 <- c(19, 16, 23)
row2 <- c(6, 6, 8)
mytable <- rbind(row1, row2)
rownames(mytable) <- c("Pass", "Defective")
colnames(mytable) <- c("Acme, Inc", "Nuts, Inc", "Bolts, Inc")
BarChart(mytable, xlab="Supplier", legend.title="Quality")

# counts are in the data file to be read directly
mydata <- read.csv(text="
Dept, Count
ACCT,5
ADMN,6
FINC,4
MKTG,6
SALE,15", header=TRUE)
# use count.levels to indicate the label for each corresponding count
BarChart(Count, count.levels=Dept)
```

BoxPlot *Boxplot*

Description

Abbreviation: bx

Uses the standard R boxplot function, boxplot to display a boxplot in color. Also display the relevant statistics such as the hinges, median and IQR.

If the provided object to analyze is a set of multiple variables, including an entire data frame, then each non-numeric variable in the data frame is analyzed and the results written to a pdf file in the

current working directory. The name of each output pdf file that contains a bar chart and its path are specified in the output.

Usage

```
BoxPlot(x=NULL, data=mydata, n.cat=getOption("n.cat"),

        col.fill=getOption("col.fill.bar"),
        col.stroke=getOption("col.stroke.bar"),
        col.bg=getOption("col.bg"),
        col.grid=getOption("col.grid"),

        cex.axis=.85, col.axis="gray30", col.ticks="gray30",
        xlab=NULL, main=NULL, digits.d=NULL,

        horiz=TRUE, add.points=FALSE,

        quiet=getOption("quiet"),
        pdf.file=NULL, pdf.width=5, pdf.height=5, ...)

bx(...)
```

Arguments

x	Variable(s) to analyze. Can be a single numerical variable, either within a data frame or as a vector in the user's workspace, or multiple variables in a data frame such as designated with the c function, or an entire data frame. If not specified, then defaults to all numerical variables in the specified data frame, mydata by default.
data	Optional data frame that contains the variable(s) of interest, default is mydata.
n.cat	For the analysis of multiple variables, such as a data frame, specifies the largest number of unique values of variable of a numeric data type for which the variable will be analyzed as a categorical. Set to 0 to turn off.
col.fill	Color of the box.
col.stroke	Color of any points that designate outliers. By default this is the same color as the box.
col.bg	Color of the plot background.
col.grid	Color of the grid lines.
cex.axis	Scale magnification factor, which by defaults displays the axis values to be smaller than the axis labels. Provides the functionality of, and can be replaced by, the standard R cex.axis.
col.axis	Color of the font used to label the axis values.
col.ticks	Color of the ticks used to label the axis values.
xlab	Label for the value axis, which defaults to the variable's name.
main	Title of graph.

digits.d	Number of decimal digits displayed in the listing of the summary statistics.
horiz	Orientation of the boxplot. Set FALSE for vertical.
add.points	If TRUE, then place a dot plot (i.e., stripchart) over the box plot.
quiet	If set to TRUE, no text output. Can change system default with set function.
pdf.file	Name of the pdf file to which graphics are redirected. If there is no filetype of .pdf, the filetype is added to the name.
pdf.width	Width of the pdf file in inches.
pdf.height	Height of the pdf file in inches.
...	Other parameter values for graphics as defined processed by boxplot and bxp such as whiskcol for the whisker color, etc. and par, including ylim to set the limits of the value axis, lwd for the line width, cex.lab for the size of the label, col.main for the title, etc.

Details

OVERVIEW

Unlike the standard R boxplot function, boxplot, the default here is for a horizontal boxplot. Also, BoxPlot does not currently process in formula mode, so use the standard R boxplot function to process a formula in which a boxplot is displayed for a variable at each level of a second, usually categorical, variable.

Other graphic parameters are available to format the display, such as main for the title, and other parameters found in boxplot and par. To minimize white space around the boxplot, re-size the graphics window before or after creating the boxplot.

DATA

The data may either be a vector from the global environment, the user's workspace, as illustrated in the examples below, or one or more variable's in a data frame, or a complete data frame. The default input data frame is mydata. Can specify the source data frame name with the data option. If multiple variables are specified, only the numerical variables in the list of variables are analyzed. The variables in the data frame are referenced directly by their names, that is, no need to invoke the standard R mechanisms of the mydata$name notation, the with function or the attach function. If the name of the vector in the global environment and of a variable in the input data frame are the same, the vector is analyzed.

To obtain a box plot of each numerical variable in the mydata data frame, use BoxPlot(). Or, for a data frame with a different name, insert the name between the parentheses. To analyze a subset of the variables in a data frame, specify the list with either a : or the c function, such as m01:m03 or c(m01,m02,m03).

COLORS

Individual colors in the plot can be manipulated with options such as col.fill for the color of the box. A color theme for all the colors can be chosen for a specific plot with the colors option with the lessR function set. The default color theme is blue, but a gray scale is available with "gray", and other themes are available as explained in set, such as "red" and "green". Use the option ghost=TRUE for a black background, no grid lines and partial transparency of plotted colors.

VARIABLE LABELS

If variable labels exist, then the corresponding variable label is by default listed as the label for the horizontal axis and on the text output. For more information, see Read.

PDF OUTPUT
Because of the customized graphic windowing system that maintains a unique graphic window for the Help function, the standard graphic output functions such as pdf do not work with the lessR graphics functions. Instead, to obtain pdf output, use the pdf.file option, perhaps with the optional pdf.width and pdf.height options. These files are written to the default working directory, which can be explicitly specified with the R setwd function.

ONLY VARIABLES ARE REFERENCED
The referenced variable in a lessR function can only be a variable name. This referenced variable must exist in either the referenced data frame, mydata by default, or in the user's workspace, more formally called the global environment. That is, expressions cannot be directly evaluated. For example:

```
> BoxPlot(rnorm(50))   # does NOT work
```

Instead, do the following:

```
> Y <- rnorm(50)   # create vector Y in user workspace
> BoxPlot(Y)     # directly reference Y
```

Value

Based on the standard R function boxplot, invisibly returns a list with components described in boxplot.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

References

Gerbing, D. W. (2013). R Data Analysis without Programming, Chapter 5, NY: Routledge.

See Also

boxplot, bxp, par, set.

Examples

```
# simulate data and get at least one outlier
y <- rnorm(100,50,10)
y[1] <- 90


# ------------------------------
# box plot for a single variable
# ------------------------------

# standard horizontal boxplot with all defaults
BoxPlot(y)

# short name
bx(y)
```

```
# save the box plot to a pdf file
BoxPlot(y, pdf.file="MyBoxPlot.pdf")

# vertical boxplot with plum color
BoxPlot(y, horiz=FALSE, col.fill="plum")

# box plot with outliers more strongly highlighted
BoxPlot(y, col.stroke="red", xlab="My Variable")


# ------------------------------------------------
# box plots for data frames and multiple variables
# ------------------------------------------------

# create data frame, mydata, to mimic reading data with rad function
# mydata contains both numeric and non-numeric data
mydata <- data.frame(rnorm(100), rnorm(100), rnorm(100), rep(c("A","B"),50))
names(mydata) <- c("X","Y","Z","C")

# box plot for variable X from data frame, referred to directly
BoxPlot(X)

# box plot with superimposed dot plot (stripchart)
BoxPlot(X, dotplot=TRUE)

# box plots for all numeric variables in data frame called mydata
BoxPlot()

# box plots for all numeric variables in data frame called mydata
#  with specified options
BoxPlot(col.fill="palegreen1", col.stroke="plum")

# Use the subset function to specify a variable list
# box plots for all specified numeric variables
BoxPlot(c(X,Y))
```

---

corCFA *Confirmatory Factor Analysis of a Multiple Indicator Measurement Model*

Description

Abbreviation: `cfa`

A multiple indicator measurement model partitions a set of indicators, such as items on a survey, into mutually exclusive groups with one common factor per group of indicators. From the input correlation matrix of the indicator variables, this procedure uses iterated centroid estimation to estimate the coefficients of the model, the factor pattern and factor-factor correlations, as well as the correlations of each factor with each indicator. The analysis is provided by Fortran code, adapted from John Hunter's (1969) program PACKAGE (Hunter and Cohen, 1969).

Corresponding scale reliabilities are provided, as well as the residuals, the difference between the indicator correlations and those predicted by the model. To visualize the relationships, a heat map of the re-ordered correlation matrix is also provided, with indicator communalities in the diagonal. To understand the meaning of each factor, the corresponding indicator content is displayed for each factor if the indicators have been read as variable labels. Also provides the code to obtain the maximum likelihood solution of the corresponding multiple indicator measurement model (MIMM) with the `cfa` function from the `lavaan` package.

The `scales` is a wrapper, which retains 1's in the diagonal of the indicator correlation matrix, so provides scale reliabilities and observed indicator-scale and scale-scale correlations.

Usage

```
corCFA(x=mycor, data=mydata,

         labels=c("include", "exclude", "only"),

         iter=50, resid=TRUE, item.cor=TRUE, sort=TRUE,

         main=NULL, heat.map=TRUE, bottom=3, right=3,

         pdf.file=NULL, pdf.width=5, pdf.height=5,

         F1=NULL, F2=NULL, F3=NULL, F4=NULL, F5=NULL,
         F6=NULL, F7=NULL, F8=NULL, F9=NULL, F10=NULL,
         F11=NULL, F12=NULL)

cfa(...)

scales(..., iter=0, resid=FALSE, item.cor=FALSE, sort=FALSE, heat.map=FALSE)
```

Arguments

`x`	Correlation matrix to be analyzed.
`data`	Data frame of the original data to be checked for any variable labels, usually indicator content. This is not to calculate correlations, which is separately provided for by the `lessR` function `Correlation`.
`labels`	If "include" or "exclude" then variable labels are displayed or not, organized by the items within each factor. If "only" then no data analysis performed, only the display of the labels by factor.
`iter`	Number of iterations for communality estimates.
`resid`	If `TRUE`, then calculate and print the residuals.
`item.cor`	If `TRUE`, display the indicator correlations.
`sort`	If `TRUE`, re-order the output correlation matrix so that indicators within each factor are sorted by their factor loadings on their own factor.
`main`	Graph title of heat map. Set to `main=""` to turn off.
`heat.map`	If `TRUE`, display a heat map of the indicator correlations with indicator communalities in the diagonal.

bottom	Number of lines of bottom margin of heat map.
right	Number of lines of right margin of heat map.
pdf.file	Name of the pdf file to which graphics are redirected.
pdf.width	Width of the pdf file in inches.
pdf.height	Height of the pdf file in inches.
F1	Variables, listed by ordinal position, that define Factor 1.
F2	Variables, listed by ordinal position, that define Factor 2.
F3	Variables, listed by ordinal position, that define Factor 3.
F4	Variables, listed by ordinal position, that define Factor 4.
F5	Variables, listed by ordinal position, that define Factor 5.
F6	Variables, listed by ordinal position, that define Factor 6.
F7	Variables, listed by ordinal position, that define Factor 7.
F8	Variables, listed by ordinal position, that define Factor 8.
F9	Variables, listed by ordinal position, that define Factor 9.
F10	Variables, listed by ordinal position, that define Factor 10.
F11	Variables, listed by ordinal position, that define Factor 11.
F12	Variables, listed by ordinal position, that define Factor 12.
...	Parameter values.

Details

OVERVIEW

A multiple indicator measurement model defines one or more latent variables, called factors, in terms of mutually exclusive sets of indicator variables, such as items from a questionnaire or survey. That is, each factor is defined by a unique set or group of indicators, and each indicator only contributes to the definition of one factor. Two sets of parameters are estimated by the model, the factor pattern coefficients, the lambda's, and the factor-factor correlations, the phi's. Also estimated here are the correlations of each indicator with the other factors.

INPUT

Unless `labels="only"`), the analysis requires the correlation matrix of the indicators and the specification of the groups of indicators, each of which defines a factor in the multiple indicator measurement model. The default name for the indicator correlation matrix is `mycor`, which is also the default name of the matrix produced by the `lessR` function `Correlation` that computes the correlations from the data, as well as the name of the matrix read by the `lessR` function `corRead` that reads the already computed correlation matrix from an external file.

The data frame from which the correlation matrix was computed is requried only if any associated variable labels are listed, organized by the items within each factor. By default, `labels="include"`, these labels are listed as part of the analysis if they are available.

Define the constituent variables, the indicators, of each factor with a listing of each variable by its name in the correlation matrix. Each of the up to 12 factors is named F1, F2, etc. If the specified variables of a factor are in consecutive order in the input correlation matrix, the list can be specified by listing the first variable, a colon, and then the last variable. To specify multiple variables, a single

variable or a list, separate each by a comma, then invoke the R combine or c function, preceded by the factor's name and an equals sign. For example, if the first factor is defined by variables in the input correlation matrix from m02 through m05, and the variable Anxiety, then define the factor in the corCFA function call according to F1=c(m02:m05,Anxiety).

OUTPUT
The result of the analysis is the correlation matrix of the indicator variables and resulting factors, plus the reliability analysis of the observed total scores or scale that corresponds to each factor. Each scale is defined as an unweighted composite. The corresponding code to analyze the model with the cfa function from the lavaan package is also provided with the default maximum likelihood estimation procedure. The comparable lavaan solution appears in the column that represents the fully standardized solution, factors and indicators, Std.all, the last column of the solution output.

VARIABLE LABELS
To display the indicator content, first read the indicators as variable labels with the lessR function Read. If this labels data frame exists, then the corresponding variable labels, such as the actual items on a survey, are listed by factor. For more information, see Read.

HEAT MAP
To help visualize the overall pattering of the correlations, the corresponding heat map of the item correlation matrix with communalities is produced when heat.map=TRUE, the default. As is true of the output correlation matrix, the correlations illustrated in the heat map are also sorted by their ordering within each factor. The corresponding color scheme is dictated by the system setting, according to the lessR function set. The default color scheme is blue.

ESTIMATION PROCEDURE
The estimation procedure is centroid factor analysis, which defines each factor, parallel to the definition of each scale score, as the unweighted composite of the corresponding items for that scale. The latent variables are obtained by replacing the 1's in the diagonal of the indicator variable correlation matrix with communality estimates. These estimates are obtained by iterating the solution to the specified number of iterations according to iter, which defaults to 50.

A communality is the percentage of the item's correlation attributable to, in this situation of a multiple indicator measurement model, its one underlying factor. As such, the communality is comparable to the item correlations for items within the same factor, which are also due only to the influence of the one common, underlying factor. A value of 0 for iter implies that the 1's remain in the observed variable correlation matrix, which then means that there are no latent factors defined. Instead the resulting correlation matrix is of the observed scale scores and the component items.

Value

The function invisibly returns a list of six components.
1. ff.cor: matrix of the factor correlations
2. if.cor: matrix of the indicator-factor correlations that includes the estimated pattern coefficients of the model that link a factor to its indicators
3. diag.cor: the indicator communalities
4. alpha: coefficient alpha for each set of indicators
5. omega: if a factor analysis with communality estimates (iter > 0), contains coefficient omega for each set of indicators
6. resid: matrix of raw indicator residuals defined as the observed correlation minus that predicted by the model and its estimates

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

References

Gerbing, D. W. (2013). R Data Analysis without Programming, Chapter 11, NY: Routledge.

Gerbing, D. W., & Hamilton, J. G. (1994). The surprising viability of a simple alternate estimation procedure for the construction of large-scale structural equation measurement models. Structural Equation Modeling: A Multidisciplinary Journal, 1, 103-115.

Hunter, J. E., Gerbing, D. W., & Boster, F. J. (1982). Machiavellian beliefs and personality: The construct invalidity of the Machiavellian dimension. Journal of Personality and Social Psychology, 43, 1293-1305.

Hunter, J. and Cohen, J. (1969). PACKAGE: A system of computer routines for the analysis of correlational data. Educational and Psychological Measurement, 1969, 29, 697-700.

Yves Rosseel (2012). lavaan: An R Package for Structural Equation Modeling. Journal of Statistical Software, 48(2), 1-36. URL http://www.jstatsoft.org/v48/i02/.

See Also

Correlation.

Examples

```
# perfect input correlation matrix for two-factor model
# Population Factor Pattern of the 3 items for each respective
#   Factor: 0.8, 0.6, 0.4
# Population Factor-Factor correlation: 0.3
mycor <- matrix(nrow=6, ncol=6, byrow=TRUE,
c(1.000,0.480,0.320,0.192,0.144,0.096,
  0.480,1.000,0.240,0.144,0.108,0.072,
  0.320,0.240,1.000,0.096,0.072,0.048,
  0.192,0.144,0.096,1.000,0.480,0.320,
  0.144,0.108,0.072,0.480,1.000,0.240,
  0.096,0.072,0.048,0.320,0.240,1.000))
colnames(mycor) <- c("X1", "X2", "X3", "X4", "X5", "X6")
rownames(mycor) <- colnames(mycor)

# the confirmatory factor analysis
# first three variables with first factor, last three with second
# default correlation matrix is mycor
# two ways to specify the items, with a colon and with commas
corCFA(F1=X1:X3, F2=c(X4,X5,X6))

# abbreviated form, one factor definition per line
cfa(
  F1=X1:X3,
  F2=X4:X6
)

# component analysis, show observed scale correlations
```

```
scales(F1=X1:X3, F2=X4:X6)

# produce a gray scale heat map of the item correlations
#   with communalities in the diagonal
# all subsequent graphics are in gray scale until changed
set(colors="gray")
corCFA(F1=X1:X3, F2=X4:X6)

# access the solution directly by saving to an object called fit
fit <- corCFA(F1=X1:X3, F2=c(X4,X5,X6))
fit
# get the pattern coefficients from the communalties
lambda <- sqrt(fit$diag.cor)
lambda

# access the lessR data set called datMach4, with variable labels
mydata <- Read("Mach4", format="lessR", quiet=TRUE)
# reverse score the reflected variables
mydata <- Recode(c(m03,m04,m06,m07,m09,m10,m11,m14,m16,m17,m19),
                 old=0:5, new=5:0, quiet=TRUE)
# calculate the correlations and store in mycor
mycor <- cr(m01:m20)
# confirmatory factor analysis of 4-factor solution of Mach IV scale
# Hunter, Gerbing and Boster (1982)
corCFA(F1=c(m06,m07,m09,m10), F2=c(m15,m02),
       F3=c(m04,m11,m16), F4=c(m01,m05,m12,m13))

# bad fitting model to illustrate indicator diagnostics
mycor <- corReflect(vars=c(m20))
corCFA(F1=c(m06,m09,m19), F2=c(m07),
       F3=c(m04,m11,m16), F4=c(m20,m18,m12,m15))
```

---

corEFA *Exploratory Factor Analysis and Multiple Indicator Measurement Model*

Description

Abbreviation: `efa`

A maximum likelihood exploratory factor analysis, provided by the standard R exploratory factor analysis `factanal`, which requires the specified number of factors as an input to the analysis. Then constructs the code to run the corresponding multiple indicator measurement model (MIMM) suggested by the exploratory factor analysis loadings in terms of both the `lessR` `corCFA` and the `cfa` function from the `lavaan` package.

Usage

```
corEFA(x=mycor, n.factors, rotate=c("promax", "varimax"),
          min.loading=.2, show.initial=FALSE, sort=TRUE, ...)
```

```
efa(...)
```

Arguments

x	Correlation matrix.
n.factors	Number of factors.
rotate	Rotation method.
min.loading	Minimum loading to include in suggested factor for confirmatory analysis and for the display of the loadings for the exploratory analysis. To ignore, set to NA.
show.initial	If TRUE, then display initial factor extraction before rotation.
sort	Sort the input variables by their highest factor loadings (but only first just list those items with loadings larger than 0.5).
...	Parameter values.

Details

Only the loadings from the exploratory factor analysis are provided, with either an oblique (promax), by default, or an orthogonal (varimax) rotation. If more information is desired, run factanal directly.

Also provides the associated multiple indicator measurement model suggested by the exploratory factor analysis. Each MIMM factor is defined by the items that have the highest loading on the corresponding exploratory factor.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

References

Gerbing, D. W. (2013). R Data Analysis without Programming, Chapter 11, NY: Routledge.

Yves Rosseel (2012). lavaan: An R Package for Structural Equation Modeling. Journal of Statistical Software, 48(2), 1-36. URL http://www.jstatsoft.org/v48/i02/.

See Also

Correlation.

Examples

```
# input correlation matrix of perfect two-factor model
# Factor Pattern for each Factor: 0.8, 0.6, 0.4
# Factor-Factor correlation: 0.3
mycor <- matrix(nrow=6, ncol=6, byrow=TRUE,
c(1.000,0.480,0.320,0.192,0.144,0.096,
  0.480,1.000,0.240,0.144,0.108,0.072,
  0.320,0.240,1.000,0.096,0.072,0.048,
  0.192,0.144,0.096,1.000,0.480,0.320,
```

```
  0.144,0.108,0.072,0.480,1.000,0.240,
  0.096,0.072,0.048,0.320,0.240,1.000))
colnames(mycor) <- c("X1", "X2", "X3", "X4", "X5", "X6")
rownames(mycor) <- colnames(mycor)

# default factor analysis of default correlation matrix mycor
#   with two factors extracted
corEFA(n.factors=2)

# abbreviated form
# use all items to construct the MIMM, regardless of their loadings
# show the initial factor extraction
efa(n.factors=2, min.loading=NA, show.initial=TRUE)
```

---

corProp *Proportionality Coefficients from Correlations*

Description

Abbreviation: prop

In the population, indicators of the same factor or latent variable have parallel correlations with all other variables. Of course, in the presence of sampling error, this parallelism will only be approximate. To assess this parallelism, proportionality coefficients are computed for each pair of variables in the input correlation matrix. Also output is a heat map of the resulting matrix of proportionality coefficients. Each graph is based on a default color theme. The original default is "blue", but other color palettes can be generated as well.

Usage

```
corProp(x=mycor,
          main=NULL, heat.map=TRUE, bottom=3, right=3,
          pdf.file=NULL, pdf.width=5, pdf.height=5)

prop(...)
```

Arguments

x	Correlation matrix.
main	Graph title. Set to main="" to turn off.
heat.map	If TRUE, display a heat map of the item correlations with the diagonal ignored.
bottom	Number of lines of bottom margin.
right	Number of lines of right margin.
pdf.file	Name of the pdf file to which graphics are redirected.
pdf.width	Width of the pdf file in inches.
pdf.height	Height of the pdf file in inches.
...	Parameter values.

Details

Proportionality coefficients indicate the extent of proportionality between two variables. Perfect proportionality of two variables is consistent with both variables being indicators of the same latent variable or factor and indicators of no other factor.

In the current version the diagonal of the input correlation matrix is ignored. To maintain parallelism, the diagonal element of 1.00 would need to be replaced the corresponding communalities, which first requires a factor analysis.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

References

Gerbing, D. W. (2013). R Data Analysis without Programming, Chapter 11, NY: Routledge.

See Also

Correlation.

Examples

```
# input correlation matrix of perfect two-factor model
# Factor Pattern for each Factor: 0.8, 0.6, 0.4
# Factor-Factor correlation: 0.3
mycor <- matrix(nrow=6, ncol=6, byrow=TRUE,
c(1.000,0.480,0.320,0.192,0.144,0.096,
  0.480,1.000,0.240,0.144,0.108,0.072,
  0.320,0.240,1.000,0.096,0.072,0.048,
  0.192,0.144,0.096,1.000,0.480,0.320,
  0.144,0.108,0.072,0.480,1.000,0.240,
  0.096,0.072,0.048,0.320,0.240,1.000))
colnames(mycor) <- c("X1", "X2", "X3", "X4", "X5", "X6")
rownames(mycor) <- colnames(mycor)

# proportionality coefficients of correlation matrix mycor
# indicators of the same factor have proportional correlations
corProp()

# abbreviated form
prop()

# calculate and store proportionality coefficients in myprop
# order the proportionality coefficients to help identify factors
myprop <- corProp()
corReorder(myprop)
```

corRead *Read Specified Correlation Matrix*

Description

Abbreviation: rd.cor

Read a correlation matrix into R. The resulting matrix is named mycor. All coefficients for each variable must be on one row. No variable names are in the file to be read.

Usage

```
corRead(ref=NULL, names=NULL)

rd.cor(...)
```

Arguments

ref	File reference, either omitted to browse for the data file, or a full path name or web URL, included in quotes. A URL begins with http://.
names	The names of the variables in the matrix.
...	Parameter values.

Details

Read a correlation, or any square, matrix into R. The resulting matrix is named mycor. All coefficients for each variable must be on one row. No variable names are in the file to be read. The coefficients within each row, that is, for a single variable, are delimited by a white space, such as one or more blanks.

The standard R function used to read the matrix is read.table.

By default the variables are named V1, V2, etc. If the names option is invoked, then the specified names are attached to the respective rows and columns of the matrix. Here it may be convenient to name the variables with the lessR function to.

The alternative is to calculate the correlations from the data, such as with the lessR function Correlation or the standard R function cor.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

References

Gerbing, D. W. (2013). R Data Analysis without Programming, Chapter 8, NY: Routledge.

See Also

Correlation, read.table.

Examples

```
# browse for the data file because ref is omitted
# name the variables with the lessR function to
# corRead(names=to("m",20))

# abbreviated form
# read a matrix with 4 variables and specify the names
# rd.cor(names=c("m06","m07","m09","m10"))
```

`corReflect` *Reflect Specified Variables in a Correlation Matrix*

Description

Abbreviation: `reflect`

Reflects the specified variables by multiplying each correlation of the variable by -1. Usually a prelude to a factor analysis, such as provided by `corCFA`.

Usage

```
corReflect(x=mycor, vars,
          main=NULL, heat.map=TRUE, bottom=3,right=3,
          pdf.file=NULL, pdf.width=5, pdf.height=5)

reflect(...)
```

Arguments

`x`	Correlation matrix.
`vars`	List of the re-ordered variables, each variable listed by its ordinal position in the input correlation matrix.
`main`	Graph title. Set to `main=""` to turn off.
`heat.map`	If `TRUE`, display a heat map of the item correlations with item communalities in the diagonal.
`bottom`	Number of lines of bottom margin.
`right`	Number of lines of right margin.
`pdf.file`	Name of the pdf file to which graphics are redirected.
`pdf.width`	Width of the pdf file in inches.
`pdf.height`	Height of the pdf file in inches.
`...`	Parameter values.

Details

Reflects the specified variables by multiplying each correlation of the variable by -1. The original data from which the correlations are computed is unmodified unless the output of the function is written into the input correlation matrix, by default `mycor`.

Define the constituent variables, the items, with a listing of each variable by its name in the correlation matrix. If the specified variables are in consecutive order in the input correlation matrix, the list can be specified by listing the first variable, a colon, and then the last variable. To specify multiple variables, a single variable or a list, separate each by a comma, then invoke the R combine or `c` function. For example, if the list of variables in the input correlation matrix is from m02 through m05, and the variable Anxiety, then define the list in the `corReflect` function call according to `vars=c(m02:m05,Anxiety)`.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

See Also

`Correlation`, `Recode`.

Examples

```
# input correlation matrix of perfect two-factor model
# Factor Pattern for each Factor: 0.8, 0.6, 0.4
# Factor-Factor correlation: 0.3
mycor <- matrix(nrow=6, ncol=6, byrow=TRUE,
c(1.000,0.480,0.320,0.192,0.144,0.096,
  0.480,1.000,0.240,0.144,0.108,0.072,
  0.320,0.240,1.000,0.096,0.072,0.048,
  0.192,0.144,0.096,1.000,0.480,0.320,
  0.144,0.108,0.072,0.480,1.000,0.240,
  0.096,0.072,0.048,0.320,0.240,1.000))
colnames(mycor) <- c("V1", "V2", "V3", "V4", "V5", "V6")
rownames(mycor) <- colnames(mycor)

# reflect all 3 indicators of the second factor
corReflect(vars=c(V4,V5,V6))

# abbreviated form
reflect(vars=c(V4,V5,V6))
```

`Correlation` *Correlation Analysis*

Description

Abbreviation: `cr, cr.brief`

For two variables the correlation coefficient with hypothesis test and confidence interval, or for a data frame, or list of variables from a data frame, the correlation matrix. The default computed coefficient(s) are the standard Pearson's product-moment correlation. For the default missing data technique of pairwise deletion, an analysis of missing data for each computed correlation coefficient is provided. For a correlation matrix a statistical summary of the missing data across all cells is provided.

Usage

```
Correlation(x, y, data=mydata,
         miss=c("pairwise", "listwise", "everything"),
         show.n=NULL, brief=FALSE, n.cat=getOption("n.cat"),
         digits.d=NULL, graphics=FALSE,
         main=NULL, bottom=3, right=3,
         pdf=FALSE, pdf.width=5, pdf.height=5, ...)

cr.brief(..., brief=TRUE)

cr(...)
```

Arguments

`x`	First variable, or list of variables for a correlation matrix.
`y`	Second variable or not specified if the first argument is a list.
`data`	Optional data frame that contains the variables of interest, default is `mydata`.
`miss`	Basis for deleting missing data values.
`show.n`	For pairwise deletion, show the matrix of sample sizes for each correlation coefficient, regardless of sample size.
`brief`	Pretains to a single correlation coefficient analysis. If `FALSE`, then the sample covariance and number of non-missing and missing observations are displayed.
`n.cat`	When analyzing all the variables in a data frame, specifies the largest number of unique values of variable of a numeric data type for which the variable will be analyzed as a categorical. Set to 0 to turn off.
`digits.d`	Specifies the number of decimal digits to display in the output.
`graphics`	If `TRUE`, generate a scatter plot matrix and heat map.
`main`	Graph title of heat map. Set to `main=""` to turn off.
`bottom`	Number of lines of bottom margin of heat map.
`right`	Number of lines of right margin of heat map.
`pdf`	If `TRUE`, generate scatter plot matrix and heat map and write to pdf files.
`pdf.width`	Width of the pdf file in inches.
`pdf.height`	Height of the pdf file in inches.
`...`	Other parameter values for internally called functions, which include `method="spearman"` and `method="kendall"` and also `alternative="less"` and `alternative="more"`.

Details

When two variables are specified, both x and y, the output is the correlation coefficient with hypothesis test, for a null hypothesis of 0, and confidence interval. Also displays the sample covariance. Based on R functions cor, cor.test, cov.

In place of two variables x and y, x can be a complete data frame, either specified with the name of a data frame, or blank to rely upon the default data frame mydata. Or, x can be a list of variables from the input data frame. In these situations y is missing. Any non-numeric variables in the data frame or specified variable list are automatically deleted from the analysis.

Set graphics=TRUE to generate a heat map and scatter plot matrix to standard graphics windows. Set pdf=TRUE to generate these graphics but have them directed to their respective pdf files.

For treating missing data, the default is pairwise, which means that an observation is deleted only for the computation of a specific correlation coefficient if one or both variables are missing the value for the relevant variable(s). For listwise deletion, the entire observation is deleted from the analysis if any of its data values are missing. For the more extreme everything option, any missing data values for a variable result in all correlations for that variable reported as missing.

Value

If a correlation matrix is computed, the matrix is returned. Usually assign the name of mycor to the output matrix, as in following examples. This matrix is ready for input into any of the lessR functions that analyze correlational data, including confirmatory factor analysis by corCFA and also exploratory factor analysis, either the standard R function factanal or the lessR function corEFA

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

References

Gerbing, D. W. (2013). R Data Analysis without Programming, Chapter 8, NY: Routledge.

See Also

cor.test, cov.

Examples

```
# data
n <- 12
f <- sample(c("Group1","Group2"), size=n, replace=TRUE)
x1 <- round(rnorm(n=n, mean=50, sd=10), 2)
x2 <- round(rnorm(n=n, mean=50, sd=10), 2)
x3 <- round(rnorm(n=n, mean=50, sd=10), 2)
x4 <- round(rnorm(n=n, mean=50, sd=10), 2)
mydata <- data.frame(f,x1, x2, x3, x4)
rm(f); rm(x1); rm(x2); rm(x3); rm(x4)

# correlation and covariance
Correlation(x1, x2)
```

```
# short name
cr(x1, x2)
# brief form of output
cr.brief(x1, x2)

# Spearman rank correlation, one-sided test
Correlation(x1, x2, method="spearman", alternative="less")

# correlation matrix of the numerical variables in mycor
mycor <- Correlation()

# correlation matrix of Kendall's tau coefficients
mycor <- cr(method="kendall")

# correlation matrix of specified variables in mycor with graphics
mycor <- Correlation(x1:x3, graphics=TRUE)

# analysis with data not from data frame mycor
data(attitude)
mycor <- Correlation(rating, learning, data=attitude)

# analysis of entire data frame that is not mycor
data(attitude)
mycor <- Correlation(attitude)
```

`corReorder` *Reorder Variables in a Correlation Matrix*

Description

Abbreviation: `reord`

Re-arranges the order of the variables in the input correlation matrix. If no variable list is specified then the variables are re-ordered according to first providing the variable with the largest sum of squared correlations of all the variables, then the variable that has the highest correlation with the first variable, and so forth.

Usage

```
corReorder(x=mycor, vars=NULL, first=0,
          heat.map=TRUE, main=NULL, bottom=3,right=3,
          pdf.file=NULL, pdf.width=5, pdf.height=5)

reord(...)
```

Arguments

`x`	Correlation matrix.
`vars`	List of the re-ordered variables, each variable listed by its ordinal position in the input correlation matrix.

first	The first variable listed in the ordered matrix.
main	Graph title. Set to main="" to turn off.
heat.map	If TRUE, display a heat map of the item correlations with item communalities in the diagonal.
bottom	Number of lines of bottom margin.
right	Number of lines of right margin.
pdf.file	Name of the pdf file to which graphics are redirected.
pdf.width	Width of the pdf file in inches.
pdf.height	Height of the pdf file in inches.
...	Parameter values.

Details

Reorder and/or delete variables in the input correlation matrix.

Define the constituent variables, the items, with a listing of each variable by its name in the correlation matrix. If the specified variables are in consecutive order in the input correlation matrix, the list can be specified by listing the first variable, a colon, and then the last variable. To specify multiple variables, a single variable or a list, separate each by a comma, then invoke the R combine or c function. For example, if the list of variables in the input correlation matrix is from m02 through m05, and the variable Anxiety, then define the list in the corReorder function call according to vars=c(m02:m05,Anxiety).

Or, define the ordering of the variables according to the following algorithm. If no variable list is specified then the variables are re-ordered such that the first variable is that which has the largest sum of squared correlations of all the variables, then the variable that has the highest correlation with the first variable, and so forth.

In the absence of a variable list, the first variable in the re-ordered matrix can be specified with the first option.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

See Also

Correlation.

Examples

```
# input correlation matrix of perfect two-factor model
# Factor Pattern for each Factor: 0.8, 0.6, 0.4
# Factor-Factor correlation: 0.3
mycor <- matrix(nrow=6, ncol=6, byrow=TRUE,
c(1.000,0.480,0.320,0.192,0.144,0.096,
  0.480,1.000,0.240,0.144,0.108,0.072,
  0.320,0.240,1.000,0.096,0.072,0.048,
  0.192,0.144,0.096,1.000,0.480,0.320,
  0.144,0.108,0.072,0.480,1.000,0.240,
```

```
  0.096,0.072,0.048,0.320,0.240,1.000))
colnames(mycor) <- c("V1", "V2", "V3", "V4", "V5", "V6")
rownames(mycor) <- colnames(mycor)

# leave only the 3 indicators of the second factor
#  in reverse order
corReorder(vars=c(V6,V5,V4))

# reorder the variables according to the ordering algorithm
corReorder()

# abbreviated form
# retain only the first three variables
reord(vars=c(V1:V3))

# reorder the variables according to the ordering algorithm
#  with the 4th variable listed first
corReorder(first=4)
```

---

corScree *Eigenvalue Plot of a Correlation Matrix*

Description

Abbreviation: scree

Plots the successive eigenvalues of an input correlation matrix. Also plots the successive differences of the eigenvalues. The purpose is usually to help determine the number of factors that explain the correlations in a correlation matrix. So usually a prelude to an exploratory factor analysis, such as provided by the lessR function corEFA. This program relies upon the standard R exploratory factor analysis factanal, which requires the specified number of factors as an input to the analysis.

Usage

```
corScree(x=mycor,
          main=NULL, pdf=FALSE, pdf.width=5, pdf.height=5, ...)

scree(...)
```

Arguments

x	Correlation matrix.
main	Graph title, which is blank by default.
pdf	Indicator as to if the graphic files should be saved as pdf files instead of directed to the standard graphics windows.
pdf.width	Width of the pdf file in inches.
pdf.height	Height of the pdf file in inches.
...	Parameter values.

Details

Interpretation of the scree plot to assist in the assessment of the number of factors that account for the structure of a correlation matrix depends primarily on the analysis of the differences between the successive eigenvalues. The differences begin to diminish where the "scree" begins, analogous to the debris that falls off of a hill top. Accordingly both the scree plot itself, the plot of the successive eigenvalues, and the plot of the differences of the successive eigenvalues are presented.

PDF OUTPUT
Because of the customized graphic windowing system that maintains a unique graphic window for the Help function, the standard graphic output functions such as pdf do not work with the lessR graphics functions. Instead, to obtain pdf output, use the pdf.file option, perhaps with the optional pdf.width and pdf.height options. These files are written to the default working directory, which can be explicitly specified with the R setwd function.

If the two plots, of the population and sample distributions respectively, are written to pdf files, according to pdf=TRUE, they are named Scree.pdf and ScreeDiff.pdf. Their names and the directory to which they are written are provided as part the console output.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

See Also

Correlation.

Examples

```
# input correlation matrix of perfect two-factor model
# Factor Pattern for each Factor: 0.8, 0.6, 0.4
# Factor-Factor correlation: 0.3
mycor <- matrix(nrow=6, ncol=6, byrow=TRUE,
c(1.000,0.480,0.320,0.192,0.144,0.096,
  0.480,1.000,0.240,0.144,0.108,0.072,
  0.320,0.240,1.000,0.096,0.072,0.048,
  0.192,0.144,0.096,1.000,0.480,0.320,
  0.144,0.108,0.072,0.480,1.000,0.240,
  0.096,0.072,0.048,0.320,0.240,1.000))
colnames(mycor) <- c("V1", "V2", "V3", "V4", "V5", "V6")
rownames(mycor) <- colnames(mycor)

# obtain the scree plots
corScree()

# abbreviated form
scree()
```

CountAll *CountAll Descriptive Analysis of all Variables in the Data Frame*

Description

Automatically call the following functions in this package: SummaryStats, Histogram and BarChart. The result is set of summary statistics for every variable in the data frame, by default called mydata, a histogram for each numerical variable and a bar chart for each categorical variable.

Usage

```
CountAll(x=mydata, ...)

ca(...)
```

Arguments

x	Data frame that contains the variables to analyze, by default mydata.
...	Other parameter values for graphics.

Details

CountAll is designed to work in conjunction with the lessR function Read, which reads a csv or other formatted data file into the data frame mydata. All the bar charts and associated summary statistics are written to one file and all the histograms and associated summary statistics for the numerical variables are written to another file.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

See Also

SummaryStats, Histogram, BarChart.

Examples

```
# create data frame called mydata
n <- 12
X <- sample(c("Group1","Group2"), size=n, replace=TRUE)
Y <- rnorm(n=n, mean=50, sd=10)
mydata <- data.frame(X,Y)
rm(X); rm(Y);

# CountAll descriptive analysis of mydata
CountAll()
# short name
ca()
```

dataBodyMeas *Data: Body Measurements*

Description

Body measurements of 170 women and 170 men who purchased motorcycle clothing from Gerbing's Heated Clothing, Inc., now Gordon's Heated Clothing, LLC.

Usage

```
data(dataBodyMeas)
```

Format

A data table with 340 observations and the following 7 variables.

Gender, "M" or "F" (factor)
Weight (integer in pounds)
Height (integer in inches)
Waist (integer in inches)
Hips (integer in inches)
Chest (integer in inches)
Hand (numeric, circumference of hand in inches to nearest quarter of an inch)
Shoe (numeric, size including half sizes)

Source

author

dataCars93 *Data: Cars93*

Description

1993 New Car Data.

Usage

```
data(dataCars93)
```

Format

A data table with 93 observations and 25 variables.

Variables
———————————————-

Make: Model
Type: Small, Sporty, Compact, Midsize, Large
MinPrice: Minimum Price (in $1,000) - Price for basic version of this model
MidPrice: Midrange Price (in $1,000) - Average of Min and Max prices
MaxPrice: Maximum Price (in $1,000) - Price for a premium version
MPGcity: City MPG
MPGhiway: Highway MPG
Airbags: 0 = none, 1 = driver only, 2 = driver & passenger
DriveTrain: 0 = rear wheel drive, 1 = front wheel drive, 2 = all wheel drive
Cylinders: Number of cylinders
Engine: Engine size (liters)
HP: maximum Horsepower
RPM: revolutions per minute at maximum horsepower
RevMile: Engine revolutions per mile in highest gear
Manual: Manual transmission available, 0 = No, 1 = Yes
FuelCap: Fuel tank capacity (gallons)
PassCap: Passenger capacity (persons)
Length: Length (inches)
Wheelbase: Wheelbase (inches)
Width: Width (inches)
Uturn: U-turn space (feet)
RearSeat: Rear seat room (inches)
LugCap: Luggage capacity (cu. ft.)
Weight: Weight (pounds)
Source: 0=non-USA manufacturer, 1=USA manufacturer

Source

Lock, R. H. (1993). 1993 new car data. Journal of Statistics Education, 1(1).

`dataEmployee` *Data: Employees*

Description

Some human resource data on 37 employees with 6 variables. Variable labels are included in the data file.

Usage

```
data(dataEmployee)
```

Format

A data table with 37 observations.

Years,"Years Employed in the Company"
Gender,"Male or Female"
Dept,"Department Employed"
Salary,"Annual Salary (USD)"
Satisfaction,"Satisfaction with Work Environment"
HealthPlan,"1=GoodHealth, 2=YellowCross, 3=BestCare"

Source

author

`dataJackets` *Data: Motorcycle Type and Thickness of Jacket*

Description

Two variables, one is type of motorcycle and the other is the thickness of the purchased jacket.

Usage

```
data(dataJackets)
```

Format

A data table with 1025 observations.

Bike,"Type of Motorcycle, Honda or BMW"
Jacket,"Lite, Med or Thick"

Source

author

dataLearn *Data: Distributed vs Massed Practice*

Description

Completely Randomized design, one-factor with two levels (CR-2): One grouping variable that specifies type of learning, distributed or massed practice, and one response variable, Learning.

Usage

```
data(dataLearn)
```

Format

A data table with 34 observations.

Source

author

dataMach4 *Data: Machiavellianism*

Description

Likert data responses to Christie and Geiss's (1970) Mach~IV scale from Hunter, Gerbing and Boster (1982).

All Likert items assessed on a 6-point scale from 0: Strongly Disagree to 5: Strongly Agree. Variable labels, the item content, are included.

The following items should be reverse scored: 3, 4, 6, 7, 9, 10, 11, 14, 16, 17, 19

Usage

```
data(dataMach4)
```

Format

A data table with 351 observations.

Gender, 1 column, 0:Male, 1:Female
Mach IV, 20 Likert items: m01, m02, . . . , m20

Christie and Geiss Mach IV Scale

1. Never tell anyone the real reason you did something unless it is useful to do so.
2. The best way to handle people is to tell them what they want to hear.

3. One should take action only when sure it is morally right.
4. Most people are basically good and kind.
5. It is safest to assume that all people have a vicious streak and it will come out when they are given a chance.
6. Honesty is the best policy in all cases.
7. There is no excuse for lying to someone else.
8. Generally speaking, people won't work hard unless they're forced to do so.
9. All in all, it is better to be humble and honest than to be important and dishonest.
10. When you ask someone to do something for you, it is best to give the real reasons for wanting it rather than giving reasons which carry more weight.
11. Most people who get ahead in the world lead clean, moral lives.
12. Anyone who completely trusts anyone else is asking for trouble.
13. The biggest difference between most criminals and other people is that the criminals are stupid enough to get caught.
14. Most people are brave.
15. It is wise to flatter important people.
16. It is possible to be good in all respects.
17. Barnum was wrong when he said that there's a sucker born every minute.
18. It is hard to get ahead without cutting corners here and there.
19. People suffering from incurable diseases should have the choice of being put painlessly to death.
20. Most people forget more easily the death of a parent than the loss of their property.

Source

author

References

Christie, R., & Geis, F. L., (1970). Studies in Machiavellianism. New York: Academic Press.

Hunter, J. E., Gerbing, D. W., and Boster, F. J. (1982). Machiavellian beliefs and personality: The construct invalidity of the Machiavellian dimension. Journal of Personality and Social Psychology, 43, 1293-1305.

`dataReading` *Data: Reading Ability*

Description

Reading ability test score and also verbal aptitude test score, number of absences from school and family income in USD $1000's. Data are simulated.

Usage

```
data(dataReading)
```

Format

A data table with 100 observations.

Source

author

`Density` *Density Curves from Data plus Histogram*

Description

Abbreviation: `dn`

Plots a normal density curve and/or a general density curve superimposed over a histogram, all estimated from the data. Also reports the Shapiro-Wilk normality test and summary statistics.

If the provided object to analyze is a set of multiple variables, including an entire data frame, then each non-numeric variable in the data frame is analyzed and the results written to a pdf file in the current working directory. The name of each output pdf file that contains a bar chart and its path are specified in the output.

Usage

```
Density(x, data=mydata, n.cat=getOption("n.cat"),

       bw="nrd0", type=c("both", "general", "normal"),
       bin.start=NULL, bin.width=NULL,

       col.fill=getOption("col.fill.pt"),
       col.bg=getOption("col.bg"),
       col.grid=getOption("col.grid"),

       col.nrm="black", col.gen="black",
       col.fill.nrm=NULL, col.fill.gen=NULL,

       cex.axis=.85, col.axis="gray30", col.ticks="gray30",
       x.pt=NULL, xlab=NULL, main=NULL, y.axis=FALSE,
       x.min=NULL, x.max=NULL, band=FALSE,

       quiet=getOption("quiet"),
       pdf.file=NULL, pdf.width=5, pdf.height=5, ...)

dn(...)
```

Arguments

`x`	Variable(s) to analyze. Can be a single numerical variable, either within a data frame or as a vector in the user's workspace, or multiple variables in a data frame such as designated with the `c` function, or an entire data frame. If not specified, then defaults to all numerical variables in the specified data frame, `mydata` by default.

data	Optional data frame that contains the variable(s) of interest, default is mydata.
n.cat	For the analysis of multiple variables, such as a data frame, specifies the largest number of unique values of variable of a numeric data type for which the variable will be analyzed as a categorical. Set to 0 to turn off.
bw	Bandwidth of kernel estimation.
type	Type of density curve plotted. By default, both the general density and the normal density are plotted.
bin.start	Optional specified starting value of the bins.
bin.width	Optional specified bin width, which can be specified with or without a bin.start value.
col.fill	Default (for default color theme of "blue") is to display the histogram in a light gray. To suppress, the histogram, specify a color of "transparent".
col.bg	Color of the plot background.
col.grid	Color of the grid lines.
col.nrm	Color of the normal curve.
col.gen	Color of the general density curve.
col.fill.nrm	Fill color for the estimated normal curve, with a partially transparent blue as the default for a blue color theme, and transparent for all other themes.
col.fill.gen	Fill color for the estimated general density curve, with a partially transparent light red as the default as the default for a blue color theme, and transparent for all other themes.
cex.axis	Scale magnification factor, which by default displays the axis values to be smaller than the axis labels.
col.axis	Color of the font used to label the axis values.
col.ticks	Color of the ticks used to label the axis values.
x.pt	Value of the point on the x-axis for which to draw a unit interval around illustrating the corresponding area under the general density curve. Only applies when requesting type=general.
xlab	Label for x-axis.
main	Title of graph.
y.axis	Specifies if the y-axis, the density axis, should be included.
x.min	Smallest value of the variable x plotted on the x-axis.
x.max	Largest value of the variable x plotted on the x-axis.
band	If TRUE, add a rug plot, a direct display of density in the form of a narrow band beneath the density curve
quiet	If set to TRUE, no text output. Can change system default with set function.
pdf.file	Name of the pdf file to which graphics are redirected. If there is no filetype of .pdf, the filetype is added to the name.
pdf.width	Width of the pdf file in inches.
pdf.height	Height of the pdf file in inches.
...	Other parameter values for graphics as defined processed by plot, including xlim, ylim, lwd and cex.lab, col.main, density, and, for the general density calculations, can set bandwidth with the standard bw, etc.

Details

OVERVIEW

Results are based on the standard dnorm function and density R functions for estimating densities from data, as well as the hist function for calculating a histogram. Colors are provided by default and can also be specified.

The default histogram can be modified with the bin.start and bin.width options. Use the Histogram function in this package for more control over the parameters of the histogram.

The limits for the axes are automatically calculated so as to provide sufficient space for the density curves and histogram, and should generally not require user intervention. Also, the curves are centered over the plot window so that the resulting density curves are symmetric even if the underlying histogram is not. The estimated normal curve is based on the corresponding sample mean and standard deviation.

If x.pt is specified, then type is set to general and y.axis set to TRUE.

DATA

The data may either be a vector from the global environment, the user's workspace, as illustrated in the examples below, or one or more variable's in a data frame, or a complete data frame. The default input data frame is mydata. Can specify the source data frame name with the data option. If multiple variables are specified, only the numerical variables in the list of variables are analyzed. The variables in the data frame are referenced directly by their names, that is, no need to invoke the standard R mechanisms of the mydata$name notation, the with function or the attach function. If the name of the vector in the global environment and of a variable in the input data frame are the same, the vector is analyzed.

COLOR THEME

Individual colors in the plot can be manipulated with options such as col.bars for the color of the histogram bars. A color theme for all the colors can be chosen for a specific plot with the colors option with the lessR function set. The default color theme is blue, but a gray scale is available with "gray", and other themes are available as explained in set, such as "red" and "green". Use the option ghost=TRUE for a black background, no grid lines and partial transparency of plotted colors.

VARIABLE LABELS

If variable labels exist, then the corresponding variable label is by default listed as the label for the horizontal axis and on the text output. For more information, see Read.

PDF OUTPUT

Because of the customized graphic windowing system that maintains a unique graphic window for the Help function, the standard graphic output functions such as pdf do not work with the lessR graphics functions. Instead, to obtain pdf output, use the pdf.file option, perhaps with the optional pdf.width and pdf.height options. These files are written to the default working directory, which can be explicitly specified with the R setwd function.

ONLY VARIABLES ARE REFERENCED

The referenced variable in a lessR function can only be a variable name (or list of variable names). This referenced variable must exist in either the referenced data frame, mydata by default, or in the user's workspace, more formally called the global environment. That is, expressions cannot be directly evaluated. For example:

```
> Density(rnorm(50))   # does NOT work
```

Instead, do the following:

```
> Y <- rnorm(50)   # create vector Y in user workspace
> Density(Y)     # directly reference Y
```

Value

Based on the standard R function density, invisibly returns an object of class "density". For details see density.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

See Also

dnorm, density, hist, plot, rgb, shapiro.test.

Examples

```
# create data frame, mydata, to mimic reading data with Read function
# mydata contains both numeric and non-numeric data
mydata <- data.frame(rnorm(50), rnorm(50), rnorm(50), rep(c("A","B"),25))
names(mydata) <- c("X","Y","Z","C")

# normal curve and general density curves superimposed over histogram
# all defaults
Density(Y)

# short name
dn(Y)

# save the density plot to a pdf file
Density(Y, pdf.file="MyDensityPlot.pdf")

# suppress the histogram, leaving only the density curves
# specify x-axis label per the xlab option for the plot function
Density(Y, col.bars="transparent", xlab="My Var")

# specify (non-transparent) colors for the curves,
# to make transparent, need alpha option for the rgb function
Density(Y, col.nrm="darkgreen", col.gen="plum")

# display only the general estimated density
#  so do not display the estimated normal curve
# specify the bandwidth for the general density curve,
#  use the standard bw option for the density function
Density(Y, type="general", bw=.6)

# display only the general estimated density and a corresponding
#  interval of unit width around x.pt
Density(Y, type="general", x.pt=2)
```

```
# variable of interest is in a data frame which is not the default mydata
# access the breaks variable in the R provided warpbreaks data set
# although data not attached, access the variable directly by its name
Density(breaks, data=warpbreaks)

# densities for all numeric variables in a data frame
Density()

# densities for all specified numeric variables in a list of variables
# e.g., use the combine or c function to specify a list of variables
Density(c(X,Y))
```

`details` *Display Contents of a Data File and Optional Variable Labels*

Description

Abbreviation: `details.brief`

Provides feedback regarding a data frame which includes the variable names, the dimensions of the resulting data frame, the data type for each variable, and the values of the variables in the data file for the first and last rows of the data. In addition, an analysis of missing data is provided, listing the number of missing values for each variable and for each observation.

Usage

```
details(data=mydata, n.mcut=1, miss.zero=FALSE, max.lines=30,
         miss.show=30, miss.matrix=FALSE, brief=getOption("brief"))

details.brief(..., brief=TRUE)
```

Arguments

`data`	Data frame for which to provide the details.
`n.mcut`	For the missing value analysis, list the row name and number of missing values if the number of missing exceeds or equals this cutoff.
`miss.zero`	For the missing value analysis, list the variable name or the row name even for values of 0. By default only variables and rows with missing data are listed.
`max.lines`	Maximum number of lines to list of the data and labels.
`miss.show`	For the missing value analysis, the number of rows, one row per observation, that has as many or missing values as `n.mcut`.
`miss.matrix`	For the missing value analysis, if there is any missing data, list a version of the complete data table with a 0 for a non-missing value and a 1 for a missing value.
`brief`	If `TRUE`, display only variable names table plus any variable labels.
`...`	Further arguments to be passed to or from methods.

Details

MISSING DATA

By default, `details` provides a list of each variable and each row with the display of the number of associated missing values, indicated by the standard R missing value code NA. To not list the variable name or row name of variables or rows without missing data, invoke the `miss.zero=FALSE` option, which can appreciably reduce the amount of output for large data sets. To view the entire data table in terms of 0's and 1's for non-missing and missing data, respectively, invoke the `miss.matrix=TRUE` option.

VARIABLE LABELS

Standard R does not provide for variable labels, but `lessR` does. Variable labels can be provided for some or all of the variables in the data frames. One way to enter the variable labels is to read them from their own file with `details` with `labels` set to the full path name or `URL` of the labels file, or just the file name if the labels file is in the same directory as the data file. Another method is to include the labels directly in the data file. To to this, specify the file of variable labels with the `label="row2"` option. The web survey application Qualtrics downloads `csv` files in this format.

For a file that contains only labels, each row of the file, including the first row, consists of the variable name, a comma, and then the label, that is, standard `csv` format such as obtained with the `csv` option from a standard worksheet application such as Microsoft Excel or LibreOffice Calc. Not all variables in the data frame that contains the data, usually `mydata`, need have a label, and the variables with their corresponding labels can be listed in any order. An example follows.

I2,This instructor presents material in a clear and organized manner.
I4,Overall, this instructor was highly effective in this class.
I1,This instructor has command of the subject.
I3,This instructor relates course materials to real world situations.

If there is a comma in the variable label, then the label needs to be enclosed in quotes.

The `lessR` functions that provide analysis, such as `Histogram` for a histogram, automatically include the variable labels in their output, such as the title of a graph. Standard R functions can also use these variable labels by invoking the `label` function, such as setting `main=label(I4)` to put the variable label for a variable named I4 in the title of a graph.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

See Also

`Read`.

Examples

```
# read the built-in data set datEmployee
# this provides an automatic call to details
mydata <- Read("Employee", format="lessR")

# manually request the details for mydata
details()
```

```
# manually request just variable names, labels for mydata
details.brief()
```

`Help` *Help System for Statistics by Topic that Suggests Related Functions*

Description

Abbreviation: `hl`

R works by entering function names and arguments. R provides extensive help for each available function based on a function's name, but these names are not apparent to someone who has not memorized the names. To alleviate this problem, this help system suggests and briefly explains various function calls regarding a requested topic for statistical analysis. The primary call is `Help()`, which displays the main help menu in a persistent graphics window, that is, which remains until explicitly closed by the user regardless of additional graphics analyses. Each specific Help window can be called with any of the function names, or their abbreviations or related expressions. The argument can be expressed in any combination of uppercase or lowercae lettes.

Usage

```
Help(topic=NULL)

hl(...)
```

Arguments

`topic`	Message reference, either null to specify a list of available topics or a specific argument to reference a specific help message.
`...`	Other parameter values to be passed to Help.

Details

`Help()` displays a list of help topics, listed below.

`Help("topic")` generally displays the available functions relevant for the specified topic. If the name of the topic is a `lessR` statistical function, the abbreviated form of the name can also be invoked.

`Help("lessR")` calls up the link to the lessR manual and news, which includes current updates.

—

`data`: Create a csv data file from a worksheet application.

`Read`: Read an external data file in csv format.

`Write`: Write the contents of mydata to a data file in csv format.

`library`: Many libraries of functions developed by others can be added to R.

`transform`: Edit and create new variables from existing variables.

`system`: System level settings, such as a color theme for graphics.

—

Histogram: Histogram, box plot, dot plot and density curve..

BarChart: Bar chart or pie chart of a categorical variable.

Plot: Scatterplot for one or two variables, line chart.

RunChart: Run chart or time series chart.

—

SummaryStats: Summary statistics for one and two variables.

one.sample: Analysis of a single sample of data.

ttest: The mean difference and related statistics.

ANOVA: Compare means across two or more groups.

power: Power analysis for the t-test.

Correlation: Correlation analysis.

Regression: Regression analysis.

Logistic: Logistic regression analysis.

factor.analysis: Confirmatory or exploratory factor analysis.

—

prob: Probabilities for Normal and t-distributions.

random: Random number generation.

sample: Generate random samples.

—

lessR: lessR manual and list of updates to current version

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

See Also

help.

Examples

```
# list the information needed to access a specific topic
Help()
# short name
hl()

# specific help message regarding a histogram and related functions
Help(Histogram)

# other ways to call the same help message
Help(histogram)  # case insensitive
Help(hs)
Help(hist)
```

```
Help(hst)
Help(boxplot)
```

`Histogram` *Histogram with Color*

Description

Abbreviation: `hs`

From the standard R function `hist` plots a frequency histogram with default colors, including background color and grid lines plus an option for a relative frequency and/or cumulative histogram, as well as summary statistics and a table that provides the bins, midpoints, counts, proportions, cumulative counts and cumulative proportions. Bins can be selected several different ways besides the default, including specifying just the bin width and/or the bin start. Also provides improved error diagnostics and feedback for the user on how to correct the problem when the bins do not contain all of the specified data.

If the provided object to analyze is a set of multiple variables, including an entire data frame, then each non-numeric variable in the data frame is analyzed and the results written to a pdf file in the current working directory. The name of each output pdf file that contains a bar chart and its path are specified in the output.

Usage

```
Histogram(x=NULL, data=mydata, n.cat=getOption("n.cat"),

          col.fill=getOption("col.fill.bar"),
          col.stroke=getOption("col.stroke.bar"),
          col.bg=getOption("col.bg"),
          col.grid=getOption("col.grid"),

          col.reg="snow2", over.grid=FALSE,
          cex.axis=.85, col.axis="gray30", col.ticks="gray30",

          breaks="Sturges", bin.start=NULL, bin.width=NULL, bin.end=NULL,

          prop=FALSE, cumul=c("off", "on", "both"),
          digits.d=NULL, xlab=NULL, ylab=NULL, main=NULL,

          quiet=getOption("quiet"),
          pdf.file=NULL, pdf.width=5, pdf.height=5, ...)

hs(...)
```

Arguments

x	Variable(s) to analyze. Can be a single numerical variable, either within a data frame or as a vector in the user's workspace, or multiple variables in a data frame such as designated with the c function, or an entire data frame. If not specified, then defaults to all numerical variables in the specified data frame, mydata by default.
data	Optional data frame that contains the variable(s) of interest, default is mydata.
n.cat	For the analysis of multiple variables, such as a data frame, specifies the largest number of unique values of variable of a numeric data type for which the variable will be analyzed as a categorical. Set to 0 to turn off.
col.fill	Color of the bars. To set transparency level, use lessR function set or use rgb function directly.
col.stroke	Color of the border of the bars. To set transparency level, use function set or use rgb function directly.
col.bg	Color of the plot background.
col.grid	Color of the grid lines.
col.reg	The color of the superimposed, regular histogram when cumul="both".
over.grid	If TRUE, plot the grid lines over the histogram.
cex.axis	Scale magnification factor, which by defaults displays the axis values to be smaller than the axis labels. Provides the functionality of, and can be replaced by, the standard R cex.axis.
col.axis	Color of the font used to label the axis values.
col.ticks	Color of the ticks used to label the axis values.
breaks	The method for calculating the bins, or an explicit specification of the bins, such as with the standard R seq function or other options provided by the hist function.
bin.start	Optional specified starting value of the bins.
bin.width	Optional specified bin width, which can be specified with or without a bin.start value.
bin.end	Optional specified value that is within the last bin, so the actual endpoint of the last bin may be larger than the specified value.
prop	Specify proportions or relative frequencies on the vertical axis. Default is FALSE.
cumul	Specify a cumulative histogram. The value of "on" displays the cumulative histogram, with default of "off". The value of "both" superimposes the regular histogram.
digits.d	Number of significant digits for each of the displayed summary statistics.
xlab	Label for x-axis. Defaults to variable name.
ylab	Label for y-axis. Defaults to Frequency or Proportion.
main	Title of graph.
quiet	If set to TRUE, no text output. Can change system default with set function.

`pdf.file`	Name of the pdf file to which graphics are redirected. If there is no filetype of `.pdf`, the filetype is added to the name.
`pdf.width`	Width of the pdf file in inches.
`pdf.height`	Height of the pdf file in inches.
`...`	Other parameter values for graphics as defined processed by `hist` and `plot`, including `xlim`, `ylim`, `lwd` and `cex.lab`, `col.main`, `density`, etc. Also includes `labels=TRUE` to display the frequency of the bin at the top of the corresponding bar.

Details

OVERVIEW

Results are based on the standard R `hist` function to calculate and plot a histogram, plus the additional provided color capabilities, a relative frequency histogram and summary statistics. However, a histogram with densities is not supported. The `freq` option from the standard R `hist` function has no effect as it is always set to `FALSE` in each internal call to `hist`. To plot densities, which correspond to setting `freq` to `FALSE`, use the `lessR` function `Density`.

DATA

The data may either be a vector from the global environment, the user's workspace, as illustrated in the examples below, or one or more variable's in a data frame, or a complete data frame. The default input data frame is `mydata`. Can specify the source data frame name with the `data` option. If multiple variables are specified, only the numerical variables in the list of variables are analyzed. The variables in the data frame are referenced directly by their names, that is, no need to invoke the standard `R` mechanisms of the `mydata$name` notation, the `with` function or the `attach` function. If the name of the vector in the global environment and of a variable in the input data frame are the same, the vector is analyzed.

To obtain a histogram of each numerical variable in the `mydata` data frame, use `Histogram()`. Or, for a data frame with a different name, insert the name between the parentheses. To analyze a subset of the variables in a data frame, specify the list with either a : or the `c` function, such as m01:m03 or c(m01,m02,m03).

COLORS

Individual colors in the plot can be manipulated with options such as `col.bars` for the color of the histogram bars. A color theme for all the colors can be chosen for a specific plot with the `colors` option with the `lessR` function `set`. The default color theme is `blue`, but a gray scale is available with `"gray"`, and other themes are available as explained in `set`, such as `"red"` and `"green"`. Use the option `ghost=TRUE` for a black background, no grid lines and partial transparency of plotted colors.

VARIABLE LABELS

If variable labels exist, then the corresponding variable label is by default listed as the label for the horizontal axis and on the text output. For more information, see `Read`.

ONLY VARIABLES ARE REFERENCED

The referenced variable in a `lessR` function can only be a variable name (or list of variable names). This referenced variable must exist in either the referenced data frame, `mydata` by default, or in the user's workspace, more formally called the global environment. That is, expressions cannot be directly evaluated. For example:

```
> Histogram(rnorm(50))   # does NOT work
```

Instead, do the following:

```
> Y <- rnorm(50)   # create vector Y in user workspace
> Histogram(Y)     # directly reference Y
```

ERROR DETECTION
A somewhat relatively common error by beginning users of the base R hist function may encounter is to manually specify a sequence of bins with the seq function that does not fully span the range of specified data values. The result is a rather cryptic error message and program termination. Here, Histogram detects this problem before attempting to generate the histogram with hist, and then informs the user of the problem with a more detailed and explanatory error message. Moreover, the entire range of bins need not be specified to customize the bins. Instead, just a bin width need be specified, bin.width, and/or a value that begins the first bin, bin.start. If a starting value is specified without a bin width, the default Sturges method provides the bin width.

PDF OUTPUT
Because of the customized graphic windowing system that maintains a unique graphic window for the Help function, the standard graphic output functions such as pdf do not work with the lessR graphics functions. Instead, to obtain pdf output, use the pdf.file option, perhaps with the optional pdf.width and pdf.height options. These files are written to the default working directory, which can be explicitly specified with the R setwd function.

Value

Based on the standard R function hist, invisibly returns an object of class "histogram". For details see hist.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

See Also

hist, plot, par, set.

Examples

```
# generate 50 random normal data values with three decimal digits
y <- round(rnorm(50),3)


# --------------------
# different histograms
# --------------------

# histogram with all defaults
Histogram(y)
# short form
hs(y)
# compare to standard R function hist
hist(y)
```

```
# save the histogram to a pdf file
Histogram(y, pdf.file="MyHistogram.pdf")

# histogram with no grid, red bars, black background, and black border
Histogram(y, col.grid="transparent", col.bg="black",
          col.fill="red", col.stroke="black")
# or set this color scheme for all subsequent analyses
set("red", col.grid="transparent", col.bg="black", col.stroke.bar="black")
Histogram(y)

# histogram with orange color theme, transparent orange bars, no grid lines
set(colors="orange", ghost=TRUE)
Histogram(y)
# back to default of "blue" color theme
set(colors="blue")

# histogram with specified bin width
# can also use bin.start
Histogram(y, bin.width=.25)

# histogram with specified bins and grid lines displayed over the histogram
Histogram(y, breaks=seq(-5,5,.25), xlab="My Variable", over.grid=TRUE)

# histogram with bins calculated with the Scott method and values displayed
Histogram(y, breaks="Scott", labels=TRUE)

# histogram with the number of suggested bins, with proportions
Histogram(y, breaks=15, prop=TRUE)

# histogram with specified colors, overriding defaults
# col.bg and col.grid are defined in histogram
# all other parameters are defined in hist, par and plot functions
Histogram(y, col.fill="darkblue", col.stroke="lightsteelblue4", col.bg="ivory",
  col.grid="darkgray", density=25, angle=-45, cex.lab=.8, cex.axis=.8,
  col.lab="sienna3", main="My Title", col.main="gray40", xlim=c(-5,5), lwd=2,
  xlab="My Favorite Variable")

# ---------------------
# cumulative histograms
# ---------------------

# cumulative histogram with superimposed regular histogram, all defaults
Histogram(y, cumul="both")

# cumulative histogram plus regular histogram
# present with proportions on vertical axis, override other defaults
Histogram(y, cumul="both", breaks=seq(-4,4,.25), prop=TRUE,
  col.reg="mistyrose")


# ------------------------------------------------
# histograms for data frames and multiple variables
# ------------------------------------------------
```

```
# create data frame, mydata, to mimic reading data with Read function
# mydata contains both numeric and non-numeric data
mydata <- data.frame(rnorm(50), rnorm(50), rnorm(50), rep(c("A","B"),25))
names(mydata) <- c("X","Y","Z","C")

# although data not attached, access the variable directly by its name
Histogram(X)

# histograms for all numeric variables in data frame called mydata
#  except for numeric variables with unique values < n.cat
# mydata is the default name, so does not need to be specified with data
Histogram()

# variable of interest is in a data frame which is not the default mydata
# access the breaks variable in the R provided warpbreaks data set
# although data not attached, access the variable directly by its name
Histogram(breaks, data=warpbreaks)

# histograms for all numeric variables in data frame called mydata
Histogram()
#  with specified options
Histogram(col.fill="palegreen1", col.bg="ivory", labels=TRUE)

# histograms for all specified numeric variables
# use the combine or c function to specify a list of variables
Histogram(c(X,Y))
```

`label` *Return or Assign a Variable Label*

Description

Display a variable label for output, either text output at the console or graphics, such as a title on a graph. To return a variable label generally applies to standard R functions such that the functions can access `lessR` variable labels. Or, the variable name and label can be listed on the standard output. To assign a variable label, invoke the `value` option and assign the output to a specified data frame.

Usage

```
label(x, value=NULL, data=mydata)
```

Arguments

`x`	The variable for which to obtain the corresponding variable label.
`value`	If assigned, then the specified data frame is updated with this asssigned label.
`data`	Data frame that contains the variable of interest. The output of the function is assigned to this data frame.

Details

Standard R does not provide for variable labels, but lessR does. Read the labels with the lessR Read function, as explained in the corresponding documentation. Individual variable labels can also be assigned with this function. Not all variables need have a label, and the variables with their corresponding labels can be listed or assigned in any order.

The function provides two different modes. The first mode is to return the variable name and label for an existing variable label. One such use is to provide the function as an argument to an existing R function call to access a lessR variable label. For example, use the function as the argument for main in graphics output, where main is the title of the graph. This mode is triggered by not invoking the value option.

The second mode is to assign a variable label to an existing variable. Invoke this mode by specifying a variable label with the value option. The function accesses the entire specified data frame, and then modifies the specified variable label. As such, assign the output of the function to the data frame of interest, such as the default mydata. One use of this function is to add a variable label to a data frame that contains a new variable created by a transformation of the existing variables.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

See Also

Read.

Examples

```
# read the internal lessR data frame that contains variable labels
mydata <- Read("Employee", format="lessR")

# variable label as the title of a graph from a standard R function
# the data are not attached, so for standard R functions, must
#   identify the relevant data frame, such as with function: with
with(mydata, barplot(table(Dept), main=label(Dept)))
with(mydata, hist(Salary, main=label(Salary)))

# assign a new label for the variable Years in mydata
mydata <- label(Years, "Years Worked")
# verify
label(Years)
# or view all variable labels in mydata
details.brief()

mydata <- Read("Employee", format="lessR")
# specify a label of variable in a data frame other than mydata
myd <- Subset(Gender=="M")
myd <- label(Gender, "Only is Male", data=myd)
details.brief(myd)
```

LineChart *Line Chart such as a Run Chart or Time-Series Chart*

Description

Abbreviation: lc

Plots a line chart, the values of the variable ordered according to their order in the data frame. Usually this ordering would be an ordering according to time, which yields a run chart. The default run chart provides the index, that is, sequential position, of each value of the variable from 1 to the last value. Optionally dates can be provided so that a time-series plot is produced.

For data of one variable exhibiting little trend, the center line is provided for the generation of a run chart, plotting the values of a variable in order of occurrence over time. When the center line, the median by default, is plotted, the analyses of the number and composition of the individual runs, number of consecutive values above or below the center line, is also displayed. Also, the defaults change for each of the types of plots. The intent is to rely on the default values for a relatively sophisticated plot, particularly when compared to the default values of the standard R plot function called with a single variable.

If the provided object to analyze is a set of multiple variables, including an entire data frame, then each non-numeric variable in the data frame is analyzed and the results written to a pdf file in the current working directory. The name of each output pdf file that contains a bar chart and its path are specified in the output.

Usage

```
LineChart(x, data=mydata, n.cat=getOption("n.cat"), type=NULL,

          col.fill=getOption("col.fill.bar"),
          col.stroke=getOption("col.stroke.pt"),
          col.bg=getOption("col.bg"),
          col.grid=getOption("col.grid"),
          col.line=getOption("col.stroke.pt"),

          col.area=NULL, col.box="black",

          shape.pts=21, cex.axis=.85, col.axis="gray30",
          col.ticks="gray30", xy.ticks=TRUE, line.width=1.1,
          xlab=NULL, ylab=NULL, main=NULL, cex=NULL,

          time.start=NULL, time.by=NULL, time.reverse=FALSE,

          center.line=c("default", "mean", "median", "zero", "off"),

          quiet=getOption("quiet"),
          pdf.file=NULL, pdf.width=5, pdf.height=5, ...)

lc(...)
```

Arguments

x	Variable(s) to analyze. Can be a single numerical variable, either within a data frame or as a vector in the user's workspace, or multiple variables in a data frame such as designated with the c function, or an entire data frame. If not specified, then defaults to all numerical variables in the specified data frame, mydata by default.
data	Optional data frame that contains the variable(s) of interest, default is mydata.
n.cat	For the analysis of multiple variables, such as a data frame, specifies the largest number of unique values of variable of a numeric data type for which the variable will be analyzed as a categorical. Set to 0 to turn off.
type	Character string that indicates the type of plot, either "p" for points, "l" for line, or "b" for both. The default is "b" for both points and lines.
col.fill	For plotted points, the interior color of the point. For a scatterplot the default value is transparent. For a run chart the default value is the color of the point's border, col.stroke.
col.stroke	Color of the border of the plotted points.
col.bg	Color of the plot background.
col.grid	Color of the grid lines, with a default of "grey90".
col.line	Color of any plotted line segments, with a default of "darkblue".
col.area	Color of area under the plotted line segments. To have a border at the bottom and right of a run chart but retain the property of no area color, specify a color of "transparent". If the values exhibit a trend and dates are specified with x.start, the default area color is slategray3, otherwise there is no default color.
col.box	Color of border around the plot background, the box, that encloses the plot, with a default of "black".
shape.pts	The standard plot character, with values defined in help(points). The default value is 21, a circle with both a border and filled area, specified here as col.stroke and col.fill. col.fill defaults to col.stroke.
cex.axis	Scale magnification factor, which by defaults displays the axis values to be smaller than the axis labels.
col.axis	Color of the font used to label the axis values.
col.ticks	Color of the ticks used to label the axis values.
xy.ticks	Flag that indicates if tick marks and associated values on the axes are to be displayed.
line.width	Width of the density lines.
xlab	Label for x-axis. For two variables specified, x and y, if xlab not specified, then the label becomes the name of the corresponding variable. If xy.ticks is FALSE, then no label is displayed. If no y variable is specified, then xlab is set to Index unless xlab has been specified.
ylab	Label for y-axis. If not specified, then the label becomes the name of the corresponding variable. If xy.ticks is FALSE, then no label displayed.

main	Label for the title of the graph. If the corresponding variable labels exist, then the title is set by default from the corresponding variable labels.
cex	Magnification factor for any displayed points, with default of cex=1.0.
time.start	Optional starting date for first data value. Format must be "%Y-%m-%d" or "%Y/%m/%d". If using with x.reverse, the first date is after the data are reverse sorted. Not needed if data are a time series with ts function.
time.by	Accompanies the time.start specification, the interval to increment the date for each sequential data value. A character string, containing one of "day", "week", "month" or "year". This string can optionally be preceded by a positive or negative integer and a space, or followed by "s", as specified in seq.Date. Not needed if data are a time series.
time.reverse	When TRUE, reverse the ordering of the dates, particularly when the data are listed such that first row of data is the newest. Accompanies the time.start specification.
center.line	Plots a dashed line through the middle of a run chart. The two possible values for the line are "mean" and "median". Provides a centerline for the "median" by default when the values randomly vary about the mean. A value of "zero" specifies the center line should go through zero.
quiet	If set to TRUE, no text output. Can change system default with set function.
pdf.file	Name of the pdf file to which graphics are redirected. If there is no filetype of .pdf, the filetype is added to the name.
pdf.width	Width of the pdf file in inches.
pdf.height	Height of the pdf file in inches.
...	Other parameters such as from par.

Details

OVERVIEW

The line chart is based on the standard R function plot when called with only a single variable.

The values on the horizontal axis of the line chart are automatically generated. The default is the index variable, the ordinal position of each data value, in which case this version of the line chart is a run chart. Or, dates on the horizontal axis can be specified from the specified starting date given by x.start and the accompanying increment as given by x.by, in which case the line chart is typically referred to as a time series chart.

If the data values randomly vary about the mean, the default is to plot the mean as the center line of the graph, otherwise the default is to ignore the center line. The default plot type for the line chart is type="b", for both points and the corresponding connected line segments. The size of the points is automatically reduced according to the number of points of points plotted, and the cex option can override the computed default. If the area below the plotted values is specified to be filled in with color, then the default line type changes to type="l".

DATA

The data may either be a vector from the global environment, the user's workspace, as illustrated in the examples below, or one or more variable's in a data frame, or a complete data frame. The default input data frame is mydata. Can specify the source data frame name with the data option. If multiple variables are specified, only the numerical variables in the list of variables are analyzed.

The variables in the data frame are referenced directly by their names, that is, no need to invoke the standard R mechanisms of the mydata$name notation, the with function or the attach function. If the name of the vector in the global environment and of a variable in the input data frame are the same, the vector is analyzed.

COLORS

Individual colors in the plot can be manipulated with options such as col.stroke for the color of the border of the plotted points. A color theme for all the colors can be chosen for a specific plot with the colors option with the lessR function set. The default color theme is blue, but a gray scale is available with "gray", and other themes are available as explained in set, such as "red" and "green". Use the option ghost=TRUE for a black background, no grid lines and partial transparency of plotted colors.

VARIABLE LABELS

Although standard R does not provide for variable labels, lessR does, obtained from the Read function. If the variable labels exist, then the corresponding variable label is by default listed as the label for the horizontal axis and on the text output. For more information, see Read.

PDF OUTPUT

Because of the customized graphic windowing system that maintains a unique graphic window for the Help function, the standard graphic output functions such as pdf do not work with the lessR graphics functions. Instead, to obtain pdf output, use the pdf.file option, perhaps with the optional pdf.width and pdf.height options. These files are written to the default working directory, which can be explicitly specified with the R setwd function.

ONLY VARIABLES ARE REFERENCED

The referenced variable in a lessR function can only be a variable name (or list of variable names). This referenced variable must exist in either the referenced data frame, mydata by default, or in the user's workspace, more formally called the global environment. That is, expressions cannot be directly evaluated. For example:

```
> LineChart(rnorm(50))   # does NOT work
```

Instead, do the following:

```
> Y <- rnorm(50)   # create vector Y in user workspace
> LineChart(Y)     # directly reference Y
```

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

See Also

plot, set.

Examples

```
# create data frame, mydata, to mimic reading data with Read function
# mydata contains both numeric and non-numeric data
mydata <- data.frame(rnorm(50), rnorm(50), rnorm(50), rep(c("A","B"),25))
names(mydata) <- c("X","Y","Z","C")

# default run chart
```

```
LineChart(Y)
# short name
lc(Y)
# compare to standard R plot
plot(mydata$Y, type="l")

# save run chart to a pdf file
LineChart(Y, pdf.file="MyLineChart.pdf")

# LineChart in gray scale, then back to default "blue"
set(colors="gray")
LineChart(Y)
set(colors="blue")

# customize run chart, pch=24: filled triangle point-up,
LineChart(Y, line.width=2, col.stroke="sienna3", shape.pts=24,
  col.area="slategray3", col.bg="mintcream", ylim=c(-3.5,3.5),
  center.line="median")

# generate steadily increasing values
Y <- sort(rexp(50))
# default line chart
LineChart(Y)
# line chart with border around plotted values
LineChart(Y, col.area="transparent")
# time series chart, i.e., with dates, and filled area
# with option label for the x-axis
LineChart(Y, time.start="2000/09/01", time.by="3 months")
# time series chart from a time series object
y.ts <- ts(Y, start=c(2000, 9), frequency=4)
LineChart(y.ts)

# LineChart with built-in data set
LineChart(breaks, data=warpbreaks)

# Line charts for all numeric variables in a data frame
LineChart()

# Line charts for all specified numeric variables in a list of variables
# e.g., use the combine or c function to specify a list of variables
LineChart(c(X,Y))
```

`Logit` *Logit Regression Analysis*

Description

Abbreviation: `lr`

Based directly on the standard R `glm` function with `family="binomial"`, automatically provides a logit regression analysis with graphics from a single, simple function call with many default settings,

each of which can be re-specified. By default the data exists as a data frame with the default name of `mydata`, such as data read by the `lessR Read` function. Specify the model in the function call according to an R `formula`, that is, the response variable followed by a tilde, followed by the list of predictor variables, each pair separated by a plus sign. The response variable is either a factor with two levels or a numeric variable with values only of 0 and 1.

Default output includes the inferential analysis of the estimated coefficients and model, sorted residuals and Cook's Distance, and sorted fitted values for existing data or new data. For a single predictor variable model, the scatterplot of the data with plotted logit function is provided.

Can also be called from the more general `model` function.

Usage

```
Logit(my.formula, data=mydata, digits.d=4, text.width=120,

     brief=getOption("brief"),

     res.rows=NULL, res.sort=c("cooks","rstudent","dffits","off"),
     pred=TRUE, pred.all=FALSE, cooks.cut=1,

     X1.new=NULL, X2.new=NULL, X3.new=NULL, X4.new=NULL,
     X5.new=NULL,

     pdf.file=NULL, pdf.width=5, pdf.height=5, ...)

lr(...)
```

Arguments

`my.formula`	Standard R `formula` for specifying a model. For example, for a response variable named Y and two predictor variables, X1 and X2, specify the corresponding linear model as Y ~ X1 + X2.
`data`	The default name of the data frame that contains the data for analysis is `mydata`, otherwise explicitly specify.
`digits.d`	For the Basic Analysis, it provides the number of decimal digits. For the rest of the output, it is a suggestion only.
`text.width`	Width of the text output at the console.
`brief`	If set to `TRUE`, reduced text output. Can change system default with `set` function.
`res.rows`	Default is 25, which lists the first 25 rows of data sorted by the specified sort criterion. To turn this option off, specify a value of 0. To see the output for all observations, specify a value of `"all"`.
`res.sort`	Default is `"cooks"`, for specifying Cook's distance as the sort criterion for the display of the rows of data and associated residuals. Other values are `"rstudent"` for Studentized residuals, and `"off"` to not provide the analysis.
`pred`	Default is `TRUE`, which, produces confidence and prediction intervals for each row, or selected rows, of data.

pred.all	Default is FALSE, which produces prediction intervals only for the first, middle and last five rows of data.
cooks.cut	Cutoff value of Cook's Distance at which observations with a larger value are flagged in red and labeled in the resulting scatterplot of Residuals and Fitted Values. Default value is 1.0.
X1.new	Values of the first listed predictor variable for which forecasted values and corresponding prediction intervals are calculated.
X2.new	Values of the second listed predictor variable for which forecasted values and corresponding prediction intervals are calculated.
X3.new	Values of the third listed predictor variable for which forecasted values and corresponding prediction intervals are calculated.
X4.new	Values of the fourth listed predictor variable for which forecasted values and corresponding prediction intervals are calculated.
X5.new	Values of the fifth listed predictor variable for which forecasted values and corresponding prediction intervals are calculated.
pdf.file	Name of the pdf file to which graphics are redirected.
pdf.width	Width of the pdf file in inches.
pdf.height	Height of the pdf file in inches.
...	Other parameter values for R function glm which provides the core computations.

Details

OVERVIEW

The purpose of Logit is to combine the following function calls into one, as well as provide ancillary analyses such as as graphics, organizing output into tables and sorting to assist interpretation of the output. The basic analysis successively invokes several standard R functions beginning with the standard R function for estimation of the logit model, glm with family="binomial". The output of the analysis is stored in the object lm.out, available for further analysis in the R environment upon completion of the Logit function. By default automatically provides the analyses from the standard R functions, summary, confint and anova, with some of the standard output modified and enhanced. The residual analysis invokes fitted, resid, rstudent, and cooks.distance functions. The option for prediction intervals calls the standard generic R function predict.

The default analysis provides the model's parameter estimates and corresponding hypothesis tests and confidence intervals, goodness of fit indices, the ANOVA table, analysis of residuals and influence as well as the fitted value and standard error for each observation in the model.

DATA FRAME

The name mydata is by default provided by the Read function included in this package for reading and displaying information about the data in preparation for analysis. If all the variables in the model are not in the same data frame, the analysis will not be complete. The data frame does not need to be attached, just specified by name with the data option if the name is not the default mydata.

GRAPHICS

For models with a single predictor variable, a scatter plot of the data is produced, which also includes the fitted values. As with the density histogram plot of the residuals and the scatterplot of the

fitted values and residuals, the scatterplot includes a colored background with grid lines. If more than a single predictor variable, then a scatter plot matrix is produced.

FORECASTS
Fitted and forecasted values are listed for all rows of data if the number of rows is less than 25 or if `pred.all=TRUE`. If only some of the rows are listed, sorted by the fitted value, the first and last four rows of data are listed. Also the 4 rows immeadiately around the fitted value of 0.5 are listed.

RESIDUAL ANALYSIS
By default the residual analysis lists the data and fitted value for each observation as well as the residual, Studentized residual, Cook's distance and dffits, with the first 20 observations listed and sorted by Cook's distance. The residual displayed is the actual difference between fitted and observed, that is, with the setting in the `residuals` of `type="response"`. The `res.sort` option provides for sorting by the Studentized residuals or not sorting at all. The `res.rows` option provides for listing these rows of data and computed statistics statistics for any specified number of observations (rows). To turn off the analysis of residuals, specify `res.rows=0`.

INVOKED R OPTIONS
The `options` function is called to turn off the stars for different significance levels (show.signif.stars=FALSE), to turn off scientific notation for the output (scipen=30), and to set the width of the text output at the console to 120 characters. The later option can be re-specified with the `text.width` option. After `reg` is finished with a normal termination, the options are re-set to their values before the `reg` function began executing.

COLORS
The default color theme is `blue`, but a gray scale is available with `"gray"`, and other themes are available as explained in `set`, such as `"red"` and `"green"`. Use the option `ghost=TRUE` for a black background, no grid lines and partial transparency of plotted colors.

Value

Following the standard `R` function `glm`, invisibly returns an object of `class` inheriting from "glm" which inherits from the `class` "lm". Particularly useful for comparing nested models. Assign the output of `Logit` for a model to an object. Then for a nested model. Then use the `anova` function to compare the models as shown in the examples below.

Author(s)

David W. Gerbing (Portland State University; `<gerbing@pdx.edu>`)

See Also

`formula`, `glm`, `summary.glm`, `anova`, `confint`, `fitted`, `resid`, `rstudent`, `cooks.distance`

Examples

```
# Gender has values of "M" and "F"
mydata <- Read("Employee", format="lessR")
# logit regression
Logit(Gender ~ Years)

# short name
lr(Gender ~ Years)
```

```
# Modify the default settings as specified
Logit(Gender ~ Years, res.row=8, res.sort="rstudent", digits.d=8, pred=FALSE)

# Multiple logistic regression model
# Provide all default analyses
Logit(Gender ~ Years + Salary)

# compare nested models
# easier and better treatment of missing data to use lessR function:  Nest
full.model <- Logit(Gender ~ Years + Salary)
reduced.model <- Logit(Gender ~ Years)
anova(reduced.model, full.model)

# Save the three plots as pdf files 4 inches square, gray scale
Logit(Gender ~ Years, pdf.file="MyModel.pdf",
      pdf.width=4, pdf.height=4, colors="gray")

# Specify new values of the predictor variables to calculate
#  forecasted values
# Specify an input data frame other than mydata
mydata <- Read("Cars93", format="lessR")
Logit(Source ~ HP + MidPrice, X1.new=seq(100,250,50), X2.new=c(10,60,10))
```

`Merge` *Merge Two Data Frames Horizontally or Vertically*

Description

Abbreviation: `mrg`

Based directly on the standard R `merge` function for a horizontal merge that combines data frames horizontally, that is, adds variables (columns) to an existing data frame according to a common shared ID field. The vertical merge is based on the `rbind` function in which the two data frames have the same variables but different cases (observations), so the rows build vertically, stacked on top of each other. Any existing variable labels in the input data frames are retained in the merged data frame.

The advantages of this `lessR` function is that it provides a single function for merging data frames, adds text output to the standard R functions that provide feedback regarding properties of the merge, provides more detailed and presumably more useful error messages, and preserves any variable labels that might exist in one or both of the merged data frames.

Usage

```
Merge(data1, data2, by=NULL, quiet=getOption("quiet"), ...)

mrg(...)
```

Arguments

data1	The name of the first data frame from which to create the merged data frame.
data2	The name of the second data frame from which to create the merged data frame.
by	If specified, then signals a horizontal merge and the ID field by which the data frames are merged. Specify "row.names" for merging according to the row names.
quiet	If set to TRUE, no text output. Can change system default with set function.
...	The specified arguments.

Details

Merge creates a merged data frame from two input data frames.

If by is specified the merge is horizontal. That is the variables in the second input data frame are presumed different from the variables in the first input data frame. The merged data frame is the combination of variables from both input data frames, with the rows aligned by the value of by, an ID field common to both data frames.

If by is not provided, then the merge is vertical. The variables are presumed the same in each input data frame. The merged data frame consists of the rows of both input data frames. The rows of the first data frame are stacked upon the rows of the second data frame.

Guidance and feedback regarding the merge are provided by default. The first five lines of each of the input data frames are listed before the merge operation, followed by the first five lines of the output data frame.

Value

The merged data frame is returned, usually assigned the name of mydata as in the examples below. This is the default name for the data frame input into the lessR data analysis functions.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

See Also

merge, rbind.

Examples

```
# Horizontal
#-----------
mydata <- Read("Employee", format="lessR", quiet=TRUE)
Emp1a <- Subset(1:4, columns = c("Years", "Gender", "Dept", "Salary"))
Emp1b <- Subset(1:4, columns = c("Satisfaction", "HealthPlan"))
# horizontal merge
mydata <- Merge(Emp1a, Emp1b, by="row.names")
# suppress output to console
mydata <- Merge(Emp1a, Emp1b, by="row.names", quiet=TRUE)
```

```
# Vertical
#---------
mydata <- Read("Employee", format="lessR", quiet=TRUE)
Emp2a <- Subset(1:4)
Emp2b <- Subset(7:10)
# vertical merge
mydata <- Merge(Emp2a, Emp2b)
```

Model *Regression Analysis, ANOVA or t-test*

Description

Abbreviation: `model`, `model.brief`

Automatically selects and then provides an analysis of a linear model: OLS regression, Logistic regression, ANOVA, or a t-test depending on the proprieties of the data. Comprehensive regression analysis with graphics from a single, simple function call with many default settings, each of which can be re-specified. By default the data exists as a data frame with the default name of `mydata`, such as data read by the `lessR` `rad` function. Specify the model in the function call according to an R `formula`, that is, the response variable followed by a tilde, followed by the list of predictor variables, each pair separated by a plus sign.

Usage

```
Model(my.formula, data=mydata, brief=getOption("brief"), ...)

model.brief(..., brief=TRUE)

model(...)
```

Arguments

`my.formula`	Standard R `formula` for specifying a model. For example, for a response variable named Y and two predictor variables, X1 and X2, specify the corresponding linear model as Y ~ X1 + X2.
`data`	The default name of the data frame that contains the data for analysis is `mydata`, otherwise explicitly specify.
`brief`	If set to `TRUE`, reduced text output. Can change system default with `set` function.
`...`	Other parameter values for R functions such as `lm` which provide the core computations.

Details

OVERVIEW
The purpose of Model is to combine many standard R function calls into one, as well as provide ancillary analyses such as as graphics, organizing output into tables and sorting to assist interpretation of the output, all from a single function. Currently the supported models are OLS regression, ANOVA and the t-test. For more details of each of these methods, see the lessR functions Regression, Logit, ANOVA and ttest, respectively, which, in turn are based on many standard R functions.

All invocations of the model function are based on the standard R formula.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

See Also

formula, lm, glm, summary.lm, anova, confint, fitted, resid, rstudent, cooks.distance

Examples

```
# Generate random data, place in data frame mydata
n <- 200
X1 <- rnorm(n)
X2 <- rnorm(n)
Y <- .7*X1 + .2*X2 + .6*rnorm(n)
Ybin <- cut(Y, breaks=2, labels=FALSE)
#  instead, if read data with the Read function
#   then the result is the data frame called mydata
mydata <- round(data.frame(X1, X2, Y, Ybin),2)
rm(Y); rm(Ybin); rm(X1); rm(X2)

# One-predictor regression
# Provide all default analyses including scatterplot etc.
Model(Y ~ X1)
# alternate form
model(Y ~ X1)

# Multiple regression model
# Provide all default analyses
Model(Y ~ X1 + X2)

# Logit analysis
# Y is binary, 0 or 1
mydata <- Recode(Ybin, old=c(1,2), new=c(0,1), quiet=TRUE)
Model(Ybin ~ X1)

# t-test
Model(breaks ~ wool, data=warpbreaks)

# ANOVA analysis
# from another data frame other than the default \code{mydata}
```

```
# breaks is numerical, wool and tension are categorical
Model(breaks ~ wool + tension, data=warpbreaks)
```

Nest *Nest the Values of an Integer or Factor Variable*

Description

Abbreviation: nt

A nested model has a subset of predictor variables from the corresponding full model. Compare a nested linear model with a full model to evaluate the effectiveness of the predictor variables deleted from the full model to define the nested model.

Usage

```
Nest(y, nested.model, full.model, method=c("ls", "logit"), data=mydata)

nt(...)
```

Arguments

y	Response variable.
nested.model	Predictor variables in the nested model.
full.model	Predictor variables in the full model.
method	Do a least squares analysis, the default, or a logit analysis.
data	The name of the data frame from which to create the subset, which is mydata by default.
...	The specified arguments.

Details

Use the standard R function anova function to compare a nested model with a corresponding full model. By default, compare models estimated with ordinary least squares from the R function lm, or compare models estimated with logistic regression from the R function glm with family="binomial". For the logistic analysis, the anova analysis is with test="Chisq".

To insure that the same data are analyzed for both models, the fit for the full model is first obtained. Then the data frame that is returned by this analysis is input into the analysis for the nested model. This guarantees that any cases with missing data values missing for the full analysis will have been deleted for the nested analysis. Otherwise rows of data could be retained for the nested analysis that were dropped for the full analysis because of missing data values for the deleted predictor variables. This method also guarantees that cases are not deleted because data was missing on variables not included in full analysis.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

See Also

anova, lm, glm.

Examples

```
mydata <- Read("Reading", format="lessR")


# compare least-squares models
mydata <- Read("Reading", format="lessR")
Nest(Reading, c(Absent), c(Verbal,Absent,Income))

# compare logistic models
mydata <- Read("BodyMeas", format="lessR")
Nest(Gender, c(Weight, Hips, Hand, Shoe),
      c(Height, Weight, Waist, Hips, Chest, Hand, Shoe), method="logit")
```

PieChart *Pie Chart*

Description

Abbreviation: pc

Plots a pie chart with default colors from a variety of different types of data. Also displays the frequency table for the variable and provides the corresponding chi-square inferential analysis.

Usage

```
PieChart(x, data=mydata,
         col.fill=NULL, col.low=NULL, col.hi=NULL,
         colors=c("rainbow", "terrain", "heat"),
         random.col=FALSE, main=NULL, cex=1, cex.main=1,
         quiet=getOption("quiet"),
         pdf.file=NULL, pdf.width=5, pdf.height=5, ...)

pc(...)
```

Arguments

x	For each level of this variable, x, display the frequencies.
data	Optional data frame that contains the variable(s) of interest, default is mydata.
col.fill	Specified color of each slice.
col.low	Only when the variable is an ordered factor, sets the color for the lowest level of the factor in the resulting ordered progression of colors.
col.hi	Only when the variable is an ordered factor, sets the color for the highest level of the factor in the resulting ordered progression of colors.

colors	Optional palettes that provide more options.
random.col	Randomizes the order of the colors within the chosen color palette, when the second variable is not ordered, otherwise is ignored. When TRUE, each run of the same function call generally yields different colors of the slices
main	Title of graph.
cex	Magnification factor of labels relative to 1.
cex.main	Magnification factor of title relative to 1.
quiet	If set to TRUE, no text output. Can change system default with set function.
pdf.file	Name of the pdf file to which graphics are redirected.
pdf.width	Width of the pdf file in inches.
pdf.height	Height of the pdf file in inches.
...	Other parameter values.

Details

OVERVIEW
Plot a pie chart with default colors for one or two variables, presumably with a relatively small number of values for each variable. By default, colors are selected for the slices, background and grid lines, all of which can be customized. The basic computations of the chart are provided with the standard R functions pie and chisq.test and the lessR function chisq.test.

COLORS
There are three ways to override the default colors.
1. There are two pre-defined color palettes, each with 7 colors. The default palette provides lighter, more pastel colors. The vivid palette, activated by color="vivid", provides brighter colors with a brighter background (cornsilk1). A third color palette, set by color="gray", provides an ordered gray scale. Three more built-in R color palettes are also available by setting color to one of "rainbow", "heat" and "terrain". The most vivid of all the palettes is "rainbow".
2. The order of the colors within the chosen palette can be randomized with the random.col="TRUE" option. For example, when this option is activated each of the seven colors in a palette has a 1/7 chance of appearing as the first color, the only color used in the plot of a single variable. When invoked for a colors="gray", the order from light to dark will generally be lost, which may be desirable if the categories do not represent an ordered factor.
3. The desired colors can be explicitly specified with the col.fill option, which overrides any other color options. When plotting one variable, include one color in this color list, the color used for all of the slices As always with R, if the list includes more than once color, the c function must be used to generate the list, as in
col.fill=c("coral3","seagreen3").

The colors provided by the gray color palette are shades of gray. The default colors in the other two provided color palettes can be viewed, in the order in which they are displayed, by running the corresponding two lines of R code, first for the default colors and second for the vivid colors:

clr <- c("slategray3", "bisque3", "darksalmon", "darkolivegreen3", "thistle", "azure3", "moccasin")
barplot(rep(1,7), names=clr, col=clr, border=NA, space=.1)

clr <- c("coral3", "seagreen3", "maroon3", "dodgerblue3", "purple3", "turquoise3", "yellow3")
barplot(rep(1,7), names=clr, col=clr, border=NA, space=.1)

When plotting ordered factor then neither of the two standard color palettes are used. Instead, the resulting slice colors for each level of the ordered factor are also ordered in a progression of colors. The default progression is based on the first color of either the regular, vivid or gray color palettes, but this can be changed with the `col.low` and `col.hi` options, or individually specify the color of each piece with the `col.fill` option. A specified palette can, for example, be from light to dark of the same hue, or from a light color of one hue to a dark color of another hue. Each color value can be specified with a color name, or with a specification with the `rgb` function. See the examples below.

Use the `showColors` function in this package to get, for each color: name, sample color swatch, and corresponding rgb specification. For a very small number of levels, such as two, it is may be desirable to specify the low and high values to not be closer to each other than the default values.

STATISTICS
In addition to the pie chart, descriptive and optional inferential statistics are also presented. First, the frequency table with proportions is displayed. Second, the corresponding chi-square test is also displayed.

PDF OUTPUT
Because `lessR` functions generate their own graphics calls, the standard graphic output functions such as `pdf` do not work with the `lessR` graphics functions. Instead, to obtain pdf output, use the `pdf.file` option, perhaps with the optional `pdf.width` and `pdf.height` options. These files are written to the default working directory, which can be explicitly specified with the R `setwd` function.

ONLY VARIABLES ARE REFERENCED
The referenced variable in a `lessR` function can only be a variable name. This referenced variable must exist in either the referenced data frame, `mydata` by default, or in the user's workspace, more formally called the global environment. That is, expressions cannot be directly evaluated. For example:

```
> PieChart(rnorm(50))   # does NOT work
```

Instead, do the following:

```
> Y <- rnorm(50)   # create vector Y in user workspace
> PieChart(Y)     # directly reference Y
```

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

See Also

`pie`, `chisq.test`.

Examples

```
# ------------------------------------------------------
# generate some data in data frame mydata for two variables
# ------------------------------------------------------

# Pain is an ordered factor, Gender is an unordered factor
# Place in data frame mydata to simulate reading with rad
```

```
Pain <- sample(c("None", "Some", "Much", "Massive"), size=25, replace=TRUE)
Pain <- factor(Pain, levels=c("None", "Some", "Much", "Massive"), ordered=TRUE)
Gender <- sample(c("Male", "Female"), size=25, replace=TRUE)
Gender <- factor(Gender)
mydata <- data.frame(Pain, Gender)
rm(Pain); rm(Gender)


# ------------------------------------------
# pie chart from the data for a single variable
# ------------------------------------------

# for each level of Pain, display the frequencies
# Pain is an ordered factor, so the slice colors are ordered
PieChart(Pain)
# short name
pc(Pain)
# compare to standard R pie chart, which requires mydata$ reference
pie(table(mydata$Pain))

# Gender is unordered, so a different color for each slice
PieChart(Gender)

# specify a unique slice color for each of the two slices
PieChart(Gender, col.fill=c("pink","lightblue"))

# specify the colors from the R palette rainbow.colors
PieChart(Gender, colors="rainbow")


# ------------------------------
# pie chart directly from counts
# ------------------------------

# pie chart of one variable with three levels
# enter counts as a vector with the combine function, c
# must supply the level names and variable name
City <- c(206, 94, 382)
names(City) <- c("LA","Chicago","NY")
PieChart(City, main="Employees in Each City")
```

---

prob.norm *Compute and Plot Normal Curve Probabilities over an Interval*

Description

Calculate the probability of an interval for a normal distribution with specified mean and standard deviation, providing both the numerical probability and a plot of the interval with the corresponding normal curve.

Usage

```
prob.norm(lo=NULL, hi=NULL, mu=0, sigma=1, col.nrm="black",
         col.fill.nrm="grey91", col.fill.int="slategray3",
         ylab="", y.axis=FALSE, z=TRUE, mag=.9, ...)
```

Arguments

lo	Lowest value in the interval for which to compute probability.
hi	Highest value in the interval for which to compute probability.
mu	Population mean of normal distribution.
sigma	Population standard deviation of normal distribution.
col.nrm	Color of the border of the normal curve.
col.fill.nrm	Fill color of the normal curve.
col.fill.int	Fill color of the interval for which the probability is computed.
ylab	Label for the optional vertical axis.
y.axis	If TRUE, then a vertical axis is included.
z	If TRUE, then include z-values on the horizontal-axis. Set to FALSE if mu=0 and sigma=1.
mag	Magnification factor for the axis labels, the value of cex.axis.
...	Other parameter values for graphics.

Details

Calculate the normal curve probability for the specified interval and normal curve. If there is no upper value of the interval provided, hi, then the upper tail probability is provided, that is, from the specified value until positive infinity. If there is no lower value, lo, then the lower tail probability is provided. The probability is calculated with pnorm.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

See Also

pnorm, plot.

Examples

```
# Mu=0, Sigma=1: Standard normal prob, values between 0 and 2
prob.norm(0,2)

# Mu=0, Sigma=1: Standard normal prob, values lower than 2
prob.norm(hi=2)

# Mu=0, Sigma=1: Standard normal prob, values larger than 2
prob.norm(lo=2)
```

```
# Mu=100, Sigma=15: Change default fill color of plotted interval
prob.norm(lo=115, hi=125, mu=100, sigma=15, col.fill.int="plum")
```

`prob.tcut` *Plot t-distribution Curve and Specified Cutoffs with Normal Curve*

Description

Plot a specified t-distribution against the standardized normal curve with the corresponding upper and lower tail cutoffs.

Usage

```
prob.tcut(df, alpha=0.05, dig.dec=3, y.axis=FALSE,
         col.fill="aliceblue", col.tail="palevioletred4",
         col.nrm=gray(.7), col.t=gray(.08), ...)
```

Arguments

`df`	Degrees of freedom for t-distribution, must be 2 or larger.
`alpha`	Alpha to define the tail cutoff area.
`dig.dec`	Number of decimal digits in the output.
`y.axis`	If `FALSE`, then the y axis is not displayed.
`col.fill`	Fill color for the interior of the t-distribution curve.
`col.tail`	Color of the tail areas of the t-distribution.
`col.nrm`	Color of the normal curve.
`col.t`	Color of the t-distribution curve.
`...`	Other parameter values for graphics.

Details

Replaces a t-table by providing the corresponding t-cutoff, the critical value based on the corresponding quantile, as well as a plot that illustrates the tail probabilities. Also compare to the standardized normal curve.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

See Also

`qt`, `pnorm`.

Examples

```
# t-distribution with 0.025 cutoffs for degrees of freedom of 15
prob.tcut(15)
```

prob.znorm *Plot a Normal Curve with Shaded Intervals by Standard Deviation*

Description

Display a normal curve with shading according to the z-score, the number of standard deviations from the mean.

Usage

```
prob.znorm(mu=0, sigma=1, col.border="gray10",
         r=.10, g=.34, b=.94, a=.20,
         xlab="", ylab="", main="",
         y.axis=FALSE, z=TRUE, mag=.9, ...)
```

Arguments

mu	Population mean of normal distribution.
sigma	Population standard deviation of normal distribution.
col.border	Color of the border of the normal curve.
r	Red component of fill color, from 0 to 1.
g	Green component of fill color, from 0 to 1.
b	Blue component of fill color, from 0 to 1.
a	Alpha component of fill color, that is, the transparency, from 0 to 1.
xlab	Label for the horizontal axis.
ylab	Label for the optional vertical axis.
main	Label for the graph title.
y.axis	If TRUE, then a vertical axis is included.
z	If TRUE, then include z-values on the horizontal-axis. Set to FALSE if mu=0 and sigma=1.
mag	Magnification factor for the axis labels, the value of cex.axis.
...	Other parameter values for graphics.

Details

Provide a normal curve with shading of each interval defined by the number of standard deviations from the mean. The layers are written with transparency, and over-written so that the middle interval is the darkest and the most extreme intervals, beyond three standard deviations from the mean, are the lightest. Specify a=0 to turn off the colors. Higher values of the alpha channel, as specified by a, yield darker colors. Specify a=1 for the same solid color for all intervals.

The normal densities are calculated with dnorm and plotted with plot.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

See Also

dnorm, plot.

Examples

```
# Mu=0, Sigma=1: Standard normal
prob.znorm()

# distribution for height of American women, mu=65.5, sigma=2.5
prob.znorm(65.5, 2.5, xlab="Height of American Women")

# do a red fill color
prob.znorm(65.5, 2.5, r=.9, xlab="Height of American Women")
```

Read — *Read Contents of a Data File with Optional Variable Labels and Feedback*

Description

Abbreviation: rd, rd.brief, Read2

Reads the contents of the specified data file with optional variable labels into an R data table (frame). By default the format of the file is detected from its filetype: comma or tab separated value text file from .csv, SPSS data file from .sav, or R data file from .rda, and, if Perl is installed, Excel file from .xls or .xlsx using the gdata package. If no filetype is recognized then Read defaults to reading a comma separated or tab-limited text data file. Specify a fixed width formatted text data file to be read with the required R widths option. Identify the data file by either browsing for the file on the local computer system with Read(), or identify the file with the first argument a character string in the form of a path name or a web URL.

Variable labels can be added to the data table when reading a text file. Any variable labels in a native SPSS of native R file are automatically included. See the details section below for more information. Variable labels can also be added and modified individually with the lessR function label.

The function also provides feedback regarding the data that is read, which includes the variable names, the dimensions of the resulting data frame, the data type for each variable, and the values of the variables in the data file for the first and last rows of the data. In addition, an analysis of missing data is provided, listing the number of missing values for each variable and for each observation. The brief form just lists the input files, the variable name table, and any variable labels.

Also see the lessR function corRead to read a correlation matrix.

Usage

```
Read(ref=NULL, format=c("csv", "SPSS", "R", "Excel", "lessR"),

         labels=NULL, widths=NULL, missing="", n.mcut=1,

         miss.show=30, miss.zero=FALSE, miss.matrix=FALSE,

         max.lines=30, sheet=1,

         brief=getOption("brief"), quiet=getOption("quiet"), ...)

rd(...)

rd.brief(..., brief=TRUE)

Read2(..., sep=";", dec=",")
```

Arguments

`ref`	File reference, either omitted to browse for the data file, or a full path name or web URL, included in quotes. A URL begins with `http://`.
`format`	Format of the data in the file, which by default is a `csv` file, which also will recognize tab-delimited text. As an option can be an Excel `.xls` or `.xlsx` file or an SPSS `.sav` file, which also reads the variable labels if present, or a native R data file with a file type of `.rda`, or a (native R) data file part of `lessR`.
`labels`	File name for the file of variable labels. Either a full path name, or just the file name if in the same directory as the data file, or no reference between the quotes, which allows the user to browse for the labels file. Or, if `row2`, then the labels are in the second line of the data file.
`widths`	Specifies the width of the successive columns for fixed width formatted data.
`missing`	Missing value code, which by default is literally a missing data value in the data table.
`n.mcut`	For the missing value analysis, list the row name and number of missing values if the number of missing exceeds or equals this cutoff.
`miss.show`	For the missing value analysis, the number of rows, one row per observation, that has as many or missing values as `n.mcut`.
`miss.zero`	For the missing value analysis, list the variable name or the row name even for values of 0. By default only variables and rows with missing data are listed.
`miss.matrix`	For the missing value analysis, if there is any missing data, list a version of the complete data table with a 0 for a non-missing value and a 1 for a missing value.
`sep`	Character that separates adjacent values in a text file of data.
`dec`	Character that serves as the decimal separator in a number.
`max.lines`	Maximum number of lines to list of the data and labels.
`sheet`	For Excel files, specifies the work sheet to read. The default is the first work sheet.

brief	If TRUE, display only variable names table plus any variable labels.
quiet	If set to TRUE, no text output. Can change the corresponding system default with set function.
...	Other parameter values define with the R read functions, such as the read.table function for text files, with row.names and header.

Details

By default Read reads text data files which are either comma delimited, csv, or tab-dilimited data files, native Excel files of type .xls or .xlsx, native R files with file type of .rda and native SPSS files with file type .sav. Invoke the widths option to allow for the reading of fixed width formatted data. Calls the lessR function details to provide feedback regarding details of the data frame that was read.

CREATE csv FILE
One way to create a csv data file is to enter the data into a text editor. A more structured method is to use a worksheet application such as MS Excel, LibreOffice Calc. Place the variable names in the first row of the worksheet. Each column of the worksheet contains the data for the corresponding variable. Each subsequent row contains the data for a specific observation, such as for a person or a company.

All numeric data in the worksheet should be displayed in the General format, so that the only non-digit character for a numeric data value is a decimal point. The General format removes all dollar signs and commas, for example, leaving only the pure number, stripped of these extra characters which R will not properly read as part of a numeric data value.

To create the csv file from a standard worksheet application such as Microsoft Excel or LibreOffice Calc, first convert any numeric data to general format to remove characters such as dollar signs and commas, and then under the File option, do a Save As and choose the csv format.

Call help(read.table) to view the other options that can also be implemented from Read.

MECHANICS
Specify the file as with the Read function for reading the data into a data frame. If no arguments are passed to the function, then interactively browse for the file. Or, enclose within quotes a full path name or a URL for reading the labels on the web.

Given a csv data file, or tab-delimited text file, read the data into an R data frame called mydata with Read. Because Read calls the standard R function read.csv, which serves as a wrapper for read.table, the usual options that work with read.table, such as row.names, also can be passed through the call to Read.

SPSS DATA
Relies upon read.spss from the foreign package. To read data in the SPSS .sav format. If the file has a file type of .sav, that is, the file specification ends in .sav, then the format is automatically set to "SPSS". To invoke this option for a relevant data file of any file type, explicitly specify format="SPSS". Any variable labels in the SPSS file are read and stored in the resulting R data table (frame).

R DATA
Relies upon the standard R function load. By convention only, data files in native R format have a file type of .rda. To read a native R data file, if the file type is .rda, the format is automatically set to "R". To invoke this option for a relevant data file of any file type, explicitly specify format="R".

Create a native R data file by saving the current data frame, usually `mydata`, with the `lessR` function `Write`.

Excel DATA
Relies upon the function `read.xls` from the `gdata` package by Gregory Warnes and others. Files with a file type of `.xls` or `.xlsx` are assigned a `format` of `"Excel"`. The default worksheet to read from the file is the first worksheet. The `read.xls` parameter `sheet` specifies the ordinal position of the worksheet in the Excel file. The `read.xls` function relies upon the `Perl` scripting language, which must be accessible to R. This language is typically found on Macintosh and Linux/Unix systems but must usually be installed on a Windows system.

To install on Windows and make accessible to R:
1. Download and install Perl from: `http://strawberryperl.com/`
Install the 64 bit version if running the 64 bit version of R.
2. Enter the following function calls into the R console:

```
library(gdata)
installXLSXsupport(perl = "C:\\strawberry\\perl\\bin\\perl.exe")
```

3. Restart R
If the `gdata` package is updated, then the `installXLSXsupport` must be re-run.

lessR DATA
`lessR` has some data sets included with the package. `Read` reads each such data set by specifying its name and setting `format="lessR"`. Also, each included data set begins with the prefix `dat`, which can be deleted when specifying the name of the data set. This option is a replacement for the standard R `data` function, offering the added information provided by `Read`.

FIXED WIDTH FORMATTED DATA
Relies upon `read.fwf`. Applies to data files in which the width of the column of data values of a variable is the same for each data value and there is no delimiter to separate adjacent data values. An example is a data file of Likert scale responses from 1 to 5 on a 50 item survey such that the data consist of 50 columns with no spaces or other delimiter to separate adjacent data values. To read this data set, invoke the `widths` option of `read.fwf`.

MISSING DATA
By default, `Read` provides a list of each variable and each row with the display of the number of associated missing values, indicated by the standard R missing value code NA. When reading the data, `Read` automatically sets any empty values as missing. Note that this is different from the R default in `read.table` in which an empty value for character string variables are treated as a regular data value. Any other valid value for any data type can be set to missing as well with the `missing` option. To mimic the standard R default for missing character values, set `missing=NA`.

To not list the variable name or row name of variables or rows without missing data, invoke the `miss.zero=FALSE` option, which can appreciably reduce the amount of output for large data sets. To view the entire data table in terms of 0's and 1's for non-missing and missing data, respectively, invoke the `miss.matrix=TRUE` option.

VARIABLE LABELS
Unlike standard R, `lessR` provides for variable labels, which can be provided for some or all of the variables in a data frame. One way to enter the variable labels is to read them from their own file with `Read` with `labels` set to the full path name or `URL` of the labels file, or just the file name if the labels file is in the same directory as the data file. The user browses for the labels file if `label=""`. Another method is to include the labels directly in text data file. To to this, specify the label of variable labels with the `label="row2"` option. The web survey application Qualtrics downloads `csv` files in this format.

These labels options work for csv files and Excel files, identified by the filetypes .xls or .xlsx. Reading from an Excel file, however, requires the use of the read.xls funciton from the gdata package, which also requires that the scripting language Perl be installed. See the Excel DATA section above for more information.

Variable labels in an SPSS data file or R data file are automatically read into the corresponding R data frame. The labels are stored within the data frame, so if the data frame is written to an external file as a native R data file, the labels are also written as part of that file.

For a file that contains only labels, each row of the file, including the first row, consists of the variable name, a comma if a csv file, and then the label. For the csv form of the file, this is the standard csv format such as obtained with the csv option from a standard worksheet application such as Microsoft Excel or LibreOffice Calc. Not all variables in the data frame that contains the data, usually mydata, need have a label, and the variables with their corresponding labels can be listed in any order. An example of this file follows for four variables, I1 through I4, and their associated labels.

I2,This instructor presents material in a clear and organized manner.
I4,Overall, this instructor was highly effective in this class.
I1,This instructor has command of the subject.
I3,This instructor relates course materials to real world situations.

If there is a comma in the variable label, then the label needs to be enclosed in quotes.

The lessR functions that provide analysis, such as Histogram for a histogram, automatically include the variable labels in their output, such as the title of a graph. Standard R functions can also use these variable labels by invoking the lessR function label, such as setting main=label(I4) to put the variable label for a variable named I4 in the title of a graph.

Value

The read data frame is returned, usually assigned the name of mydata as in the examples below. This is the default name for the data frame input into the lessR data analysis functions.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

References

Gerbing, D. W. (2013). R Data Analysis without Programming, Chapter 2, NY: Routledge.

Gregory R. Warnes, Ben Bolker, Gregor Gorjanc, Gabor Grothendieck, Ales Korosec, Thomas Lumley, Don MacQueen, Arni Magnusson, Jim Rogers and others (2013). gdata: Various R programming tools for data manipulation. R package version 2.13.2. http://CRAN.R-project.org/package=gdata

See Also

read.csv,read.spss, read.fwf, corRead, label, details.

Examples

```
# remove the # sign before each of the following Read statements to run

# to browse for a data file on the computer system, invoke Read with
#   the ref argument empty
# mydata <- Read()
# abbreviated name
# mydata <- rd()
# reduced output to the console
# mydata <- rd.brief()

# browse for a file and then read the variable labels from
#  the specified label file, here a Excel file with two columns,
#  the first column of variable names and the second column the
#  corresponding labels
# mydata <- Read(labels="employee_lbl.xlsx")

# same as above, but include standard read.csv options to indicate
#  no variable names in first row of the csv data file
#   and then provide the names
# also indicate that the first column is an ID field
# mydata <- Read(header=FALSE, col.names=c("X", "Y"), row.names=1)

# read a csv data file from the web
# mydata <- Read("http://web.pdx.edu/~gerbing/data/twogroup.csv")

# read a csv data file with -99 and XXX set to missing
# mydata <- Read(missing=c(-99, "XXX"))

# do not display any output
# mydata <- Read(quiet=TRUE)

# read tab-delimited (or any other white-space) data
# mydata <- Read(sep="")

# read the built-in data set datEmployee
mydata <- Read("Employee", format="lessR")

# read a data file that consists of a
#   5 column ID field, 2 column Age field
#   and 75 single columns of data, no spaces between columns
#   name the variables with lessR function: to
#   the variable names are Q01, Q02, ..., Q74, Q75
# mydata <- Read(widths=c(5,2,rep(1,75)), col.names=c("ID", "Age", to("Q", 75)))
```

Recode *Recode the Values of an Integer or Factor Variable*

Description

Abbreviation: rec

Recodes the values of one or more integer variables in a data frame. The values of the original variable may be overwritten with the recoded values, or the recoded values can be designated to be placed in a new variable, indicated by the new.name option. Valid values may be converted to missing, and missing values may be converted to valid values. Any existing variable labels are retained in the recoded data frame.

There is no provision to recode integer values to character strings because that task is best accomplished with the standard R factor function.

Usage

```
Recode(old.vars, new.vars=NULL, old, new, data=mydata,
       quiet=getOption("quiet"))

rec(...)
```

Arguments

old.vars	One or more variables to be recoded.
new.vars	Name of the new variable or variables that contain the recoded values, each name in quotes. If not provided, then the values of the original variable are replaced.
old	The values of the variables that are to be recoded. If the value is "missing" then any existing missing values are replaced by the value specified with new.
new	The recoded values, which match one-to-one with the values in old. If the value is "missing" then instead any values specified in old are converted to missing.
data	The name of the data frame from which to create the subset, which is mydata by default.
quiet	If set to TRUE, no text output. Can change system default with set function.
...	Parameter values.

Details

Specify the values to be recoded with the required old parameter, and the corresponding recoded values with the required new parameter. Use new.vars to specify the name of the variable that contains the recoded values. If new.vars is not present, then the values of the original variable are overwritten with the recoded values.

Not all of the existing values of the variable to be recoded need be specified. Any value not specified is unchanged in the values of the recoded variable. Unless otherwise specified, missing values are unchanged. To modify missing values, set old="missing" to covert missing data values to the specified value data value given in new. Or, set new="missing" to covert the one or more existing valid data values specified in old to missing data values.

Diagnostic checks are performed before the recode. First, it is verified that the same number of values exist in the old and new lists of values. Second, it is verified that all of the values specified to be recoded in fact exist in the original data.

If the levels of a factor were to be recoded with Recode, then the factor attribute would be lost as the resulting recoded variable would be character strings. Accordingly, this type of transformation is not allowed, and instead should be accomplished with the Transform and factor functions as shown in the examples.

Value

The recoded data frame is returned, usually assigned the name of mydata as in the examples below. This is the default name for the data frame input into the lessR data analysis functions.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

See Also

transform, factor.

Examples

```
# construct data frame
mydata <- read.table(text="Severity Description
1 Mild
4 Moderate
3 Moderate
2 Mild
1 Severe", header=TRUE)

# recode Severity into a new variable called SevereNew
mydata <- Recode(Severity, new.vars="SevereNew", old=1:4, new=c(10,20,30,40))

# abbreviated form,  replace original with recoded
# another option, the sequence function, to generate list of values
mydata <- rec(Severity, old=1:4, new=seq(10,40,by=10))

# reverse score four Likert variables: m01, m02, m03, m10
mydata <- Read("Mach4", format="lessR")
mydata <- Recode(c(m01:m03,m10), old=0:5, new=5:0)

# convert any 1 for HealthPlan to missing
# use Read to put data into mydata data frame
# write results to newdata data frame
mydata <- Read("Employee", format="lessR")
newdata <- Recode(HealthPlan, old=1, new="missing")

# for Years and Salary convert any missing value to 99
mydata <- Read("Employee", format="lessR")
mydata <- Recode(c(Years, Salary), old="missing", new=99)

# recode levels of a factor with the Transform and factor functions
# using Recode destroys the factor attribute, converting to
#   character strings instead, so Recode does not allow
```

```
mydata <- Read("Employee", format="lessR")
mydata <- Transform(
    Gender=factor(Gender, levels=c("F", "M"), labels=c("Female", "Male"))
)
```

`Regression` *Regression Analysis*

Description

Abbreviation: `reg`, `reg.brief`, `reg.explain`

Provides a comprehensive regression analysis with graphics from a single, simple function call with many default settings, each of which can be re-specified. By default the data exists as a data frame with the default name of `mydata`, such as data read by the `lessR` function `Read`. Specify the model in the function call according to an R `formula`, that is, the response variable followed by a tilde, followed by the list of predictor variables, each pair separated by a plus sign.

Default output includes the inferential analysis of the estimated coefficients and model, correlation matrix, sorted residuals and Cook's Distance, and sorted prediction intervals for existing data or new data. For multiple regression models, also included is an analysis of the fit of all possible model subsets and an analysis of collinearity. The default output also includes three graphs beginning with a histogram of the residuals with superimposed normal and general density curves. The second graph is a scatterplot of the fitted values with the residuals and the corresponding `lowess` curve. The point corresponding to the largest value of Cook's Distance is labeled accordingly. Also provided, for a model with one predictor variable, is a scatterplot of the data with regression line and confidence and prediction intervals. For multiple regression models the graph is the scatterplot matrix of the model variables with the `lowess` curve displayed for each constituent scatterplot. If the model has exactly two predictor variables, a 3D scatterplot about the regression plane is optionally produced.

Can also be called from the more general `Model` function.

The resulting scatterplot or scatterplot matrix, when written to a pdf file according to `pdf=TRUE`, is named RegScatterplot.pdf or RegScatterMatrix.pdf. If residuals are reported, then the two additional pdf files are named RegResiduals.pdf and RegResidFitted.pdf. Their names and the directory to which they are written are provided as part the console output.

Usage

```
Regression(my.formula, data=mydata, digits.d=NULL, standardize=FALSE,
         text.width=120, brief=getOption("brief"), explain=FALSE, show.R=FALSE,

         res.rows=NULL, res.sort=c("cooks","rstudent","dffits","off"),
         pred.rows=NULL, pred.sort=c("predint", "off"),
         subsets=TRUE, cooks.cut=1,

         scatter.coef=FALSE, scatter.3D=FALSE,

         X1.new=NULL, X2.new=NULL, X3.new=NULL, X4.new=NULL,
```

```
          X5.new=NULL,

          pdf=FALSE, pdf.width=5, pdf.height=5, refs=FALSE, ...)

reg(...)

reg.brief(..., brief=TRUE)

reg.explain(..., explain=TRUE)
```

Arguments

`my.formula`	Standard R `formula` for specifying a model. For example, for a response variable named Y and two predictor variables, X1 and X2, specify the corresponding linear model as Y ~ X1 + X2.
`data`	The default name of the data frame that contains the data for analysis is `mydata`, otherwise explicitly specify.
`digits.d`	For the Basic Analysis, it provides the number of decimal digits, set by default to at least 3 or the largest number of digits in the values of the response variable plus 1.
`standardize`	Standardize each of the variables in the regression model before conducting the analysis.
`text.width`	Width of the text output at the console.
`brief`	If set to `TRUE`, reduced text output. Can change system default with `set` function.
`explain`	Off by default, but if set to `TRUE` then brief labels of different sections of output are replaced by more complete explanations of the output.
`show.R`	Display the R instructions that yielded the `lessR` output, albeit without the additional formatting of the results such as combining output of different functions into a table.
`res.rows`	Default is 20, which lists the first 20 rows of data sorted by the specified sort criterion. To disable residuals, specify a value of 0. To see the output for all observations, specify a value of `"all"`.
`res.sort`	Default is `"cooks"`, for specifying Cook's distance as the sort criterion for the display of the rows of data and associated residuals. Other values are `"rstudent"` for externally Studentized residuals, `"dffits"` for dffits and `"off"` to not sort the rows of data.
`pred.rows`	Default is 4, which lists prediction intervals only for the first, middle and last 4 rows of data, unless there are 25 or less rows of data when all rows are displayed. To disable prediction intervals, specify a value of 0. To see the output for rows of data, specify a value of `"all"`.
`pred.sort`	Default is `"predint"`, which sorts the rows of data and associated intervals by the lower bound of each prediction interval. Turn off this sort by specifying a value of `"off"`.
`subsets`	Default is to produce the analysis of the fit based on adjusted R-squared for all possible model subsets from the `leaps` package. Set to `FALSE` to turn off.

cooks.cut	Cutoff value of Cook's Distance at which observations with a larger value are flagged in red and labeled in the resulting scatterplot of Residuals and Fitted Values. Default value is 1.0.
scatter.coef	Display the correlation coefficients in the upper triangle of the scatterplot matrix.
scatter.3D	A 3D scatterplot with best fitting regression plane, which applies only to models with exactly two predictor variables. To obtain, set to TRUE.
X1.new	Values of the first listed predictor variable for which forecasted values and corresponding prediction intervals are calculated.
X2.new	Values of the second listed predictor variable for which forecasted values and corresponding prediction intervals are calculated.
X3.new	Values of the third listed predictor variable for which forecasted values and corresponding prediction intervals are calculated.
X4.new	Values of the fourth listed predictor variable for which forecasted values and corresponding prediction intervals are calculated.
X5.new	Values of the fifth listed predictor variable for which forecasted values and corresponding prediction intervals are calculated.
pdf	If TRUE, then graphics are written to pdf files.
pdf.width	Width of the pdf file in inches.
pdf.height	Height of the pdf file in inches.
refs	If TRUE, then list the references for R and the packages used from which functions were used to generate the output.
...	Other parameter values for R function lm which provides the core computations.

Details

OVERVIEW

The purpose of Regression is to combine the following function calls into one, as well as provide ancillary analyses such as as graphics, organizing output into tables and sorting to assist interpretation of the output.

The basic analysis successively invokes several standard R functions beginning with the standard R function for estimation of a linear model, lm. The output of the analysis of lm is stored in the object lm.out, available for further analysis in the R environment upon completion of the Regression function. By default reg automatically provides the analyses from the standard R functions, summary, confint and anova, with some of the standard output modified and enhanced. The correlation matrix of the model variables is obtained with cor function. The residual analysis invokes fitted, resid, rstudent, and cooks.distance functions. The option for prediction intervals calls the standard R function predict, once with the argument interval="confidence" and once with interval="prediction". The lessR Density function provides the histogram and density plots for the residuals and the ScatterPlot function provides the scatter plots of the residuals with the fitted values and of the data for the one-predictor model. The pairs function provides the scatterplot matrix of all the variables in the model. Thomas Lumley's leaps package contains the leaps function that provides the analysis of the fit of all possible model subsets. The car package provides Henric Nilsson and John Fox's vif function for the computation of the variance inflation factors for the collinearity analysis. The scatter3d function from Fox and Weisberg's car package provides the interactive 3d scatterplot for models with exactly two predictor variables.'

The default analysis provides as text output to the console the model's parameter estimates and corresponding hypothesis tests and confidence intervals, goodness of fit indices, the ANOVA table, correlation matrix of the model's variables, analysis of residuals and influence as well as the confidence and prediction intervals for each observation in the model. Also provided, for multiple regression models, collinearity analysis of the predictor variables and adjusted R-squared for the corresponding models defined by each possible subset of the predictor variables.

DATA FRAME

The name `mydata` is by default provided by the `Read` function included in this package for reading and displaying information about the data in preparation for analysis. If all the variables in the model are not in the same data frame, the analysis will not be complete. The data frame does not need to be attached, just specified by name with the `data` option if the name is not the default `mydata`.

DECIMAL DIGITS

The number of decimal digits displayed on the output is, by default, the maximum number of decimal digits for all the data values of the response variable. Or, this value can be explicitly specified with the `digits.d` parameter.

GRAPHICS

Three default graphs are provided. By default the graphs are written to separate graphics windows (which may overlap each other completely, in which case move the top graphics windows). Or, the `graphics.save` option may be invoked to save the graphs to a single pdf file called `regOut.pdf`. The directory to which the file is written is displayed on the console text output.

1. A histogram of the residuals includes the superimposed normal and general density plots from the `Density` function included in this `lessR` package. The overlapping density plots, which both overlap the histogram, are filled with semi-transparent colors to enhance readability.

2. A scatterplot of the residuals with the fitted values is also provided from the `ScatterPlot` function included in this package. The point corresponding to the largest value of Cook's distance, regardless of its size, is plotted in red and labeled and the corresponding value of Cook's distance specified in the subtitle of the plot. Also by default all points with a Cook's distance value larger than 1.0 are plotted in red, a value that can be specified to any arbitrary value with the `cooks.cut` option. This scatterplot also includes the `lowess` curve.

3. For models with a single predictor variable, a scatterplot of the data is produced, which also includes the regression line and corresponding confidence and prediction intervals. As with the density histogram plot of the residuals and the scatterplot of the fitted values and residuals, the scatterplot includes a colored background with grid lines. For multiple regression models, a scatterplot matrix of the variables in the model with the `lowess` best-fit line of each constituent scatterplot is produced. If the `scatter.coef` option is invoked, each scatterplot in the upper-diagonal of the correlation matrix is replaced with its correlation coefficient.

RESIDUAL ANALYSIS

By default the residual analysis lists the data and fitted value for each observation as well as the residual, Studentized residual, Cook's distance and dffits, with the first 20 observations listed and sorted by Cook's distance. The `res.sort` option provides for sorting by the Studentized residuals or not sorting at all. The `res.rows` option provides for listing these rows of data and computed statistics statistics for any specified number of observations (rows). To turn off the analysis of residuals, specify `res.rows=0`.

PREDICTION INTERVALS

The output for the confidence and prediction intervals includes a table with the data and fitted value

for each observation, the lower and upper bounds for the confidence interval and the prediction interval, and the wide of the prediction interval. The observations are sorted by the lower bound of each prediction interval. If there are 25 or more observations then the information for only the first four, the middle four and the last four observations is displayed. To turn off the analysis of prediction intervals, specify `pred=FALSE`, which also removes the corresponding intervals from the scatterplot produced with a model with exactly one predictor variable, yielding just the scatterplot and the regression line.

The data for the default analysis of the prediction intervals is for the values of the predictor variables for each observation, that is, for each row of the data. New values of the predictor variables can be specified for the calculation of the prediction intervals by providing values for the options `X1.new` for the values of the first listed predictor variable in the model, `X2.new` for the second listed predictor variable, and so forth for up to five predictor variables. To provide these values, use functions such as `seq` for specifying a sequence of values and `c` for specifying a vector of values. For multiple regression models, all combinations of the specified new values for all of the predictor variables are analyzed.

RELATIONS AMONG THE VARIABLES

By default the correlation matrix of all the variables in the model is displayed, and, for multiple regression models, collinearity analysis is provided with the `vif` function from the Fox and Weisberg (2011) `car` package. Also provided are the first 50 models with the largest R squared adjusted from each possible model from an analysis of all possible subsets of the predictor variables. This all subsets analysis requires the `leaps` function from the `leaps` package. These contributed packages are automatically loaded if available. To turn off the all possible sets option, set `subsets=FALSE`.

OUTPUT

The text output is organized to provide the most relevant information while at the same time minimizing the total amount of output, particularly for analyses with large numbers of observations (rows of data), the display of which is by default restricted to only the most interesting or representative observations in the analyses of the residuals and predicted values. Additional economy can be obtained by invoking the `brief=TRUE` option, or run `reg.brief`, which limits the analysis to just the basic analysis of the estimated coefficients and fit. An explanation of each section of the output can be obtained by setting the `explain` option to `TRUE`, or run `reg.explain`. Much of the underlying relevant R code run by `Regression` is obtained by setting the `show.R` option to `TRUE`.

INVOKED R OPTIONS

The `options` function is called to turn off the stars for different significance levels (show.signif.stars=FALSE), to turn off scientific notation for the output (scipen=30), and to set the width of the text output at the console to 120 characters. The later option can be re-specified with the `text.width` option. After `Regression` is finished with a normal termination, the options are re-set to their values before the `Regression` function began executing.

COLOR THEME

A color theme for all the colors can be chosen for a specific plot with the `colors` option. Or, the color theme can be changed for all subsequent graphical analysis with the `lessR` function `set`. The default color theme is `blue`, but a gray scale is available with `"gray"`, and other themes are available as explained in `set`.

VARIABLE LABELS

If variable labels exist, then the corresponding variable label is by default listed as the label for the horizontal axis and on the text output. For more information, see `Read`.

Value

Based on the standard R function lm, invisibly returns an object of class "lm". For details see lm. This object is particularly useful to compare nested models. Assign the output of Regression for a model to an object. Then for a nested model. Then use the standard R function anova to compare the models, as shown in the examples below.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

References

Fox, J., & Weisberg, S. (2011). An R companion to applied regression (Second ed.). Thousand Oaks CA: Sage.

Lumley, T., leaps function from the leaps package.
Nilsson, H. and Fox, J., vif function from the car package.

Gerbing, D. W. (2013). R Data Analysis without Programming, Chapters 9 and 10, NY: Routledge.

See Also

formula, lm, summary.lm, anova, confint, fitted, resid, rstudent, cooks.distance

Examples

```
# Generate random data, place in data frame mydata
X1 <- rnorm(20)
X2 <- rnorm(20)
Y <- .7*X1 + .2*X2 + .6*rnorm(20)
#  instead, if read data with the rad function
#   then the result is the data frame called mydata
mydata <- round(data.frame(X1, X2, Y),2)
rm(Y); rm(X1); rm(X2)

# One-predictor regression
# Provide all default analyses including scatterplot etc.
Regression(Y ~ X1)
# short name
reg(Y ~ X1)
# Provide only the brief analysis on the standardized variables
reg.brief(Y ~ X1, standardize=TRUE)
# Provide an explanation for each section of output
reg.explain(Y ~ X1)
# Provide a brief analysis with explanation
reg.brief(Y ~ X1, explain=TRUE)

# compare nested models
Nest(Y, X1, c(X1,X2))

# Modify the default settings as specified
```

```
Regression(Y ~ X1, res.row=8, res.sort="rstudent", pred.rows=0, digits.d=4)

# Multiple regression model
# Provide all default analyses
Regression(Y ~ X1 + X2)
# Save the three plots as pdf files 4 inches square
Regression(Y ~ X1, pdf=TRUE, pdf.width=4, pdf.height=4)

# Specify new values of the predictor variables to calculate
#  forecasted values and the corresponding prediction intervals
# Specify an input data frame other than mydata, see help(mtcars)
Regression(mpg ~ hp + wt + disp, data=mtcars,
  X1.new=seq(50,350,50), X2.new=c(2,3), X3.new=c(100,300))

# Plot the standardized residuals against the values of X
# Save the results of the analysis into an object, here called reg.out
reg.out <- reg(Y ~ X1)
# Use the R rstandard function to get the standardized residuals
res.stnd <- rstandard(reg.out)
# Plot with lessR LineChart function
LineChart(res.stnd, col.line="transparent", center.line="zero",
          xlab="X1", ylab="Standardized Residuals")
```

ScatterPlot *Scatterplot for One (Dot Plot) or Two Variables*

Description

Abbreviation: sp

Generates scatter plots for one or two variables. For two variables also produces an analysis of the correlation coefficient. If the values of the first specified value are sorted, then points are connected via line segments. The first variable can be numeric or a factor. The second variable must be numeric. For Likert style response data of two variables, so that each value has less than 10 unique integer values, the points in the plot are transformed into a bubble plot with the size of each bubble, i.e., point, determined by the corresponding joint frequency. An alternate name for ScatterPlot is just Plot.

One enhancement over the standard R plot function is the automatic inclusion of color. The color of the line segments and/or the points, background, area under the plotted line segments, grid lines, and border can each be explicitly specified, with default colors provided by one of the pre-defined color themes as defined by the set function.

If a scatterplot of two numeric variables is displayed, then the corresponding correlation coefficient as well as the hypothesis test of zero population correlation and the 95% confidence interval are also displayed. The same numeric values of the standard R function cor.test function are generated, though in a more readable format. Also, an option for the .95 data ellipse from John Fox's car package can enclose the points of the scatterplot.

For one variable, based on the standard R function stripchart, plots a one dimensional scatterplot, that is, a dot chart, also called a strip chart. Also identifies outliers according to the criteria specified

by a box plot and displays the summary statistics for the variable. The dot plot is also invoked with the function names DotPlot or just dp, which are just alternate names for ScatterPlot when a single variable is referenced.

Usage

```
ScatterPlot(x, y=NULL, by=NULL, data=mydata, type=NULL, n.cat=getOption("n.cat"),

          col.fill=getOption("col.fill.pt"),
          col.stroke=getOption("col.stroke.pt"),
          col.bg=getOption("col.bg"),
          col.grid=getOption("col.grid"),

          col.area=NULL, col.box="black",

          shape.pts="circle", cex.axis=.85, col.axis="gray30",
          col.ticks="gray30", xy.ticks=TRUE,
          xlab=NULL, ylab=NULL, main=NULL, cex=NULL,

          kind=c("default", "regular", "bubble", "sunflower"),

          fit.line=c("none", "loess", "ls"), col.fit.line="grey55",

          bubble.size=.25, method="overplot",

          ellipse=FALSE, col.ellipse="lightslategray", fill.ellipse=TRUE,

          pt.reg="circle", pt.out="circle",
          col.out30="firebrick2", col.out15="firebrick4", new=TRUE,

          diag=FALSE, col.diag=par("fg"), lines.diag=TRUE,

          quiet=getOption("quiet"),
          pdf.file=NULL, pdf.width=5, pdf.height=5, ...)

sp(...)

Plot(...)

DotPlot(...)
dp(...)
```

Arguments

x	If both x and y are specified, then the x values are plotted on the horizontal axis. If x is sorted, then the points are joined by line segments by default. If only x is specified with no y, then these x values are plotted as a dot chart, a one-dimensional scatterplot.
y	Coordinates of points in the plot on the vertical axis.

by	An optional grouping variable such that the points of all (x,y) pairs are plotted in the same plotting symbol and/or same color, with a different symbol or symbol and/or color for each group. Applies only to kind="regular" scatterplots.
data	Optional data frame that contains one or both of the variables of interest, default is mydata.
type	Character string that indicates the type of plot, either "p" for points, "l" for line, or "b" for both. If x and y are provided and x is sorted so that a function is plotted, the default is "l", or, when x is not sorted, the default is "p" for point, yielding a scatterplot.
n.cat	When analyzing all the variables in a data frame, specifies the largest number of unique values of variable of a numeric data type for which the variable will be analyzed as categorical. Set to 0 to turn off.
col.fill	For plotted points, the interior color of the points. By default, is a partially transparent version of the border color, col.stroke. Does not apply if there is a by variable, which relies upon the default.
col.stroke	Border color of the plotted points. If there is a by variable, specified as a vector, one value for each level of by.
col.bg	Color of the plot background.
col.grid	Color of the grid lines, with a default of "grey90".
shape.pts	The standard plot character, with values defined in points. The default value is 21, a circle with both a border and filled area, specified here with col.pts and col.fill.
col.area	Color of area under the plotted line segments.
col.box	Color of border around the plot background, the box, that encloses the plot, with a default of "black".
cex.axis	Scale magnification factor, which by defaults displays the axis values to be smaller than the axis labels.
col.axis	Color of the font used to label the axis values.
col.ticks	Color of the ticks used to label the axis values.
xy.ticks	Flag that indicates if tick marks and associated values on the axes are to be displayed.
xlab	Label for x-axis. For two variables specified, x and y, if xlab not specified, then the label becomes the name of the corresponding variable. If xy.ticks is FALSE, then no label is displayed. If no y variable is specified, then xlab is set to Index unless xlab has been specified.
ylab	Label for y-axis. If not specified, then the label becomes the name of the corresponding variable. If xy.ticks is FALSE, then no label displayed.
main	Label for the title of the graph. If the corresponding variable labels exist, then the title is set by default from the corresponding variable labels.
cex	Magnification factor for any displayed points, with default of cex=1.0.
kind	Default is "default", which becomes a "regular" scatterplot for most data. If Likert style response data is plotted, that is, each variable has less than 10 integer values, then instead by default a bubble plot is plotted with the corresponding joint frequency determining the size of the bubble. A sunflower plot can also be requested.

fit.line	The best fitting line. Default value is "none", with options for "loess" and "ls".
col.fit.line	Color of the best fitting line, if the fit.line option is invoked.
bubble.size	Size of the bubbles in a bubble plot of Likert style data.
method	Applies to one variable plots. Default is "overplot", but can also provide "stack" to stack the points or "jigger" to scramble the points.
ellipse	If TRUE, enclose a scatterplot with the .95 data ellipse from the car package.
col.ellipse	Color of the ellipse.
fill.ellipse	If TRUE, fill the ellipse with a translucent shade of col.ellipse.
pt.reg	For dot plot, type of regular (non-outlier) point. Default is 21, a circle with specified fill.
pt.out	For a dot plot, type of point for outliers. Default is 19, a filled circle.
col.out30	For a dot plot, color of outliers.
col.out15	For a dot plot, color of potential outliers.
new	If FALSE, then add the dot plot to an existing graph.
diag	Applies just to scatter plots of 2 numeric variables. If TRUE, then add a diagonal line to a 2-dimensional scatter plot.
col.diag	Color of diagonal line if diag=TRUE.
lines.diag	If lines.diag=TRUE, then if diag=TRUE, each point is connected to the diagonal line with a line segment.
quiet	If set to TRUE, no text output. Can change system default with set function.
pdf.file	Name of the pdf file to which graphics are redirected.
pdf.width	Width of the pdf file in inches.
pdf.height	Height of the pdf file in inches.
...	Other parameter values for graphics as defined by and then processed by plot and par, including xlim, ylim, lwd, and cex to specify a magnification factor for the plotting symbol. For one variable, parameters from stripchart. For type of correlation from cor.test, method="spearman" and method="kendall".

Details

DATA

The default input data frame is mydata. Specify another name with the data option. Regardless of its name, the data frame need not be attached to reference the variables directly by its name, that is, no need to invoke the mydata$name notation. The referenced variables can be in the data frame and/or the user's workspace, the global environment.

ADAPTIVE GRAPHICS

Results for two variables are based on the standard plot and related graphic functions, with the additional provided color capabilities and other options including a center line. The plotting procedure utilizes "adaptive graphics", such that ScatterPlot chooses different default values for different characteristics of the specified plot and data values. The goal is to produce a desired graph from simply relying upon the default values, both of the ScatterPlot function itself, as well as the base

R functions called by ScatterPlot, such as plot. Familiarity with the options permits complete control over the computed defaults, but this familiarity is intended to be optional for most situations.

TWO VARIABLE PLOT

When two variables are specified to plot, by default if the values of the first variable, x, are unsorted, or if there are unequal intervals between adjacent values, or if there is missing data for either variable, a scatterplot is produced, that is, a call to the standard R plot function with type="p" for points. By default, sorted values with equal intervals between adjacent values of the first of the two specified variables yields a function plot if there is no missing data for either variable, that is, a call to the standard R plot function with type="l", which connects each adjacent pair of points with a line segment.

BY VARIABLE

A variable specified with by= is a grouping variable that specifies that the plot is produced with the points for each group plotted with a different shape and/or color. By default, the shapes vary by group, and the color of the plot symbol remains the same for the groups. The default shapes, in this order, are "circle", "diamond", "square", "triup" for a triangle pointed up, and "tridown" for a triangle pointed down.

To explicitly vary the shapes, use shape.pts and a list of shape values in the standard R form with the c function to combine a list of values, one specified shape for each group, as shown in the examples. To explicitly vary the colors, use col.pts, such as with R standard color names. If col.pts is specified without shape.pts, then colors are varied, but not shapes. To vary both shapes and colors, specify values for both options, always with one shape or color specified for each level of the by variable.

Shapes beyond the standard list of named shapes, such as "circle", are also available as single characters. Any single letter, uppercase or lowercase, any single digit, and the characters "+", "*" and "#" are available, as illustrated in the examples. In the use of shape.pts, either use standard named shapes, or individual characters, but not both in a single specification.

SCATTERPLOT ELLIPSE

For a scatterplot of two numeric variables, the ellipse=TRUE option draws the .95 data ellipse as computed by the dataEllipse function, written by Georges Monette and John Fox, from the car package. Usually the minimum and maximum values of the axes should be manually extended beyond their default to accommodate the entire ellipse. To accomplish this extension, use the xlim and ylim options, such as xlim=c(30,350). Obtaining the desired axes limits may involve multiple runs of the ScatterPlot function. To provide more control over the display of the data ellipse beyond the provided col.ellipse and fill.ellipse options, run the dataEllipse function directly with the plot.points=FALSE option following ScatterPlot with ellipse=FALSE, the default.

ONE VARIABLE PLOT

The one variable plot is a one variable scatterplot, that is, a dot chart. Results are based on the standard stripchart function. Colors are provided by default and can also be specified. For gray scale output, potential outliers are plotted with squares and actual outliers are plotted with diamonds, otherwise shades of red are used to highlight outliers. The definition of outliers are from the R boxplot function.

LIKERT DATA

A scatterplot of Likert type data is problematic because there are so few possibilities for points in the scatterplot. For example, for a scatterplot of two five-point Likert response data, there are only 25 possible paired values to plot, so most of the plotted points overlap with others. In this situation, that is, when there are less than 10 values for each of the two variables, a bubble plot is

automatically provided, with the size of each point relative to the joint frequency of the paired data values. A sunflower plot can be requested in lieu of the bubble plot.

DIAGONAL

Useful particularly when comparing pre- and post- scores on some assessement, a diagonal line that runs from the lower-left corner of the graph to the upper-right corner represents the values of no change from a value on the x-axis that equals the corresponding value on the y-axis, where the pre and post scores are equal. Points on either side of that diagonal indicate + or - change. To provide this line, specify `diag=TRUE`, which will apply only to scatter plots with two numeric, non-categorical, variables. When so specified, for each data coordinate, a vertical line is drawn from the diagonal of no change to the point, unless `lines.diag` is set to `FALSE`. If `diag=TRUE`, then the axes limits are set so that each axis has the same beginning and ending point.

VARIABLE LABELS

Although standard R does not provide for variable labels, `lessR` can store the labels in the data frame with the data, obtained from the `Read` function. If this labels data frame exists, then the corresponding variable label is by default listed as the label for the corresponding axis and on the text output. For more information, see `Read`.

COLORS

Individual colors in the plot can be manipulated with options such as `col.bars` for the color of the histogram bars. A color theme for all the colors can be chosen for a specific plot with the `colors` option with the `lessR` function `set`. The default color theme is `blue`, but a gray scale is available with `"gray"`, and other themes are available as explained in `set`, such as `"red"` and `"green"`. Use the option `ghost=TRUE` for a black background, no grid lines and partial transparency of plotted colors.

Colors can also be changed for individual aspects of a scatterplot as well. To provide a warmer tone by slightly enhancing red, try `col.bg=snow`. Obtain a very light gray with `col.bg=gray99`. To darken the background gray, try `col.bg=gray97` or lower numbers. See the `lessR` function `showColors` which provides an example of all available named colors.

PDF OUTPUT

Because of the customized graphic windowing system that maintains a unique graphic window for the Help function, the standard graphic output functions such as `pdf` do not work with the `lessR` graphics functions. Instead, to obtain pdf output, use the `pdf.file` option, perhaps with the optional `pdf.width` and `pdf.height` options. These files are written to the default working directory, which can be explicitly specified with the R `setwd` function.

ADDITIONAL OPTIONS

Commonly used graphical parameters that are available to the standard R function `plot` are also generally available to `ScatterPlot`, such as:

lwd Line width, see `par`.

cex Numerical vector giving the amount by which plotting characters and symbols should be scaled relative to the default. This works as a multiple of `par`("cex"). NULL and NA are equivalent to 1.0. Note that this does not affect annotation.

cex.main, col.lab, font.sub, etc. Settings for main- and sub-title and axis annotation, see `title` and `par`.

main Title of the graph, see `title`.

xlim The limits of the plot on the x-axis, expressed as c(x1,x2), where x1 and x2 are the limits. Note that x1 > x2 is allowed and leads to a reversed axis.

ylim The limits of the plot on the y-axis.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

References

Monette, G. and Fox, J., dataEllipse function from the car package.

Gerbing, D. W. (2013). R Data Analysis without Programming, Chapter 8, NY: Routledge.

See Also

plot, stripchart, title, par, Correlation, set.

Examples

```
# scatterplot
# create simulated data, no population mean difference
# X has two values only, Y is numeric
# put into a data frame, required for formula version
n <- 12
Gender <- sample(c("Women", "Men"), size=n, replace=TRUE)
x <- round(rnorm(n=n, mean=50, sd=10), 2)
y <- round(rnorm(n=n, mean=50, sd=10), 2)
z <- round(rnorm(n=n, mean=50, sd=10), 2)
mydata <- data.frame(Gender,x,y,z)
rm(Gender); rm(x); rm(y); rm(z)

# default scatterplot, x is not sorted so type is set to "p"
# although data not attached, access each variable directly by its name
ScatterPlot(x, y)

# short name
sp(x,y)

# compare to standard R plot, which requires the mydata$ notation
plot(mydata$x, mydata$y)

# save scatterplot to a pdf file
ScatterPlot(x, y, pdf.file="MyScatterScatterPlot.pdf")

# scatterplot, with ellipse and extended axes to accommodate the ellipse
ScatterPlot(x, y, ellipse=TRUE, xlim=c(20,80), ylim=c(20,80))

# scatterplot, with loess line
ScatterPlot(x, y, fit.line="loess")

# increase span (smoothing) from default of .75
ScatterPlot(x, y, fit.line="loess", span=1.25)

# custom scatterplot, with diagonal line, connecting line segments
```

```
ScatterPlot(x, y, col.stroke="darkred", col.fill="plum", diag=TRUE)

# scatterplot with a gray scale color theme
#   or, use set(colors="gray") to invoke for all subsequent analyses
#   until reset back to default color of "blue"
set(colors="gray")
ScatterPlot(x, y)
set(colors="blue")

# by variable scatterplot with default point color, vary shapes
ScatterPlot(x,y, by=Gender)

# by variable scatterplot with custom colors, keeps only 1 shape
ScatterPlot(x,y, by=Gender, col.stroke=c("steelblue", "hotpink"))

# by variable with values of Gender for plotting symbols
# reduce the size of Gender the plotted symbols with cex<1
ScatterPlot(x, y, by=Gender, shape.pts=c("M","F"), cex=.6)

# vary both shape and color
ScatterPlot(x, y, by=Gender, col.stroke=c("steelblue", "hotpink"),
            shape.pts=c("M","F"))

# Default dot plot
ScatterPlot(y)

# Dot plot with custom colors for outliers
ScatterPlot(y, pt.reg=23, col.out15="hotpink", col.out30="darkred")

# one variable scatterplot with added jitter of points
ScatterPlot(x, method="jitter", jitter=0.05)

# by variable dot plot with custom colors, keeps only 1 shape
ScatterPlot(x, by=Gender, col.stroke=c("steelblue", "hotpink"))

# bubble plot of simulated Likert data, 1 to 7 scale
# size of each plotted point (bubble) depends on its joint frequency
# triggered by default when  < 10 unique values for each variable
x1 <- sample(1:7, size=100, replace=TRUE)
x2 <- sample(1:7, size=100, replace=TRUE)
ScatterPlot(x1,x2)

# compare to usual scatterplot of Likert data, transparency helps
plot(x1,x2)
ScatterPlot(x1,x2, kind="regular", cex=3)

# plot Likert data and get sunflower plot with loess line
ScatterPlot(x1,x2, kind="sunflower", fit.line="loess")

# scatterplot of continuous Y against categorical X, a factor
Pain <- sample(c("None", "Some", "Much", "Massive"), size=25, replace=TRUE)
Pain <- factor(Pain, levels=c("None", "Some", "Much", "Massive"), ordered=TRUE)
Cost <- round(rnorm(25,1000,100),2)
```

```
ScatterPlot(Pain, Cost)

# for this purpose, improved version of standard R stripchart
stripchart(Cost ~ Pain, vertical=TRUE)

# function curve
x <- seq(10,500,by=1)
y <- 18/sqrt(x)
# x is sorted with equal intervals so type set to "l" for line
# can use Plot or ScatterPlot, here Plot seems more appropriate
Plot(x, y)
# custom function plot
Plot(x, y, ylab="My Y", xlab="My X", col.stroke="blue",
  col.bg="snow", col.area="lightsteelblue", col.grid="lightsalmon")

# modern art
n <- sample(2:30, size=1)
x <- rnorm(n)
y <- rnorm(n)
clr <- colors()
color1 <- clr[sample(1:length(clr), size=1)]
color2 <- clr[sample(1:length(clr), size=1)]
ScatterPlot(x, y, type="l", lty="dashed", lwd=3, col.area=color1,
   col.stroke=color2, xy.ticks=FALSE, main="Modern Art",
   cex.main=2, col.main="lightsteelblue", kind="regular",
   n.cat=0)


# ------------------------------------------------
# variables in a different data frame than mydata
# ------------------------------------------------

# variables of interest are in a data frame which is not the default mydata
# although data not attached, access the variable directly by its name
data(dataEmployee)
ScatterPlot(Years, Salary, by=Gender, data=dataEmployee)
```

`set` *Set the Default Color Theme and Other System Settings*

Description

Each graph is based on a default color theme. The original default is `colors="blue"`, but `set` allows other color palettes to be set as default as well. Setting `ghost=TRUE` provides transparency effects against a black background. Other system settings are also available.

Usage

```
set(colors=c("blue", "gray", "rose", "green", "gold", "red",
         "dodgerblue", "purple", "sienna", "orange.black",
```

```
            "gray.black", "white"),

            col.fill.bar=NULL, trans.fill.bar=NULL,
            col.fill.pt=NULL, trans.fill.pt=NULL,
            col.stroke.bar=NULL, col.stroke.pt=NULL,
            col.bg=NULL, col.grid=NULL, col.heat=NULL, ghost=NULL,

            n.cat=getOption("n.cat"), quiet=getOption("quiet"),
            brief=getOption("brief"), width=120, show=FALSE)
```

Arguments

colors	The specified color scheme.
col.fill.bar	Color of a filled region, such as a histogram bar.
trans.fill.bar	The transparency of a filled bar or rectangular region, such as a histogram bar or the box in a box plot. Value from 0 to 1, opaque to transparent.
col.fill.pt	Color of a filled region, such as a plotted point.
trans.fill.pt	The transparency of the inner region of a plotted point. Value from 0 to 1, opaque to transparent.
col.stroke.bar	Color of the border of a filled region such as a histogram bar.
col.stroke.pt	Color of a line or outline of a filled region, such as the border of a plotted point.
col.bg	Color of the plot background.
col.grid	Color of the grid lines.
col.heat	Color of the heat map for correlation matrices.
ghost	If TRUE, add a black background, transparent grid lines and transparency for the bars. Overrides individual settings for those values. Setting to FALSE has no effect.
n.cat	Number of categories, which specifies the largest number of unique values of variable of a numeric data type for which the variable will be analyzed as categorical. Default is turned off, a value of 0.
quiet	If TRUE then some functions suppress console output.
brief	If set to TRUE, reduced text output. Can change system default with set function.
width	Maximum width of each line displayed at the console, just accesses the standard R options function for width.
show	Option for showing all settings.

Details

Sets the default color palette via the R options statement, as well as the transparency of plotted bars and points. Each time colors is reset, the specific color options are reset to their default values, which includes a transparency fill for plotted points of 0.66.

The gray color scheme is based on the colors used in Hadley Wickham's ggplot2 package.

Set ghost=TRUE is equivalent to setting col.bg to "black" and col.grid to "transparent" with a bar transparency of 0.7. It overrides these settings individually, so must turn ghost=FALSE and set each of the three settings individually to change from the standard ghost settings.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

References

Wickham, Hadley, ggplot2: Elegant Graphics for Data Analysis, 2nd edition, Springer, 2009.

See Also

options.

Examples

```
# set all subsequent graphic output to gray scale
set(colors="gray")

# define a custom color theme: sienna
# this works regardless of the set color theme because
#   all of the individual components of the theme are changed
# the trans options are the default values, so not necessary to
#   set unless already changed previously
# the three ghost settings - trans.fill.bar, col.grid and col.bg --
#   must be manually entered to achieve  a ghost plot
set(col.fill.bar="seagreen3")
set(col.fill.pt="seagreen3")
set(col.stroke.bar="seagreen4")
set(col.stroke.pt="seagreen4")
set(col.bg="seashell1")
set(col.grid="seashell2")
set(trans.fill.bar=.00)
set(trans.fill.pt=.66)

# all numeric variables with 8 or less unique values are analyzed
#   as categorical variables
set(n.cat=8)
```

---

showColors *Display All Named R Colors and Corresponding rgb Values*

Description

For each specified color, displays the color, the name and the associated rgb definition.

Usage

```
showColors(file="colors.pdf", color=NULL)
```

Arguments

file	Name of pdf file that contains the list of colors with a default of colors.pdf.
color	NULL for all colors, otherwise specify a color and all colors which include that color as part of their name are displayed.

Details

Every color name is defined in terms of a red, a green and a blue component. This function lists the rgb definitions for the specified colors, as well as the name and a display of each color. The output should be routed to an external pdf file for storage. The directory and file name of the output file are displayed.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

Examples

```
# all colors
#showColors()

# all colors with 'blue' in their name
#showColors(file="theblues.pdf", color="blue")
```

---

simCImean *Pedagogical Simulation for the Confidence Interval of the Mean*

Description

Show a sequence of confidence intervals, all calculated from repeated samples of simulated data from the same normal population, and show which intervals contain the true population mean.

Usage

```
simCImean(ns, n, mu=0, sigma=1, cl=0.95,
          ylim.bound=NULL, show.data=FALSE, show.title=TRUE,
          miss.only=FALSE, col.hit="gray40", col.miss="red",
          col.grid="grey90", pause=FALSE,
          main=NULL, pdf.file=NULL, pdf.width=5, pdf.height=5)
```

Arguments

ns	Number of samples, that is, repetitions of the experiment.
n	Size of each sample.
mu	Population mean.
sigma	Population standard deviation.

cl	Confidence level.
ylim.bound	Specify the maximum deviation of the mean in either direction for the extent of the vertical axis.
show.data	Plot the data for each sample as well as the confidence interval.
show.title	Place a title on the graph that contains the parameter values.
miss.only	For the text output, only display information for samples that missed the mean.
col.hit	Color of the confidence intervals that contains the mean.
col.miss	Color of the confidence intervals that miss the mean.
col.grid	Color of the grid lines.
pause	Build the graph and the text output confidence interval by confidence interval.
main	Title of graph.
pdf.file	Name of the pdf file to which graphics are redirected.
pdf.width	Width of the pdf file in inches.
pdf.height	Height of the pdf file in inches.

Details

Simulate random normal data and display the resulting confidence intervals.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

Examples

```
# 25 confidence intervals with a sample size each of 100
# mu=0, sigma=1, that is, sample from the standard normal
simCImean(25, 100)

# 25 confidence intervals with a sample size each of 100
# mu=100, sigma=15
# pause after each interval and show the data
simCImean(25, 100, mu=100, sigma=15, show.data=TRUE)
```

---

simCLT *Pedagogical Simulation for the Central Limit Theorem*

Description

Show the distribution of sample means and relevant summary statistics, such as the 95% range of variation. Provide a plot of both the specified population and the corresponding distribution of sample means.

Usage

```
simCLT(ns, n, p1=0, p2=1,
       type=c("normal", "uniform", "lognormal", "antinormal"),
       col.fill="lightsteelblue3", n.display=2, digits.d=3,
       subtitle=TRUE, pop=TRUE,
       main=NULL, pdf=FALSE, pdf.width=5, pdf.height=5, ...)
```

Arguments

ns	Number of samples, that is, repetitions of the experiment.
n	Size of each sample.
p1	First parameter value for the population distribution, the mean, minimum or meanlog for the normal, uniform and lognormal populations, respectively. Must be 0, the minimum, for the anti-normal distribution.
p2	Second parameter value for the population distribution, the standard deviation, maximum or sdlog for the normal, uniform and lognormal populations, respectively. Is the maximum for the anti-normal, usually left at the default value of 1.
type	The general population distribution.
col.fill	Fill color of the graphs.
n.display	Number of samples for which to display the sample mean and data values.
digits.d	Number of decimal digits to display on the output.
subtitle	If TRUE, then display the specific parameter values of the population or sample, depending on the graph.
pop	If TRUE, then display the graph of the population from which the data are sampled.
main	Title of graph.
pdf	Indicator as to if the graphic files should be saved as pdf files instead of directed to the standard graphics windows.
pdf.width	Width of the pdf file in inches.
pdf.height	Height of the pdf file in inches.
...	Other parameter values for R function lm which provides the core computations.

Details

Provide a plot of both the specified population and the corresponding distribution of sample means. Include descriptive statistics including the 95% range of sampling variation in raw units and standard errors for comparison to the normal distribution. Also provide a few samples of the data and corresponding means.

Four different populations are provided: normal, uniform, lognormal for a skewed distribution, and what is called the anti-normal, the combining of two side-by-side triangular distributions so that most of the values are in the extremes and fewer values are close to the middle.

For the lognormal distribution, increase the skew by increasing the value of p2, which is the population standard deviation.

The anti-normal distribution requires the `triangle` package. No population mean and standard deviation are provided for the anti-normal distribution, so the 95% range of sampling variable of the sample mean in terms of standard errors is not provided. ** Not activated until the triangle package is updated. **

If the two plots, of the population and sample distributions respectively, are written to pdf files, according to `pdf=TRUE`, they are named SimPopulation.pdf and SimSample.pdf. Their names and the directory to which they are written are provided as part the console output.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

Examples

```
# plot of the standardized normal
#  and corresponding sampling distribution with 10000 samples
#  each of size 2
simCLT(ns=1000, n=2)

# plot of the uniform dist from 0 to 4
#  and corresponding sampling distribution with 10000 samples
#  each of size 2
simCLT(ns=1000, n=2, p1=0, p2=4, type="uniform", bin.width=0.01)

# save the population and sample distributions to pdf files
simCLT(100, 10, pdf=TRUE)
```

---

simFlips *Pedagogical Binomial Simulation, Coin flips*

Description

Simulate a sequence of coin flips.

Usage

```
simFlips(n, prob=.5, show.title=TRUE,
         show.flips=TRUE, col.grid="grey90", pause=FALSE,
         main=NULL, pdf.file=NULL, pdf.width=5, pdf.height=5)
```

Arguments

`n`	Size of each sample, that is, the number of trials or flips.
`prob`	Probability of a success on any one trial.
`show.title`	Place a title on the graph that contains the parameter values.
`show.flips`	Plot the outcome of each flips.
`col.grid`	Color of the grid lines.

pause	Build the graph and the text output confidence interval by confidence interval.
main	Title of graph.
pdf.file	Name of the pdf file to which graphics are redirected.
pdf.width	Width of the pdf file in inches.
pdf.height	Height of the pdf file in inches.

Details

Generate and plot successive values of a Head or a Tail using standard R rbinom function.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

Examples

```
# 10 flips of a fair coin
simFlips(10, .5)
```

---

simMeans *Pedagogical Simulation of Sample Means over Repeated Samples*

Description

Show a sequence of sample means and data, all simulated from the same normal population. Useful for developing an intuition for developing an informal confidence interval, that is, specifying a likely range of values that contain the true population mean, but without a formal probability.

Usage

```
simMeans(ns, n, mu=0, sigma=1, ylim.bound=NULL,
         show.title=TRUE, show.data=TRUE, max.data=10,
         col.grid="grey90", pause=FALSE,
         sort=NULL, set.mu=FALSE, digits.d=2,
         main=NULL, pdf.file=NULL, pdf.width=5, pdf.height=5)
```

Arguments

ns	Number of samples, that is, repetitions of the experiment.
n	Size of each sample.
mu	Population mean.
sigma	Population standard deviation.
ylim.bound	Specify the maximum deviation of the mean in either direction for the extent of the vertical axis.
show.title	Place a title on the graph that contains the parameter values.

show.data	Show the data values on the text output.
max.data	Maximum number of data values per sample on the text output.
col.grid	Color of the grid lines.
pause	Build the graph and the text output sample by sample.
sort	Sort the output by the means in ascending order. By default is TRUE unless se.mu or pause is TRUE
set.mu	Have the program randomly set mu and sigma, usually to guess the correct value.
digits.d	Sort the output by the means in ascending order.
main	Title of graph.
pdf.file	Name of the pdf file to which graphics are redirected.
pdf.width	Width of the pdf file in inches.
pdf.height	Height of the pdf file in inches.

Details

Simulate random normal data and display the resulting sample means, both as text output and graphic output.

If pause=TRUE, then the true population values are not revealed as the simulation progresses. These values are saved in the user's workspace and can be revealed by entering their names at the user prompt, mu and sigma.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

Examples

```
# 8 samples, each with a sample size of 10
# mu=0, sigma=1, that is, sample from the standard normal
simMeans(8, 10)

# 25 sample means with a sample size each of 100
# mu=100, sigma=15
# pause after each interval and show the data
simMeans(25, 100, mu=100, sigma=15, show.data=FALSE)
```

Sort *Sort the Rows of a Data Frame*

Description

Abbreviation: srt

Sorts the values of a data frame according to the values of one or more variables contained in the data frame, or the row names. Variable types include numeric and factor variables. Factors are sorted by the ordering of their values, which, by default is alphabetical. Sorting by row names is also possible.

Usage

```
Sort(by, direction=NULL, data=mydata, quiet=getOption("quiet"), ...)

srt(...)
```

Arguments

by	One or more variables to be sorted, or just the character string row.names or random.
direction	Default is ascending for all variables listed in by. Or, specify a list of "+" for ascending and "-" for descending, one for each variable to be sorted.
data	The name of the data frame from which to create the subset, which is mydata by default.
quiet	If set to TRUE, no text output. Can change system default with set function.
...	Parameter values.

Details

Sort sorts the rows of a data frame and lists the first five rows of the sorted data frame. Specify the values upon which to base the sort with the required by parameter. If not all sorted variables are sorted in ascending order, then also specify a sequence of "+" for ascending and "-" for descending, respectively, one for each variable to be sorted. If row.names or random is specified, then no other variables can be specified.

A list of consecutive variables can be specified using the colon notation, such as Years:Salary To specify a list of multiple variables, or "+" and "-" signs, or sets of variables, separate each set of variables or each sign by a comma, then invoke the R combine or c function. For example, if three variables are to be sorted, the first two ascending and the last descending, then specify, direction=c("+","+","-").

Sort is based on the standard R function order, though the Sort function allows for the sorting of factors, whereas order does not.

Value

The sorted data frame is returned, usually assigned the name of mydata as in the examples below. This is the default name for the data frame input into the lessR data analysis functions.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

See Also

order.

Examples

```
# construct data frame
mydata <- read.table(text="Severity Description
1 Mild
4 Moderate
3 Moderate
2 Mild
1 Severe", header=TRUE)

# sort the data frame called mydata according to Severity
#   in ascending order
mydata <- Sort(Severity)

# abbreviated form, replace original with sorted
mydata <- srt(Severity)

# sort Description in descending order, sort Severity within
#  each level of Description in ascending order
mydata <- Sort(c(Description, Severity), direction=c("-", "+"))

# data in a different data frame than mydata
data(dataEmployee)
mydata <- Sort(Gender, data=dataEmployee)

# sort by row names in ascending order
mydata <- Read("Employee", format="lessR")
mydata <- Sort(row.names)

# randomly re-shuffle the rows of data
mydata <- Read("Employee", format="lessR")
mydata <- Sort(random)
```

Subset *Subset the Values of an Integer or Factor Variable*

Description

Abbreviation: subs

Based directly on the standard R subset function to only include or exclude specified rows or data, and for specified columns of data. Output provides feedback and guidance regarding the specified subset operations. Rows of data may be randomly extracted, and also with the code provided to generate a hold out validation sample created. The hold out sample is created from the original data frame, usually named mydata, so the subset data frame must be directed to a data frame with a new name or the data re-read to construct the holdout sample. Any existing variable labels are retained in the subset data frame.

Usage

```
Subset(rows, columns, data=mydata, holdout=FALSE,
       random=0, quiet=getOption("quiet"), ...)

subs(...)
```

Arguments

`rows`	Specify the rows, i.e., observations, to be included or deleted, such as with a logical expression or by direct specification of the numbers of the corresponding rows of data.
`columns`	Specify the columns, i.e., variables, to be included or deleted.
`data`	The name of the data frame from which to create the subset, which is `mydata` by default.
`holdout`	Create a hold out sample for validation if rows is a proportion or an integer to indicate random extraction of rows of data.
`random`	If an integer or proportion, specifies number of rows to data to randomly extract.
`quiet`	If set to `TRUE`, no text output. Can change system default with `set` function.
`...`	The list of variables, each of the form, `variable = equation`. Each `variable` can be the name of an existing variable in the data frame or a newly created variable.

Details

`Subset` creates a subset based on one or more rows of data and one or more variables in the input data frame, and lists the first five rows of the revised data frame. Guidance and feedback regarding the subsets are provided by default. The first five lines of the input data frame are listed before the subset operation, followed by the first five lines of the output data frame.

The argument `rows` can be a logical expression based on values of the variables, or it can be an integer or proportion to indicate random extraction of rows. An integer specifies the number of rows to retain, and a proportion specifies the corresponding proportion, which is then rounded to an integer. If `holdout=TRUE`, then the code to create a hold out data frame with a subsequent `Subset` analysis is also created. Copy and run this code on the original data frame to create the hold out sample.

To indicate retaining an observation, specify at least one variable name and the value of the variable for which to retain the corresponding observations, using two equal signs to indicate the logical equality. If no rows are specified, all rows are retained. Use the `row.names` function to identify rows by their row names, as illustrated in the examples below.

To indicate retaining a variable, specify at least one variable name. To specify multiple variables, separate adjacent variables by a comma, and enclose the list within the standard R combine function, `c`. A single variable may be replaced by a range of consecutive variables indicated by a colon, which separates the first and last variables of the range. To delete a variable or variables, put a minus sign, -, in front of the `c`.

Value

The subset of the data frame is returned, usually assigned the name of `mydata` as in the examples below. This is the default name for the data frame input into the `lessR` data analysis functions.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

See Also

`subset`, `factor`.

Examples

```
# construct data frame
mydata <- read.table(text="Severity Description
1 Mild
4 Moderate
3 Moderate
2 Mild
1 Severe", header=TRUE)

# only include those with a value of Moderate for Description
mydata <- Subset(rows=Description=="Moderate")
# use abbreviation and do not need the rows= for the first argument
mydata <- subs(Description=="Moderate")

# locate, that is, display only, the 2nd and 4th rows of data
Subset(row.names(mydata)=="2" | row.names(mydata)=="4")

# retain only the first and fourth rows of data, store in myd
myd <- Subset(c(1,4))

# delete only the first and fourth rows of data, store in myd
myd <- Subset(-c(1,4))

# only retain females and Years and Salary as variables in dataEmployee
# write result to newdata
data(dataEmployee)
newdata <- Subset(Gender=="F", columns=c(Years, Salary),
       data=dataEmployee)

# delete Years and Salary from datEmployee
mydata <- Read("Employee", format="lessR")
mydata <- Subset(columns=-c(Years, Salary))

# locate, display only, a specified row by its row.name
mydata <- Read("Employee", format="lessR", brief=TRUE)
Subset(row.names(mydata)=="Fulton, Scott")

# randomly extract 60% of the data
# generate code to create the hold out sample of the rest
```

```
mydata <- Read("Employee", format="lessR", brief=TRUE)
mysubset <- Subset(random=.6, holdout=TRUE)
```

SummaryStats *Summary Statistics for One or Two Variables*

Description

Abbreviation: ss, ss.brief

Descriptive or summary statistics for a numeric variable or a factor, one at a time or for all numeric and factor variables in the data frame. For a single variable, there is also an option for summary statistics at each level of a second, usually categorical variable or factor, with a relatively few number of levels. For a numeric variable, output includes the sample mean, standard deviation, skewness, kurtosis, minimum, 1st quartile, median, third quartile and maximum, as well as the number of non-missing and missing values. For a categorical variable, the output includes the table of counts for each value of a factor, the total sample size, and the corresponding proportions.

If the provided object to analyze is a set of multiple variables, including an entire data frame, then each non-numeric variable in the data frame is analyzed and the results written to a pdf file in the current working directory. The name of each output pdf file that contains a bar chart and its path are specified in the output.

Usage

```
SummaryStats(x=NULL, by=NULL, data=mydata, n.cat=getOption("n.cat"),
         digits.d=NULL, brief=getOption("brief"), ...)

ss.brief(..., brief=TRUE)

ss(...)
```

Arguments

x	Variable(s) to analyze. Can be a single variable, either within a data frame or as a vector in the user's workspace, or multiple variables in a data frame such as designated with the c function, or an entire data frame. If not specified, then defaults to all variables in the specified data frame, mydata by default.
by	Applies to an analysis of a numeric variable, which is then analyzed at each level of the by variable. The variable is coerced to a factor.
data	Optional data frame that contains the variable of interest, default is mydata.
n.cat	When analyzing all the variables in a data frame, specifies the largest number of unique values of variable of a numeric data type for which the variable will be analyzed as a categorical. Set to 0 to turn off.
digits.d	Specifies the number of decimal digits to display in the output.
brief	If set to TRUE, reduced text output. Can change system default with set function.
...	Further arguments to be passed to or from methods.

Details

OVERVIEW
The by option specifies a categorical variable or factor, with a relatively few number of values called levels. The variable of interest is analyzed at each level of the factor.

The `digits.d` parameter specifies the number of decimal digits in the output. It must follow the formula specification when used with the formula version. By default the number of decimal digits displayed for the analysis of a variable is one more than the largest number of decimal digits in the data for that variable.

Reported outliers are based on the boxplot criterion. The determination of an outlier is based on the length of the box, which corresponds, but may not equal exactly, the interquartile range. A value is reported as an outlier if it is more than 1.5 box lengths away from the box.

Skewness is computed with the usual adjusted Fisher-Pearson standardized moment skewness coefficient, the version found in many commercial packages.

The `lessR` function `Read` reads the data from an external csv file into the data frame called `mydata`. To describe all of the variables in a data frame, invoke `SummaryStats(mydata)`, or just `SummaryStats()`, which then defaults to the former.

In the analysis of a categorical variable, if there are more than 10 levels then an abbreviated analysis is performed, only reporting the values and the associated frequencies. If all the values are unique, then the user is prompted with a note that perhaps this is actually an ID field which should be specified using the `row.names` option when reading the data.

DATA
If the variable is in a data frame, the input data frame has the assumed name of `mydata`. If this data frame is named something different, then specify the name with the `data` option. Regardless of its name, the data frame need not be attached to reference the variable directly by its name, that is, no need to invoke the `mydata$name` notation.

To analyze each variable in the `mydata` data frame, use `SummaryStats()`. Or, for a data frame with a different name, insert the name between the parentheses. To analyze a subset of the variables in a data frame, specify the list with either a : or the `c` function, such as m01:m03 or c(m01,m02,m03).

VARIABLE LABELS
If variable labels exist, then the corresponding variable label is by default listed as the label for the horizontal axis and on the text output. For more information, see `Read`.

ONLY VARIABLES ARE REFERENCED
The referenced variable in a `lessR` function can only be a variable name. This referenced variable must exist in either the referenced data frame, `mydata` by default, or in the user's workspace, more formally called the global environment. That is, expressions cannot be directly evaluated. For example:

```
> SummaryStats(rnorm(50))   # does NOT work
```

Instead, do the following:

```
> Y <- rnorm(50)   # create vector Y in user workspace
> SummaryStats(Y)     # directly reference Y
```

Value

If the analysis is of a single numeric variable, the full analysis invisibly returns as a list: n, miss, mean, sd, skew, kurtosis, min, quartile1, median, quartile3, max, IQR. The brief analysis returns the corresponding subset of the summary statistics. If a by variable is included in the analysis, then nothing is returned.

If the analysis is of a single categorical variable, a list is invisibly returned with two tables, the frequencies and the proportions, respectively named freq and prop. If two categorical variables are analyzed, then nothing is returned.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

See Also

summary, formula, boxplot.

Examples

```
# ------------------------------------------
# one or two numeric or categorical variables
# ------------------------------------------

# create data frame, mydata, to mimic reading data with rad function
# mydata contains both numeric and non-numeric data
# X has two character values, Y is numeric
n <- 12
X <- sample(c("Group1","Group2"), size=n, replace=TRUE)
Y <- round(rnorm(n=n, mean=50, sd=10),3)
mydata <- data.frame(X,Y)
rm(X); rm(Y)

# Analyze the values of numerical Y
# Calculate n, mean, sd, skew, kurtosis, min, max, quartiles
SummaryStats(Y)
# short name
ss(Y)

# Analyze the values of categorical X
# Calculate frequencies and proportions, totals, chi-square
SummaryStats(X)

# Only a subset of available summary statistics
ss.brief(Y)
ss.brief(X)

# Reference the summary stats in the object: stats
stats <- ss(Y)
my.mean <- stats$mean
```

```
# Get the summary statistics for Y at each level of X
# Specify 2 decimal digits for each statistic displayed
SummaryStats(Y, by=X, digits.d=2)


# ----------
# data frame
# ----------

# Analyze all variables in data frame mydata at once
# Any variables with a numeric data type and 4 or less
#  unique values will be analyzed as a categorical variable
SummaryStats()

# Analyze all variables in data frame mydata at once
# Any variables with a numeric data type and 7 or less
#  unique values will be analyzed as a categorical variable
SummaryStats(n.cat=7)

# analyze just a subset of a data frame
mydata <- Read("Employee", format="lessR", quiet=TRUE)
SummaryStats(c(Salary,Years))


# ----------------------------------------
# data frame different from default mydata
# ----------------------------------------

# variables in a data frame which is not the default mydata
# access the breaks variable in the R provided warpbreaks data set
# although data not attached, access the variable directly by its name
data(warpbreaks)
SummaryStats(breaks, by=wool, data=warpbreaks)

# Analyze all variables in data frame warpbreaks at once
SummaryStats(warpbreaks)


# ----------------------------
# can enter many types of data
# ----------------------------

# generate and enter integer data
X1 <- sample(1:4, size=100, replace=TRUE)
X2 <- sample(1:4, size=100, replace=TRUE)
SummaryStats(X1)
SummaryStats(X1,X2)

# generate and enter type double data
X1 <- sample(c(1,2,3,4), size=100, replace=TRUE)
X2 <- sample(c(1,2,3,4), size=100, replace=TRUE)
SummaryStats(X1)
SummaryStats(X1, by=X2)
```

```
# generate and enter character string data
# that is, without first converting to a factor
Travel <- sample(c("Bike", "Bus", "Car", "Motorcycle"), size=25, replace=TRUE)
SummaryStats(Travel)
```

---

to *Create a Sequence of Numbered Variable Names with a Common Prefix*

Description

Generates sequentially numbered variable names, all starting with the same prefix, usually in conjunction with reading data values into R.

Usage

```
to(prefix, until, from=1, same.size=TRUE)
```

Arguments

prefix	Character string that begins each variable name.
until	Last name in the sequence, the one with the last number.
from	First name in the sequence, the one with the initial number.
same.size	If TRUE, pads the beginning of each number for the variable name with leading zeros so that all names are of the same width.

Details

Some data sets, particularly those from surveys, have sequentially numbered variable names, each beginning with the same prefix, such as the first later of the name of a set of related attitude items. This function generates the string of such variable names, generally intended for use in a read statement for reading the data and then naming the variables, or for a subsequent assignment of the names with a names. Relies upon the R paste function.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

See Also

paste.

Examples

```
# generate: "m01" "m02" "m03" "m04" "m05" "m06" "m07" "m08" "m09" "m10"
to("m", 10)

# generate: "m1"  "m2"  "m3"  "m4"  "m5"  "m6"  "m7"  "m8"  "m9"  "m10"
to("m",10, same.size=FALSE)

# generate a 10 x 10 data frame
mydata <- data.frame(matrix(rnorm(100), nrow=10))
# name the variables in the data frame
names(mydata) <- to("m", 10)
```

Transform *Transform the Values of an Integer or Factor Variable*

Description

Abbreviation: trans

Based closely on the standard R transform function. Unlike the R function, output is provided that provides feedback and guidance regarding the specified transformation(s), and variable labels are accomodated. Existing variable labels are retained in the transformed data frame. If the transformation creates a new variable, then provide a corresponding variable label with the lessR function label.

Usage

```
Transform(data=mydata, quiet=getOption("quiet"), ...)

trans(...)
```

Arguments

data	The name of the data frame from which to create the subset, which is mydata by default.
quiet	If set to TRUE, no text output. Can change system default with set function.
...	The list of transformations, each of the form, variable = equation. Each variable can be the name of an existing variable in the data frame or a newly created variable.

Details

The first five rows of the data frame are listed before the transformation, and the first five values of the transformed variables are listed after the transformation. The default input data frame is mydata.

Guidance and feedback regarding the transformations are provided by default. The first five lines of the input data frame are listed before the transformation, then the specified transformations are listed, followed by the first five lines of the transformed data frame.

Multiple transformations can be defined with a single statement. Note that a newly created transformed variable cannot then be used to define another transformed variable in the same `Transform` statement. Instead, the transformed variable that depends on an earlier created transformed variable must be defined in its own `Transform` statement.

Value

The transformed data frame is returned, usually assigned the name of `mydata` as in the examples below. This is the default name for the data frame input into the `lessR` data analysis functions.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

See Also

`transform`, `factor`.

Examples

```
# construct data frame
mydata <- read.table(text="Status Severity
1 Mild
4 Moderate
3 Moderate
2 Mild
1 Severe", header=TRUE)

# replace Status with a transformed version
mydata <- Transform(Status=Status-1)

# abbreviated form
# replace original with recoded
mydata <- trans(StatusNew=Status-1)

# replace Status with a transformed version
# leave input mydata unmodified
# save transformed data frame to the created data frame called newdata
newdata <- Transform(Status=Status-1)

# construct data frame
# recode Status into a factor
mydata <- Transform(Status=factor(Status, labels=c("OK","Hurts","Painful","Yikes")))

# read lessR data set dataEmployee into data frame mydata
mydata <- Read("Employee", format="lessR")
# multiple transformations in one statement
#  Months is a new variable
#  Salary is a new version of the old Salary
#  Satisfaction was read as non-numeric, so as a factor, but is also ordinal
#  HealthPlan was read as numeric values 0,1,2, now converted to a factor
mydata <- Transform(
```

```
  Months=Years*12,
  Salary=Salary/1000,
  Satisfaction=factor(Satisfaction, levels=c("low", "med", "high"), ordered=TRUE),
  HealthPlan=factor(HealthPlan,
     levels=c(0,1,2), labels=c("GoodHealth", "YellowCross", "BestCare"))
)
# new variable Months now exists
# if relevant, supply a corresponding variable label
mydata <- label(Months, "Months Employed in the Company")
# confirm
details.brief()
```

`ttest` *Generic Method for t-test and Standardized Mean Difference with Enhanced Graphics*

Description

Abbreviation: `tt`, `tt.brief`

Provides enhanced output from the standard `t.test` function applied to the analysis of the mean of a single variable, or the independent groups analysis of the mean difference, from either data or summary statistics. Includes the analysis of a dependent-groups analysis from the data. The data can be in the form of a data frame or separate vectors of data, one for each group. This output includes the basic descriptive statistics, analysis of assumptions and the hypothesis test and confidence interval. For two groups the output also includes the analysis for both with and without the assumption of homogeneous variances, the pooled or within-group standard deviation, and the standardized mean difference or Cohen's *d* and its confidence interval.

The output from data for two groups introduces the ODDSMD plot, which displays the Overlapping Density Distributions of the two groups as well as the means, mean difference and Standardized Mean Difference. The plot also includes the results of the descriptive and inferential analyses. For the dependent-groups analysis, a scatter plot of the two groups of data also is produced, which includes the diagonal line through the scatter plot that represents equality, and a line segment for each point in the scatter plot which is the vertical distance from the point to the diagonal line to display the amount of change.

Can also be called from the more general `model` function.

Usage

```
ttest(x=NULL, y=NULL, data=mydata, paired=FALSE,

         n=NULL, m=NULL, s=NULL, mu0=NULL,
         n1=NULL, n2=NULL, m1=NULL, m2=NULL, s1=NULL, s2=NULL,

         Ynm="Y", Xnm="X", X1nm="Group1", X2nm="Group2",

         brief=getOption("brief"), digits.d=NULL, conf.level=0.95,
         alternative=c("two.sided", "less", "greater"),
```

```
          mmd=NULL, msmd=NULL, Edesired=NULL,

          show.title=TRUE, bw1="bcv", bw2="bcv",

          graph=TRUE, line.chart=FALSE,
          pdf.file=NULL, pdf.width=5, pdf.height=5, ...)

tt.brief(..., brief=TRUE)

tt(...)
```

Arguments

x	A formula of the form Y ~ X, where Y is the numeric response variable compared across the two groups, and X is a grouping variable with two levels that define the corresponding groups, or, if the data are submitted in the form of two vectors, the responses for the first group.
y	If x is not a formula, the responses for the second group, otherwise NULL.
n	Sample size for one group.
m	Sample size for one group.
s	Sample size for one group.
mu0	Hypothesized mean for one group. If not present, then confidence interval only.
n1	Sample size for first of two groups.
n2	Sample size for second of two groups.
m1	Sample mean for first of two groups.
m2	Sample mean for second of two groups.
s1	Sample standard deviation for first of two groups.
s2	Sample standard deviation for second of two groups.
data	Data frame that contains the variable of interest, default is mydata.
paired	Set to TRUE for a dependent-samples t-test with two data vectors or variables from a data frame.
Ynm	Name of response variable.
Xnm	Name of predictor variable, the grouping variable or factor with exactly two levels.
X1nm	Value of grouping variable, the level that defines the first group.
X2nm	Value of grouping variable, the level that defines the second group.
brief	If set to TRUE, reduced text output. Can change system default with set function.
digits.d	Number of decimal places for which to display numeric values. Suggestion only.
conf.level	Confidence level of the interval, expressed as a proportion.
alternative	Default is "two.sided". Other values are "less" and "greater".
mmd	Minimum Mean Difference of practical importance, the difference of the response variable between two group means. The concept is optional, and only one of mmd and msmd is provided.

msmd	For the Standardized Mean Difference, Cohen's d, the Minimum value of practical importance. The concept is optional, and only one of mmd and msmd is provided.
Edesired	The desired margin of error for the needed sample size calculation for a 95% confidence interval, based on Kupper and Hafner (1989).
show.title	Show the title on the graph of the density functions for two groups.
bw1	Bandwidth for the computation of the densities for the first group.
bw2	Bandwidth for the computation of the densities for the second group.
graph	If TRUE, then display the graph of the overlapping density distributions.
line.chart	Plot the run chart for the response variable for each group in the analysis.
pdf.file	Name of the pdf file to which the density graph is redirected. Also specifies to save the line charts if they are present with pre-assigned names.
pdf.width	Width of the pdf file in inches.
pdf.height	Height of the pdf file in inches.
...	Further arguments to be passed to or from methods.

Details

OVERVIEW

If n or n1 are set to numeric values, then the analysis proceeds from the summary statistics, the sample size and mean and standard deviation of each group. Missing data are counted and then removed for further analysis of the non-missing data values. Otherwise the analysis proceeds from data, which can be in a data frame, by default named mydata, with a grouping variable and response variable, or in two data vectors, one for each group.

Following the format and syntax of the standard t.test function, to specify the two-group test with a formula, formula, the data must include a variable that has exactly two values, a grouping variable or factor generically referred to as X, and a numerical response variable, generically referred to as Y. The formula is of the form Y ~ X, with the names Y and X replaced by the actual variable names specific to a particular analysis. The formula method automatically retrieves the names of the variables and data values for display on the resulting output.

The values of the response variable Y can be organized into two vectors, the values of Y for each group in its corresponding vector. When submitting data in this form, the output is enhanced if the actual names of the variables referred to generically as X and Y, as well as the names of the levels of the factor X, are explicitly provided.

For the output, when computed from the data the two groups are automatically arranged so that the group with the larger mean is listed as the first group. The result is that the resulting mean difference, as well as the standardized mean difference, is always non-negative.

The inferential analysis in the full version provides both homogeneity of variance and the Welch test which does not assume homogeneity of variance. Only a two-sided test is provided. The null hypothesis is a population mean difference of 0.

If computed from the data, the bandwidth parameter controls the smoothness of the estimated density curve. To obtain a smoother curve, increase the bandwidth from the default value.

The confidence interval of the standardized mean difference is computed by the ci.smd function, written by Ken Kelley, from the MBESS package.

DATA
If the input data frame is named something different than mydata, then specify the name with the data option. Regardless of its name, the data frame need not be attached to reference the variable directly by its name without having to invoke the mydata$name notation.

PRACTICAL IMPORTANCE
The practical importance of the size of the mean difference is addressed when one of two parameter values are supplied, the minimum mean difference of practical importance, mmd, or the corresponding standardized version, msmd. The remaining value is calculated and both values are added to the graph and the console output.

DECIMAL DIGITS
The number of decimal digits is determined by default from the largest number of decimal digits of the entered descriptive statistics. The number of decimal digits is then set at that value, plus one more with a minimum of two decimal digits by default. Or, override the default with the digits.d parameter.

VARIABLE LABELS
If variable labels exist, then the corresponding variable label is by default listed as the label for the horizontal axis and on the text output. For more information, see Read.

PDF OUTPUT
Because of the customized graphic windowing system that maintains a unique graphic window for the Help function, the standard graphic output functions such as pdf do not work with the lessR graphics functions. Instead, to obtain pdf output, use the pdf.file option, perhaps with the optional pdf.width and pdf.height options. These files are written to the default working directory, which can be explicitly specified with the R setwd function.

Value

Returned value is NULL except for a two-group analysis from a formula. Then the values for the response variable of the two groups are separated and returned invisibly as a list for further analysis as indicated in the examples below. The first group of data values is the group with the largest sample mean.

value1	Value of the grouping variable for the first group.
group1	Data values for the first group.
value2	Value of the grouping variable for the second group.
group2	Data values for the second group.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

References

Ken Kelley and Keke Lai (2012). MBESS: MBESS. R package version 3.3.3. http://CRAN.R-project.org/package=MBESS Kupper and Hafner (1989). The American Statistician, 43(2):101-105.

See Also

t.test, density, plot.density, ttestPower, formula.

Examples

```
# ------------------------------------------------------------
# tt for two groups, from a formula
# ------------------------------------------------------------

# create simulated data, no population mean difference
# X has two values only, Y is numeric
# put into a data frame, required for formula version
n <- 24
X <- sample(c("Group1","Group2"), size=n, replace=TRUE)
Y <- round(rnorm(n=n, mean=50, sd=10),2)
mydata <- data.frame(X,Y)
rm(X); rm(Y)

# analyze data with formula version
# variable names and levels of X are automatically obtained from data
# although data frame not attached, reference variable names directly
ttest(Y ~ X)

# short form
tt(Y ~ X)

# brief version of results
tt.brief(Y ~ X)

# return the vectors group1 and group2 into the object t.out
# separate the data values for the two groups and analyze separately
t.out <- ttest(Y ~ X)
Histogram(group1, data=t.out)
Histogram(group2, data=t.out)

# compare to standard R function t.test
t.test(mydata$Y ~ mydata$X, var.equal=TRUE)

# consider the practical importance of the difference
ttest(Y ~ X, msmd=.5)

# obtain the line chart of the response variable for each group
ttest(Y ~ X, line.chart=TRUE)

# variable of interest is in a data frame which is not the default mydata
# access the data frame in the lessR dat.twogroup data set
# although data not attached, access the variables directly by their name
data(dataLearn)
ttest(Score ~ StudyType, data=dataLearn)


# ------------------------------------------------------------
```

```
# tt for a single group, from data
# ------------------------------------------------------------

# confidence interval only, from data
ttest(Y)

# confidence interval and hypothesis test, from data
ttest(Y, mu0=52)


# ------------------------------------------------------
# tt for two groups from data stored in two vectors
# ------------------------------------------------------

# create two separate vectors of response variable Y
# the vectors exist are not in a data frame
#   their lengths need not be equal
Y1 <- round(rnorm(n=10, mean=50, sd=10),2)
Y2 <- round(rnorm(n=10, mean=60, sd=10),2)

# analyze the two vectors directly
# usually explicitly specify variable names and levels of X
#   to enhance the readability of the output
ttest(Y1, Y2, Ynm="MyY", Xnm="MyX", X1nm="Group1", X2nm="Group2")

# dependent groups t-test from vectors in global environment
ttest(Y1, Y2, paired=TRUE)

# dependent groups t-test from variables in data frame mydata
mydata <- data.frame(Y1,Y2)
rm(Y1);  rm(Y2)
ttest(Y1, Y2, paired=TRUE)
# independent groups t-test from variables (vectors) in a data frame
ttest(Y1, Y2)


# ------------------------------------------------------
# tt from summary statistics
# ------------------------------------------------------

# one group: sample size, mean and sd
# optional variable name added
tt(n=34, m=8.92, s=1.67, Ynm="Time")

# confidence interval and hypothesis test, from descriptive stats
tt(n=34, m=8.92, s=1.67, mu0=9, conf.level=0.90)

# two groups: sample size, mean and sd for each group
# specify the briefer form of the output
tt.brief(n1=19, m1=9.57, s1=1.45, n2=15, m2=8.09, s2=1.59)
```

`ttestPower` *Compute a Power Curve for a One or Two Group t-test*

Description

Abbreviation: `ttp`

From one or two sample sizes, and either the within-cell (pooled) standard deviation, or one or two separate group standard deviations, generate and calibrate a power curve for either the one-sample t-test or the independent-groups t-test, as well as ancillary statistics. Uses the standard R function `power.t.test` to calculate power and then the `ScatterPlot` function in this package to automatically display the annotated power curve with colors.

For both the one and two-group t-tests, power is calculated from a single sample size and single standard deviation. For the two-sample test, the within-group standard deviation is automatically calculated from the two separate group standard deviations if not provided directly. Similarly, the harmonic mean of two separate sample sizes is calculated if two separate sample sizes are provided.

Usage

```
ttestPower(n=NULL, s=NULL, n1=NULL, n2=NULL, s1=NULL, s2=NULL,
         mmd=NULL, msmd=NULL, mdp=.8, mu0=NULL,
         pdf.file=NULL, pdf.width=5, pdf.height=5, ...)

ttp(...)
```

Arguments

`n`	Sample size for each of the two groups.
`s`	Within-group, or pooled, standard deviation.
`n1`	Sample size for Group 1.
`n2`	Sample size for Group 2.
`s1`	Sample standard deviation for Group 1.
`s2`	Sample standard deviation for Group 2.
`mmd`	Minimum Mean Difference of practical importance, the difference of the response variable between two group means. The concept is optional, and only one of `mmd` and `msmd` is provided.
`msmd`	For the Standardized Mean Difference, Cohen's `d`, the Minimum value of practical importance. The concept is optional, and only one of `mmd` and `msmd` is provided.
`mdp`	Minimum Desired Power, the smallest value of power considered to provide sufficient power. Default is 0.8. If changed to 0 then the concept is dropped from the analysis.
`mu0`	Hypothesized mean, of which a provided value triggers a one-sample analysis.
`pdf.file`	Name of the pdf file to which graphics are redirected.

pdf.width	Width of the pdf file in inches.
pdf.height	Height of the pdf file in inches.
...	Other parameter values, such as lwd and cex.lab from plot and col.line and col.bg from ScatterPlot.

Details

This function relies upon the standard power.t.test function to calibrate and then calculate the power curve according to the relevant non-central t-distribution. The Plot function from this package, which in turn relies upon the standard plot function, plots the power curve. As such, parameters in Plot for controlling the different colors and other aspects of the display are also available, as are many of the more basic parameters in the usual plot function.

Also plotted, if provided, is the minimal meaningful difference, mmd, as well as the minimal desired power, mdp, provided by default. Relevant calculations regarding these values are also displayed at the console. One or both concepts can be deleted from the analysis. Not providing a value mmd implies that the concept will not be considered, and similarly for setting mdp to 0.

Invoke the function with the either the within-group (pooled) standard deviation, s, or the two separate group standard deviations, s1 and s2, from which s is computed. If the separate standard deviations are provided, then also provide the sample sizes, either as a single value of n or as two separate sample sizes, n1 and n2. If separate sample sizes n1 and n2 are entered, their harmonic mean serves as the value of n.

For power analysis of the two-sample t-test, the null hypothesis is a zero population mean difference. For a one-sample test, the null hypothesis is specified, and it is this non-null specification of mu0 that triggers the one-sample analysis. Only non-directional or two-tailed tests are analyzed.

The effect size that achieves a power of 0.8 is displayed. If a minimal meaningful difference, mmd, is provided, then the associated power is also displayed, as well as the needed sample size to achieve a power of 0.8.

If the function is called with no parameter values, that is, as ttp(), then the values of n1, n2 and sw must already exist before the function call. If they do, these values are used in the power computations.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

See Also

Plot, plot, power.t.test.

Examples

```
# default power curve and colors
ttestPower(n=20, s=5)
# short name
ttp(n=20, s=5)

# default power curve and colors
# plus optional smallest meaningful effect to enhance the analysis
```

```
ttestPower(n=20, s=5, mmd=2)

# power curve from both group standard deviations and sample sizes
# also provide the minimum standardized mean difference of
#   practical importance to obtain corresponding power
ttestPower(n1=14, n2=27, s1=4, s2=6, msmd=.5)

# power curve for one sample t-test, triggered by non-null mu0
ttestPower(n=20, s=5, mu0=30, mmd=2)
```

---

values *List the Values of a Variable*

Description

List the values of a variable from the global environment or a data frame.

Usage

```
values(x, data=mydata, ...)
```

Arguments

x	Variable for which to construct the histogram and density plots.
data	Data frame that contains the variable of interest, default is mydata.
...	Other parameter values for as defined processed by print, including digits.

Details

Provided for listing the values of a variable in an unattached data frame. All lessR functions that access data for analysis from a data frame, such as the default mydata provided by the Read function that reads the data frame from an external data file, do not require the data frame to be attached. Attaching a data frame can lead to some confusing issues, but one negative of not attaching is that simply listing the name of a variable within the data frame leads to an 'object not found' error. The values function provides access to that variable within a data frame just as is true for any other lessR function that accesses data.

The function displays the values of the specified variable with the standard R print function, so parameter values for print can also be passed to values.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

See Also

print

Examples

```
# generate 10 random normal data values
Y <- rnorm(10)
mydata <- data.frame(Y)
rm(Y)

# list the values of Y
values(Y)

# variable of interest is in a data frame which is not the default mydata
# access the breaks variable in the R provided warpbreaks data set
# although data not attached, access the variable directly by its name
data(warpbreaks)
values(breaks, data=warpbreaks)
```

`Write` *Write the Contents of a Data Frame to an External File*

Description

Abbreviation: `wrt`, `wrt.r`

Writes the contents of the specified data frame, such as with the default `mydata`, to the current working directory as either the default `csv` data file or an R native data file of the specified data frame. If the write is of a `csv` file, then any variable labels are written to a second `csv` file with "_lbl" appended to the file name.

Usage

```
Write(ref, format=c("csv", "R"), data=mydata, ...)

wrt(...)

wrt.r(..., format="R")
```

Arguments

`ref`	Name of the output file as a character string, that is, with quotes. If not included in the name, the file type is automatically added to the name, either `.csv` or `.rda`, depending of the value of `format`.
`format`	Format of file to be written with `.csv` as the default.
`data`	Data frame to be written as an object, that is, no quotes.
`...`	Other parameter values consistent with the usual `write.table`, such as `row.names=FALSE` to suppress writing the row names.

Details

Can specify the file name without the file type, which `Write` adds automatically, `.csv` for a comma separated values data file and `.rda` for a native R data file. The default file name is the name of the data frame to be written. The name of the file that is written, as well as the name of the working directory into which the file was written, are displayed at the console.

`Write` is designed to work in conjunction with the function `Read` from this package, which reads a csv, fixed width format, or native SPSS or R data files into the data frame `mydata`. `Write` relies upon the R functions `write.csv` and `save`.

Author(s)

David W. Gerbing (Portland State University; <gerbing@pdx.edu>)

See Also

`Read`, `write.csv`, `save`.

Examples

```
# create data frame called mydata
n <- 12
X <- sample(c("Group1","Group2"), size=n, replace=TRUE)
Y <- rnorm(n=n, mean=50, sd=10)
mydata <- data.frame(X,Y)

# write the current contents of default data frame mydata to GoodData.csv
Write("GoodData")
# short name
# write the default data frame mydata to the R data file mydata.rda
wrt.r()

# access the R data frame warpbreaks
data(warpbreaks)
# write the file warpbreaks.rda
wrt.r(data=warpbreaks)
```

Index