Package ‘lmSupport’

April 8, 2018

Type Package
Title Support for Linear Models
Version 2.9.13
Date 2018-04-08
Author John Curtin <jjcurtin@wisc.edu>
Maintainer John Curtin <jjcurtin@wisc.edu>
Description Provides tools and a consistent interface to support analyses using General, Generalized, and Multi-level Linear Models.
License GPL (>= 2)
Depends
Imports AICcmodavg, car, gplots, graphics, grDevices, gvlma, lme4, pbkrtest, psych, pwr, stats, utils
LazyLoad yes
LazyData true
URL http://dionysus.psych.wisc.edu/
NeedsCompilation no
Repository CRAN
Date/Publication 2018-04-08 13:40:34 UTC

R topics documented:

BAC ......................................................... 2
dfMerge ................................................... 3
dfReadDat ............................................... 4
dfRemoveCases ........................................... 5
dfRownames .............................................. 6
dfWriteDat ............................................... 7
figAxis .................................................... 8
figBarPlot ............................................... 9
figConfidenceBand ..................................... 10
The BAC data frame has 96 rows and 4 columns. The observations are the fear-potentiated startle scores by blood alcohol concentration, trait anxiety and sex.
dfMerge

Usage

BAC

Format

This data frame contains the following columns:

- **BAC**: Blood alcohol concentration.
- **TA**: Trait anxiety
- **Sex**: Participant sex
- **FPS**: Fear-potentiated startle

Source

Loosely based on real data collected by Curtin et al from psychophysiological studies of alcohol effects on FPS.

---

**dfMerge**

*Merges two data frames*

Description

Merges variables from two data frames (DataX, DataY) by default or merges cases (if AddVars=FALSE). When merging variables, by default matches on row names but can use other variable names in DataX (ByX) and DataY (ByY) as needed. Also by default, includes all cases in DataX and DataY but can limit to only matching (AllX=FALSE, AllY=FALSE) or left join (AllY=FALSE) or right join (AllX=FALSE).

When merging cases, will add variables to DataX or DataY as needed and set added variables to NA.

Usage

\[
\text{dfMerge(DataX, DataY, ByX = 0, ByY = 0, AllX = TRUE, AllY = TRUE, AddVars=TRUE)}
\]

Arguments

- **DataX**: first data frame for merge
- **DataY**: second data frame for merge
- **ByX**: Name of variable in DataX to match cases on. Column can be specified by name or number. Default is 0 which uses rownames
- **ByY**: Name of variable in DataY to match cases on. Column can be specified by name or number. Default is 0 which uses rownames
dfReadDat

Description

Opens a tab-delimited data file with standard Curtin lab format which include using a header and setting delimiter to tab and as.is=TRUE.

If variable named SubID (default) or other text supplied by SubID variable exists in dat file, row names will be set with this variable and then variable is removed from new data frame.

Usage

dfReadDat(File, SubID = "SubID", SubIDDigits = NULL)
**dfRemoveCases**

**Arguments**

- **File**
  - File name for .dat file including extension
- **SubID**
  - String to indicate name of SubID variable. Default is 'SubID'. If set to NULL, rownames will not be altered
- **SubIDDigits**
  - Length of SubID rowname string. If NULL, will be set to max length in data

**Value**

- returns a data frame

**Author(s)**

John J. Curtin <jjcurtin@wisc.edu>

**See Also**

- read.table(), read.delim(), write.table(), dfRownames()

**Examples**

```r
#dfReadDat('Sample1.dat') #not executable unless Sample1.dat exists in path
#dfReadDat('Sample2.dat, SubID = 'subnum') #not executable unless Sample2.dat exists in path
```

---

**Description**

Removes cases from dataframe. Cases can be numeric or character. If numeric, rownames must be able to be converted to numeric. Returns warning if cases not found in dataframe.

**Usage**

```r
dfRemoveCases(Data, Cases)
```

**Arguments**

- **Data**
  - a dataframe
- **Cases**
  - a vector of numeric or character case IDs/rownames

**Value**

- Returns dataframe with cases removed.

**Author(s)**

John J. Curtin <jjcurtin@wisc.edu>
Examples

d = dfRemoveCases(BAC, c('0125', '0111'))

Description

Sets the row names of the data frame to the variable name listed as SubID. SubID should be text name of variable. Also keeps number of characters constant by default (numeric SubID only) and removes SubID by default.

Usage

dfRownames(Data, SubID = "SubID", FixedWidth = TRUE, Remove = TRUE, MaxNumDigits=NULL)

Arguments

Data a data frame with a variable containing subject ID numbers
SubID Text name of subject ID variable. Default is SubID
FixedWidth logical. If TRUE (default), all row names will be the same length by padding with leading 0's. Only applies to numeric SubIDs
Remove logical. If TRUE (default), the subject ID variable will be removed from data frame after setting rownames
MaxNumDigits Length of rowname string. If NULL, will be set to max length in data. Only applies to numeric SubIDs

Value

Returns data frame with rownames set (and SubID removed if requested)

Author(s)

John J. Curtin <jjcurtin@wisc.edu>

Examples

d <- data.frame(SubID = c(1,2,3,10,20), v1=c(1,2,3,4,5), v2=c(1,NA,NA,2,4), data=1:5)
d=dfRownames(d)
**dfWriteDat**

Saves dataframe as tab-delimited text file with typical Curtin lab parameters

---

### Description

Saves a dataframe as a tab-delimited data file with standard Curtin lab format. Will add rownames as a first column in .dat file and label this column with SubID.

### Usage

```
dfWriteDat(Data, File, SubID = "SubID")
```

### Arguments

- **Data**: a dataframe
- **File**: file name for .dat file
- **SubID**: Name for new column with data from rownames. If NULL, rownames will not be added to .dat file) Default is 'SubID'

### Details

Uses these parameters with write.table no append, quote, separator is tab, no rownames, yes for columns.

### Value

No return value but creates .dat file in current wd.

### Author(s)

John J. Curtin <jjcurtin@wisc.edu>

### See Also

read.table(), read.delim(), write.table()

### Examples

```r
# Not run
#data(BAC)
#dfWriteDat(BAC, File="Test1.dat")
#dfWriteDat(BAC, File="Test2.dat", SubID = 'ID')
#dfWriteDat(BAC, File="Test3.dat", SubID = NULL)
```
figAxis  

Wrapper for standardized use of axis()

Description

Wrapper function for standardized use of axis() with lab defaults for display

Usage

```r
figAxis(side, lab.text, scale.at=NULL, scale.text=NULL,
  scale.lwd=NULL, scale.cex=NULL, scale.font=NULL,
  lab.line= NULL, lab.cex=NULL, lab.font=NULL)
```

Arguments

- `side`: an integer specifying which side of the plot the axis is to be drawn on. The axis is placed as follows: 1=below, 2=left, 3=above and 4=right.
- `lab.text`: name label for the axis
- `scale.at`: the points at which tick-marks are to be drawn. Non-finite (infinite, NaN or NA) values are omitted. By default (when NULL) tickmark locations are computed, see 'Details' below.
- `scale.text`: this can either be a logical value specifying whether (numerical) annotations are to be made at the tickmarks, or a character or expression vector of labels to be placed at the tickpoints.
- `scale.lwd, scale.font, scale.cex`: lwd, font, and cex for scale annotations. Accessed from options if NULL
- `lab.line, lab.cex, lab.font`: line number, cex, and font for axis label. Accessed from options if NULL

Value

None

Author(s)

John J. Curtin <jjcurtin@wisc.edu>

See Also

axis(), figLabDefaults(), figSetDefaults(), figNewDevice(), figLines(), figLines()
Examples

```r
X = rep(2:9,4)+jitter(rep(0,32))
Y = X + rnorm(length(X),0,5)
m = lm(Y ~ X)
dNew = data.frame(X=seq(2,9,by=.01))
p = modelPredictions(m,dNew)
figNewDevice()
figPlotRegion(x=c(0,10),y=c(0,10))
figConfidenceBand(p$pX,p$predicted,p$CILo,p$CIHi)
figPoints(X,Y)
figLines(p$pX,p$predicted)
figAxis(side=1,lab.text='X-axis 1', scale.at=seq(from=0,to=10,by=2))
figAxis(side=2,lab.text='Startle Response', scale.at=seq(from=0,to=10,by=2))
```

---

**figBarPlot**  
*Wrapper for standarized use of barplot2() from gplots*

---

**Description**

Wrapper function for standardized use of barplot2() with lab defaults for display

**Usage**

```r
figBarPlot(Means, ylim=NULL, lab.text=NULL, main.text=NULL, se=NULL,
bars.col=NULL, bars.density=NULL, bars.angle=NULL, bars.space=NULL,
scale.cex=NULL, lab.cex=NULL, lab.font=NULL,
ci.plot=NULL, ci.col=NULL, ci.lty=NULL, ci.lwd = NULL, ci.width = NULL)
```

**Arguments**

- `Means`  
  matrix of means to plot.
- `ylim`  
  vector of min and max for y axis
- `lab.text`  
  label for x-axis
- `main.text`  
  main label for plot. See barplot2
- `se`  
  standard error of mean for CI plotting, if needed
- `bars.col`, `bars.density`, `bars.angle`, `bars.space`  
  color, density, angle, and space for bars. see barplot2 for additional detail
- `scale.cex`  
  cex for x axis scale
- `lab.font`, `lab.cex`  
  cex and font for x axis label
- `ci.plot`  
  boolean to indicate if CIs should be plotted
- `ci.col`, `ci.lty`, `ci.lwd`, `ci.width`  
  col, lty, lwd, and width of CI lines
figConfidenceBand

Value
None

Author(s)
John J. Curtin <jjcurtin@wisc.edu>

See Also
barplot2(), figLabDefaults(), figSetDefaults(), figNewDevice(), figLines(), figLines()

Examples

```r
## not run
##Means = matrix(c(70,65,68,91,100,90), nrow=2,ncol=3, byrow=TRUE)
##colnames(Means) = c('ITI', 'CUE-', 'CUE+')
##rownames(Means) = c('Non-deprived', 'Deprived')
##se = matrix(c(5,10,4,5,10,4), nrow=2,ncol=3, byrow=TRUE)
##bars.col = c('gray', 'white', 'black')
##bars.density = c(-1,-1,10)  # negative density suppresses lines
##bars.angle = c(0,0,45)

##figNewDevice()
##figBarPlot(Means,ylim=c(0,130), lab.text='Group', ci.plot=TRUE, se=se,
##   bars.col=bars.col,bars.density=bars.density,
##   bars.angle = bars.angle)
##figAxis(side=2,lab.text='Startle Response', scale.at=seq(0,120,by=20))
##figLegend('topright', legend=colnames(Means),fill=bars.col, angle=bars.angle,
##   density=bars.density)
```

---

figConfidenceBand  Creates confidence band for regression line

Description

Adds a confidence band around a regression line in a plot

Usage

```r
figConfidenceBand(X, Y, CILo, CIHi, Color)
```
Arguments

x  Vector of data for X to plot
y  Vector of data for Y to plot
CILo  Vector of data for lower bound of confidence interval
CIHi  Vector of data for upper bound of confidence interval
Color  String to indicate R color. Will be .15 transparent in plot

Value

No value is returned

Author(s)

John J. Curtin <jjcurtin@wisc.edu>

---

figErrBars  Wrapper for standarized use of error bars

Description

Wrapper function for standardized use of error bars with segments() with lab defaults for display

Usage

figErrBars(x, y, yplus, yminus, errbars.cap = NULL, 
errbars.lwd = NULL, errbars.col = NULL)

Arguments

x, y, yplus, yminus  coordinate vectors of x and y points for error bars
errbars.cap  Width of caps on error bars. Accessed from options if NULL
errbars.lwd  Line width. Accessed from options if NULL
errbars.col  Line color. Accessed from options if NULL

Value

None

Author(s)

John J. Curtin <jjcurtin@wisc.edu>

See Also

lines(), figLabDefaults(), figSetDefaults(), figNewDevice(), figLines(), figPoints()
Examples

```r
figNewDevice()
figPlotRegion(x=c(0,5), y=c(0,10))
figLines(c(0,10), c(0,10))
figAxis(side=1, lab.text='X-axis 1', scale.at=seq(from=0, to=10, by=2))
figAxis(side=2, lab.text='Startle Response', scale.at=seq(from=0, to=10, by=2))
```

---

**figLabDefaults**

*Generate list graphing parameters*

**Description**

Generates a list of detailed default graphing parameters that can be used by fig functions in lmSupport for standardized graphing. Need to use figSetDefaults with this list to save in options.

**Usage**

```r
figLabDefaults()
```

**Value**

Returns a list that includes all graphing parameters

**Author(s)**

John J. Curtin <jjcurtin@wisc.edu>

---

**figLayout**

*Wrapper for standardized use of layout()*

**Description**

Wrapper function for standardized use of layout() and layout.show()

**Usage**

```r
figLayout(nRows, nCols, heights=rep(1, nRows), widths=rep(1, nCols),
          layout.display=TRUE)
```

**Arguments**

- `nRows, nCols` integers specifying number of rows and columns in matrix
- `heights` vector indicating relative heights of rows; Default is equal heights
- `widths` vector indicating relative width of columns; Default is equal widths
- `layout.display` Boolean if outlines and numbers of panels should be displayed
figLegend

Value
None

Author(s)
John J. Curtin <jjcurtin@wisc.edu>

See Also
layout(), layout.show(), figLabDefaults(), figSetDefaults(), figNewDevice(), figLines(), figLines

Examples

```r
X = rep(2:9,4)+jitter(rep(0,32))
Y = X + rnorm(length(X),0,5)
m = lm(Y ~ X)
dNew = data.frame(X=seq(2,9,by=.01))
p = modelPredictions(m,dNew)

figNewDevice()
figLayout(2,1)
figPlotRegion(x=c(0,10),y=c(0,10))
figConfidenceBand(p$X,p$Predicted,p$CIlo,p$CIHi)
figLines(p$X,p$Predicted)
figAxis(side=1,lab.text='X-axis 1', scale.at=seq(from=0,to=10,by=2))
figAxis(side=2,lab.text='Startle Response', scale.at=seq(from=0,to=10,by=2))

figPlotRegion(x=c(0,10),y=c(0,10))
figPoints(X,Y)
figAxis(side=1,lab.text='X-axis 1', scale.at=seq(from=0,to=10,by=2))
figAxis(side=2,lab.text='Startle Response', scale.at=seq(from=0,to=10,by=2))
```

---

### Description

Wrapper function for standardized use of Legend() with lab defaults for display

### Usage

```r
figLegend(x, y=NULL, legend, fill=NULL, border='black',
angle=NULL, density=NULL, pch=NULL, leg.cex=NULL, leg.lty,
leg.lwd=NULL, leg.font=NULL, leg.bty=NULL)
```
Arguments

x, y  the x and y co-ordinates to be used to position the legend. They can be specified by keyword or in any way which is accepted by xy.coords: See 'Details' in legend()

legend  a character or expression vector of length = 1 to appear in the legend. Other objects will be coerced by as.graphicsAnnot.

fill  if specified, this argument will cause boxes filled with the specified colors (or shaded in the specified colors) to appear beside the legend text.

border  border of box surrounding legend points. see fill

angle  angle of shading lines.

density  the density of shading lines, if numeric and positive. If NULL or negative or NA color filling is assumed.

pch  the plotting symbols appearing in the legend, as numeric vector or a vector of 1-character strings (see points). Unlike points, this can all be specified as a single multi-character string. Must be specified for symbol drawing.

leg.cex,leg.lty,leg.lwd,leg.font,leg.bty  cex, lty, lwd, font, and bty for legend. Defaults to values in options if NULL. Set leg.lty & leg.lwd to NA if you want bars rather than lines in legend

Value

None

Author(s)

John J. Curtin <jjcurtin@wisc.edu>

See Also

legend(), figLabDefaults(), figSetDefaults(), figNewDevice(), figLines(),figLines()

Examples

```r
##not run
##Means = matrix(c(70,65,68,91,100,90), nrow=2,ncol=3, byrow=TRUE)
##colnames(Means) = c('ITI', 'CUE-', 'CUE')
##rownames(Means) = c('Non-deprived', 'Deprived')
##se = matrix(c(5,10,4,5,10,4), nrow=2,ncol=3, byrow=TRUE)

##bars.col = c('gray', 'white', 'black')
##bars.density = c(-1,-1,10) #negative density suppresses lines
##bars.angle = c(0,0,45)

##figNewDevice()
##figBarPlot(Means,ylim=c(0,130), lab.text='Group', ci.plot=TRUE, se=se, bars.col=bars.col,bars.density=bars.density, bars.angle = bars.angle)
##figAxis(side=2,lab.text='Startle Response', scale.at=seq(0,120,by=20))
##figLegend(x='topright', legend=colnames(Means),fill=bars.col,
## angle=bars.angle, density=bars.density)
```
figLines

Wrapper for standardized use of lines()

Description

Wrapper function for standardized use of lines() with lab defaults for display

Usage

figLines(x, y, lines.lwd=NULL, lines.lty=NULL, lines.col=NULL, lines.pch=NULL)

Arguments

x, y coordinate vectors of points to join
lines.lwd Line width. Accessed from options if NULL
lines.lty Line type. Accessed from options if NULL
lines.col Line color. Accessed from options if NULL
lines.pch point type. Default is no points. See points() for other types. Accessed from options if NULL

Value

None

Author(s)

John J. Curtin <jjcurtin@wisc.edu>

See Also

lines(), figLabDefaults(), figSetDefaults(), figNewDevice(), figLines(),figPoints()

Examples

figNewDevice()
figPlotRegion(x=c(0,5), y=c(0,10))
figLines(c(0,10),c(0,10))
figAxis(side=1,lab.text='X-axis 1', scale.at=seq(from=0,to=10,by=2))
figAxis(side=2,lab.text='Startle Response', scale.at=seq(from=0,to=10,by=2))
figNewDevice

Opens device for graphing

Description

Open a device for graphing (window, pdf, tiff) and establishes default parameters for standardized graphs.

Usage

```R
figNewDevice(Width=7, Height=7, Type='window', File, Res=300)
```

Arguments

- `Width, Height`: the (nominal) width and height of the canvas of the plotting window in inches. Default = 7.
- `Type`: Device type: Window, pdf, tiff. Default = 'Window'. Windows will open a window using either `windows()`, `quartz()`, or `X11()` depending on the OS. tiff and pdf will graph to that type of file.
- `File`: File name as string. Used by tiff and pdf.
- `Res`: The nominal resolution in ppi used by tiff. Default = 300

Value

None

Author(s)

John J. Curtin <jjcurtin@wisc.edu>

See Also

`figLabDefaults()`, `figSetDefaults()`, `windows()`, `tiff()`, `pdf()`

Examples

```R
figNewDevice(Type='tiff', File='Test.tiff', Res=72)
figNewDevice(Type='Windows')
```
figPlotRegion  

Sets up a plot region for later plotting

Description

Sets up a plot region for later plotting with fig functions. Typically use is to establish the x and y ranges for region and otherwise leave blank for later drawing with fig functions.

Usage

```r
figPlotRegion(x, y, xlab = NA, ylab = NA, axes=FALSE, type='n')
```

Arguments

- `x,y`: min and max for x and y plot region
- `xlab, ylab`: Labels for x and y axes. Typically left blank (NA)
- `axes`: a logical value indicating whether both axes should be drawn on the plot. Typically not included (FALSE)
- `type`: 1-character string giving the type of plot desired. Typically no data are plotted ('n'). see type in plot() for more info

Value

None

Author(s)

John J. Curtin <jjcurtin@wisc.edu>

See Also

plot(), figLabDefaults(), figSetDefaults(), figNewDevice(), figLines(), figPoints()

Examples

```r
figNewDevice()
figPlotRegion(x=c(0,5), y=c(0,10))
figLines(c(0,10),c(0,10))
figAxis(side=1,lab.text='X-axis 1', scale.at=seq(from=0,to=10,by=2))
figAxis(side=2,lab.text='Startle Response', scale.at=seq(from=0,to=10,by=2))
```
figPoints

Wrapper for standardized use of points()

Description

Wrapper function for standardized use of points() with lab defaults for display

Usage

figPoints(x, y, type='p', points.lwd=NULL, points.pch=NULL, points.col=NULL, points.bg=NULL, points.cex=NULL)

Arguments

- x, y: coordinate vectors of points to join
- type: character indicating the type of plotting; actually any of the types as in plot.default. Default = 'p'
- points.lwd: Line width for points. Accessed from options if NULL
- points.pch: plotting 'character', i.e., symbol to use. This can either be a single character or an integer code for one of a set of graphics symbols. The full set of S symbols is available with pch = 0:18, see the examples below. (NB: R uses circles instead of the octagons used in S.). Accessed from options if NULL
- points.col, points.bg, points.cex: Point color, bg, and cex. Accessed from options if NULL

Value

None

Author(s)

John J. Curtin <jjcurtin@wisc.edu>

See Also

points(), figLabDefaults(), figSetDefaults(), figNewDevice(), figLines(), figLines()

Examples

X = rep(2:9,4)+jitter(rep(0,32))
Y = X + rnorm(length(X),0,5)
m = lm(Y ~ X)
dNew = data.frame(X=seq(2,9,by=.01))
p = modelPredictions(m,dNew)
figNewDevice()  #default is for windows(), can use quartz, tiff, or pdf as Type
figPlotRegion(x=c(0,10),y=c(0,10))
figConfidenceBand(p$pX,p$Predicted,p$CILo,p$CIHi)
figSetDefaults

figPoints(X,Y)
figLines(p$X,p$Predicted)
figAxis(side=1,lab.text='X-axis 1', scale.at=seq(from=0,to=10,by=2))
figAxis(side=2,lab.text='Startle Response', scale.at=seq(from=0,to=10,by=2))

---

**figSetDefaults**  
*Saves list of graphing parameters in options*

**Description**

Saves a list of graphing parameters, typically created by figLabDefaults) in options for later use in graphing by fig functions.

**Usage**

`figSetDefaults(FigPars)`

**Arguments**

- **FigPars**  
A list of graphing parameters

**Value**

None

**Author(s)**

John J. Curtin <jjcurtin@wisc.edu>

**See Also**

- `figLabDefaults()`, `options()`

**Examples**

```r
FigPars = figLabDefaults()
FigPars$plot.lwd = 2
figSetDefaults(FigPars)
```
figStripChart  

*Create strip chart on plot*

**Description**

Adds a strip chart (variant of a rug plot that includes density info) to X (or other) axis on a plot.

**Usage**

```r
figStripChart(x, side=1, sshift=0.3, adjoffset=1, strip.col='gray',
strip.pch=15, strip.cex = 0.2)
```

**Arguments**

- `x` vector of data to plot
- `side` axis for plot, 1=bottom (default), 2=left, 3= top, 4= right
- `sshift` scaling parameter for location of plot. Use default
- `adjoffset` scaling parameter for dot spacing
- `strip.col` color of dots. Default is gray
- `strip.pch` point type for dots. Default is 15 (small dot)
- `strip.cex` scaling parameter for size of dots

**Value**

No value is returned

**Author(s)**

John J. Curtin <jjcurtin@wisc.edu>

---

figText  

*Wrapper for standardized use of text()*

**Description**

Wrapper function for standardized use of text() with lab defaults for display

**Usage**

```r
figText(x, y, label, text.font = NULL, text.cex = NULL, text.adj = NULL, text.col=NULL)
```
Arguments

- \(x, y\) coordinates to plot text
- \(label\) label/text to plot
- \(text.font\) Text font. Accessed from options if NULL
- \(text.cex\) Text cex. Accessed from options if NULL
- \(text.adj\) Text adj. Accessed from options if NULL
- \(text.col\) Text color. Accessed from options if NULL

Value

None

Author(s)

John J. Curtin <jjcurtin@wisc.edu>

See Also

- \lines()\n- \figLabDefaults()\n- \figSetDefaults()\n- \figNewDevice()\n- \figLines()\n- \figPoints()\n
Examples

```r
figNewDevice()
figPlotRegion(x=c(0,5), y=c(0,10))
figLines(c(0,10),c(0,10))
figAxis(side=1, lab.text='X-axis 1', scale.at=seq(from=0,to=10,by=2))
figAxis(side=2, lab.text='Startle Response', scale.at=seq(from=0,to=10,by=2))
figText(0,9, 'Figure label')
```

Description

These functions are provided for compatibility with older versions of the `lmSupport` package and may be removed eventually. These functions may not necessarily work as in previous versions of the `lmSupport` package. It is strongly recommended that you update your code to use the new functions.
Usage

```
lm.boxCox(...)
lm.codeRegressors(...)
lm.correctSE(...)
lm.deltaR2(...)
lm.describeData(...)
lm.describeGroups(...)
lm.figSum(...)
lm.mergeData(...)
lm.pointEstimates(...)
lm.readDat(...)
lm.removeCases(...)
lm.renameVar(...)
lm.setContrasts(...)
lm.setRowNames(...)
lm.stripChart(...)
lm.sumSquares(...)
lm.writeDat(...)
```

Arguments

... pass arguments down.

Details

`lm.boxCox` is now a synonym for the `modelBoxCox` function. `lm.codeRegressors` is now a synonym for the `varRegressors` function. `lm.correctSE` is now a synonym for the `modelCorrectSE` function. `lm.deltaR2` is now a synonym for the `modelCompare` function. `lm.describeData` is now a synonym for the `varDescribe` function. `lm.describeGroups` is now a synonym for the `varDescribeBy` function. `lm.figSum` is now a synonym for the `varPlot` function. `lm.mergeData` is now a synonym for the `dfMerge` function. `lm.pointEstimates` is now a synonym for the `modelPredictions` function. `lm.readDat` is now a synonym for the `dfReadDat` function. `lm.removeCases` is now a synonym for the `dfRemoveCases` function. `lm.renameVar` is now a synonym for the `varRename` function. `lm.setContrasts` is now a synonym for the `varContrasts` function. `lm.setRowNames` is now a synonym for the `dfRowNames` function. `lm.sumSquares` is now a synonym for the `modelEffectSizes` function. `lm.stripChart` is now a synonym for the `figStripChart` function. `lm.writeDat` is now a synonym for the `dfWriteDat` function.

---

**modelAssumptions**  
Assess Linear Model Assumptions

**Description**

Provides diagnostic graphs and score tests to evaluate linear model assumptions of normality, constant variance and linearity. Follows best practices and uses many functions from car package.
Usage

modelAssumptions(Model, Type = "NORMAL", ID=row.names(Model$model), one.page = TRUE)

Arguments

Model a linear model produced by lm.
Type Type =c("NORMAL", "CONSTANT", "LINEAR") for normally distributed residuals with constant variance, and linear (e.g., mean of residuals 0 for all Y)
ID Use to identify points. Default = row.names(model$model). NULL = no identification
one.page logical; display all graphs on one page if TRUE (Default).

Author(s)

John J. Curtin <jjcurtin@wisc.edu>

References


Examples

data(BAC)
m = lm(FPS~BAC+TA, data=BAC)
modelAssumptions(m,"NORMAL")
modelAssumptions(m,"CONSTANT")
modelAssumptions(m,"LINEAR", ID=NULL)

modelBoxCox

Calculates lambda for Box-Cox power transformation

Description

Calculates and plots log-likelihoods lambda for power transformation of response variable. Reports chi-square test of lambda <> 1. All values of Y must > 0 or function will crash. Add offset to Y if necessary (see example). Default lambda range is -2 to 2. Uses boxCox() from car package.

Usage

modelBoxCox(Model, Lambdas = seq(-2, 2, by = 0.1))

Arguments

Model an unweighted linear model, produced by lm.
Lambdas a vector of lambda values to plot. Default is seq(-2,2,by=0.1)
Author(s)
John J. Curtin <jjcurtin@wisc.edu>

References

See Also
boxCox(), boxcox()

Examples
```r
# Not run
# m = lm(FPS + 99 ~ BAC+TA, data=BAC)
# modelBoxCox(m)
```

```r
modelCaseAnalysis
Provides graphs and/or tests for problematic cases for a linear model

Description
Provides diagnostic graphs and visual cut points for identification of points that are univariate outliers, high leverage, regression outliers, and/or influential

Usage
modelCaseAnalysis(Model, Type = "RESIDUALS", Term = NULL, ID = row.names(Model$model))

Arguments
- **Model**: a linear model produced by `lm`
- **Type**: Type = c('RESIDUALS', 'UNIVARIATE', 'HATVALUES', 'COOKSD', 'DFBETAS', 'INFLUENCEPLOT', 'COVRATIO') RESIDUALS (default) = regression outliers, UNIVARIATE = univariate outliers, HATVALUES = leverage, COOKSD = model influence, DFBETAS= individual parameter influence, INFLUENCEPLOT= leverage X influence, COVRATIO = inflation of SEs.
- **Term**: Term from model to display. Used only by DFBETAS. DEFAULT is NULL with all terms displayed
- **ID**: Use to identify points. Default = row.names(Model$model). NULL = no identification

Value
Side effect of plot is main goal for function. Also returns a list with Rownames and CaseAnalysis Values for cases identified. No list returned if DFBETAS without single term identified.
**modelCompare**

**Description**
Calculates F-test to compare two models to determine if ModelA significantly reduces SSE from ModelC. Also reports Partial eta2 and Delta R2 for this model comparison. ModelC should contain subset of ModelA regressors.

**Usage**
modelCompare(ModelC, ModelA)

**Arguments**
- **ModelC**: a linear model, produced by lm. This compact model should include a subset of regressors from ModelA
- **ModelA**: a linear model, produced by lm. This augmented model should include all regressors from ModelC plus additional regressors.

**Details**
Calculates F test for model comparison \( F = \frac{(\text{sseC} - \text{sseA})/ (\text{pA} - \text{pC})}{(\text{sseA} / (N-\text{pA}))} \) \( \text{ndf} = \text{pA} - \text{pC} \) \( \text{ddf} = N - P \)

**Value**
Returns a list with results for model comparison, sses, and other relevant fields
Author(s)
John J. Curtin <jjcurtin@wisc.edu>

Examples
###NOT RUN
###mC = lm(FPS~BAC, data=BAC)
###mA = lm(FPS~BAC+TA, data=BAC)
###modelCompare(mC, mA)

---

**modelCorrectSE**

Calculates White (1980)’s heteroscedascity-corrected SEs and Tests
for a linear model

---

**Description**

Calculates heteroscedascity-corrected SEs and associated tests for regression coefficients based on method described by White (1980) using hccm() from car package. Prints tables with orginal and corrected results and returns corrected coefficient table

**Usage**

```r
modelCorrectSE(Model, Digits=3)
```

**Arguments**

- **Model**: an unweighted linear model, produced by `lm`
- **Digits**: digits to print in table output. Default =3

**Value**

Returns the `lm` coefficients table with corrected SEs and associated tests

**Author(s)**
John J. Curtin <jjcurtin@wisc.edu>

**References**

modelEffectSizes

See Also

hccm() in car package

Examples

```r
## NOT RUN
##m = lm(FPS~BAC+TA, data=BAC)
##modelCorrectSE(m)
```

```
summary(modelEffectSizes(m))
```

modelEffectSizes
\[ \text{Calculates effect size indices based on Sums of Squares} \]

Description

Calculates unique SSRs, SSE, SST. Based on these SSs, it calculates partial eta2 and delta R2 for all effects in a linear model object. For categorical variables coded as factors, it calculates these for multi-df effect. Manually code regressors to get 1 df effects. Uses car::Anova() with Type 3 error

Usage

```r
modelEffectSizes(Model, Print = TRUE, Digits = 4)
```

Arguments

- `Model` a linear model, produced by `lm`
- `Print` Display results to screen. Default = `TRUE`
- `Digits` Number of digits for printing effect sizes

Value

Returns a list with fields for effect sizes, SSE, and SST.

Author(s)

John J. Curtin <jjcurtin@wisc.edu>

See Also

Anova()

Examples

```r
## NOT RUN
##m = lm(FPS~BAC+TA, data=BAC)
##modelEffectSizes(m)
```
modelErrors

Description
Simple wrapper to return model errors using residuals() function. Implemented simply to match terminology to 610/710 GLM course. Also prints (but does not return) model SSE.

Usage
modelErrors(Model)

Arguments
Model an lm model object

Value
Returns vector of model errors (residuals) from sample

Author(s)
John J. Curtin <jjcurtin@wisc.edu>

See Also
residuals, lm

Examples

```r
## NOT RUN
## data(BAC)
## m = lm(FPS~BAC+TA, data=BAC)
## modelErrors(m)
```

modelPower

Description
Wrapper to calculate power for tests of parameter estimates or full model in GLM based on Cohen’s tables and using pwr.f2.test in pwr package. Allows use of partial eta squared or delta R2 rather than just f2 as effect size. If you provide power, it returns N, if you provide N, it returns power. You must specify effect size as either f2, partial eta2, or delta R2 with model R2. You must also specify the number of parameters in the compact (pc) and augmented (pa) for the model comparison that will test the effect.
**modelPower**

### Usage

```r
df = modelPower(pc=NULL, pa=NULL, N=NULL, alpha=0.05, power=NULL,
                 f2=NULL, peta2=NULL, dR2=NULL, R2=NULL)
```

### Arguments

- **pc**: Number of parameters in the compact model; i.e., intercept + all parameters excluding the effect of interest; This is the numerator df of the F test for the effect.
- **pa**: Number of parameters in the augmented model; i.e., the intercept and all parameters including the effect of interest.
- **N**: sample size.
- **alpha**: alpha for statistical test.
- **power**: power for statistical test.
- **f2**: $f^2$ effect size.
- **peta2**: partial eta2 effect size.
- **dR2**: delta R2 effect size; if provided must also specify R2.
- **R2**: Model R2, only need if using Delta R2 as effect size.

### Value

Returns either power or N from analysis.

### Author(s)

John J. Curtin <jjcurtin@wisc.edu>

### See Also

- `pwr.f2.test`

### Examples

```r
modelPower(pc=3, pa=4, power=.90, peta2=.157)
modelPower(pc=1, pa=3, N=100, peta2=.157, alpha=.01)
```
modelPredictions  

*Provides predicted values for sample or new data. New predictions include SEs*

**Description**

If no data are provided, modelPredictions returns a numeric vector predicted values for the sample, functioning as a simple wrapper for fitted.values(). If a dataframe with new values for Xs are provided, modelPredictions adds predicted values and SEs for these new data to the dataframe using predict() from car package.

**Usage**

```r
modelPredictions(Model, Data=NULL, Label = NULL, Type = 'response')
```

**Arguments**

- **Model**: a linear model, produced by `lm`.
- **Data**: a dataframe containing cases for predictions. Must include all regressors from model. Default is NULL with predictions returned for the current sample.
- **Label**: A string label to append to variable names for predicted values, CIs and SE. Default is NULL with no append.
- **Type**: 'response' or 'link'. Used only for glm objects. see predict()

**Value**

If Data=NULL, returns a numeric vector of predicted values for sample. If Data are provided, adds four new columns at the front of the dataframe. These variables are named Predicted (predicted value), CIlo (lower bound of -1 SE from Predicted), CIHi (upper bound of +1 SE), and SE (Standard error of predicted value). NOTE: For GLM, +1 SE are calculated on the link scale and then converted to the response scale (which will be asymmetric) if Type = response. If Label is not NULL, than Label is appended to end of these four variable names.

**Author(s)**

John J. Curtin <jjcurtin@wisc.edu>

**See Also**

predict(), fitted.values()
Examples

```r
# NOT RUN
# make plot of predicted values with 1SE error bands for CAN
#m = lm(interlocks~assets+nation, data=Ornstein)
dNew = data.frame(assets = seq(1000,100000, by=1000),nation='CAN')
dNew = modelPredictions(m, dNew)
plot(dNew$assets,dNew$Predicted, type = 'l', col= 'red')
lines(dNew$assets,dNew$CIlo, type = 'l', col= 'gray', lwd = .5)
lines(dNew$assets,dNew$CIhi, type = 'l', col= 'gray', lwd = .5)

# Return predicted values for sample
#P = modelPredictions(m)
```

modelR2

---

### Model R2, adjusted R2 and F-test

**Description**

Reports model R2, adjusted R2, and F-test of model R2.

**Usage**

```r
modelR2(Model, Print=TRUE)
```

**Arguments**

- `Model` an lm model object
- `Print` print results to screen. Default is TRUE

**Value**

Returns full list object from modelSummary() with many stats

**Author(s)**

John J. Curtin <jjcurtin@wisc.edu>

**See Also**

lm, modelSummary, summary

**Examples**

```r
# NOT RUN
#m = lm(FPS~BAC+TA, data=BAC)
#modelR2(m)
```
modelRmd

Returns a formatted string for stats reporting from a model in R Markdown

Description

Returns a formatted string to report the B, CI, partial-eta2, t, and p-value for an effect from an lm model. This formatted string is appropriate for use in an R Markdown document for a dynamic report of research results.

Usage

modelRmd(effect, mod, B=1, CI=8, statistic='t', pe=2)

Arguments

effect  Text label for effect in model
mod     object returned from lm() or Anova()
B       number of decimal places for report of B if lm model; NULL if B should not be reported. Ignored for Anova model
CI      number of decimal places for report of Bs in 95 CI; Null if CI should not be reported. Ignored for Anova model
statistic test statistic to report: 't' or 'F'. Not currently implemented. t for lm and F for Anova
pe      number of decimal places for report of partial eta2. Null if should not be reported

Value

Returns a formatted string that can be directly included in a R Markdown file for a dynamic report

Author(s)

John J. Curtin <jjcurtin@wisc.edu>

modelSummary

summary of results for lm model

Description

This is a modified version of summary for use with an lm, glm, or lmer object. It provides results that align better with Brauer/Curtin perspective on these linear models from their graduate statistics series
varContrasts

Usage
modelSummary(Model, t = TRUE, Print= TRUE, Digits = 4)

Arguments
Model a linear model, produced by lm.
t Indicates if t-statistics (TRUE; Default) or F-statistics should be reported for tests of parameter estimates
Print Print output to screen. Default is TRUE
Digits Number of digits for values in coefficients table. Default = 4

Details
Reports model summary results from an lm object. Results include parameter estimates and their tests, SSE, model R2

Value
Returns a list with results for model.

Author(s)
John J. Curtin <jjcurtin@wisc.edu>

See Also
summary, modelR2

Examples
###NOT RUN
###m = lm(FPS~BAC+TA, data=BAC)
###modelSummary(m)

---

varContrasts Set Factor Contrasts

Description
Calculates contrast matrix for a specified contrast type. Options include DUMMY, POC, HELMERT, EFFECTS

Usage
varContrasts(TheFactor, Type = "DUMMY", RefLevel = length(levels(TheFactor)),
POCList = NULL, Labels = NULL)
Arguments

**TheFactor**
- factor from dataframe

**Type**
- type of contrast, Options include DUMMY (default), POC, HELMERT, or EFFECTS

**RefLevel**
- Reference level for contrast. Only applies to DUMMY, HELMERT, and EFFECTS. For DUMMY: RefLevel is numeric index of control/reference category (i.e. coded 0 for all regressors). For HELMERT: RefLevel = 1 indicates reverse HELMERT (i.e., last vs. earlier, second to last vs. earlier, etc). For EFFECTS: RefLevel is numeric index of excluded level.

**POCList**
- if Type = POC, a list of Contrasts is required in POCList; e.g., list(c(1,0,-1), c(-1,2,-1)). Best to provide as whole numbers. Function will re-scale to unit weighted contrasts.

**Labels**
- if Type = POC, Labels can be provided. If NULL (Default), contrast labels are POC1, POC2, etc.

Details

Use the contrast matrix with contrasts() to set contrast for a specific factor in dataframe.

Value

Returns contrast matrix for indicated type of contrast.

Author(s)

John J. Curtin <jjcurtin@wisc.edu>

See Also

contrasts

Examples

d = data.frame(f=factor(c('f1', 'f2', 'f3'))) contrasts(d$f)

##set as DUMMY with last category as reference
contrasts(d$f) = varContrasts(d$f, Type='DUMMY', RefLevel = 3)

##set as POC with user defined labels
contrasts(d$f) = varContrasts(d$f, Type='POC', POCList = list(c(2,-1,-1),c(0,1,-1)),
  Labels = c('f1_v_f2f3', 'f2_v_f3'))

##set as reverse HELMERT
contrasts(d$f) = varContrasts(d$f, Type='HELMERT', RefLevel = 1)

##set as EFFECTS, excluding f3 vs. grand mean contrast
contrasts(d$f) = varContrasts(d$f, Type='EFFECTS', RefLevel = 3)
**varDescribe**

*Provides typical descriptive statistics for data frame*

**Description**

Provides three levels of detail regarding descriptive statistics for a data frame. Based on `describe()` function from psych package.

**Usage**

`varDescribe(data, Detail = 2, Digits=2)`

**Arguments**

- `data`: a data frame
- `Detail`: Indicates level of detail for descriptives, 1=minimal, 2=typical (default), 3= detailed
- `Digits`: Number of decimal places to display; NULL = display all sig digits. Default =2.

**Value**

Returns table with descriptive statistics rounded to digits.

**Author(s)**

John J. Curtin `<jjcurtin@wisc.edu>`

**See Also**

- `describe`
- `lm.describeGroups`
- `describe.by`

**Examples**

```r
## NOT RUN
##varDescribe(BAC)
##varDescribe(BAC, Detail=3)
##varDescribe(BAC, Detail=2, Digits=1)
```
varDescribeBy  
*Provides common descriptives for dataframe by factor(s)*

**Description**

Provides commons descriptive statistics for a data frame split on some factor or combination of factors. Essentially a wrapper for varDescribe() and by().

**Usage**

```r
varDescribeBy(Data, IVList)
```

**Arguments**

- `Data`: a dataframe
- `IVList`: list of one or more factors from data frame

**Value**

An object of class "by", giving the results from varDescribe() applied to each subset.

**Author(s)**

John J. Curtin <jjcurtin@wisc.edu>

**Examples**

```r
# NOT RUN
# varDescribeBy(Adler, list(Adler$expectation, Adler$instruction))
```

---

varMarkdown  
*Returns a formatted string for stats reporting in R Markdown*

**Description**

Returns a formatted string to report the B, CI, partial-eta2, t, and p-value for an effect from an lm mode. This formatted string is appropriaite for use in an R Markdown document for a dynamic report of research results.

**Usage**

```r
varMarkdown(effect, mod, modsum, statistic='t', B=1, CI=B, pe=2)
```
varOdd

Arguments

- `effect`: Text label for effect from `lm`
- `mod`: object returned from `lm()`
- `modsum`: object returned from `summary()` or `modelSummary()`
- `statistic`: test statistic to report: 't' or 'F'
- `B`: number of decimal places for report of B; NULL if B should not be reported
- `CI`: number of decimal places for report of Bs in 95 CI; NULL if CI should not be reported
- `pe`: number of decimal places for report of partial eta2. Null if should not be reported

Value

Returns a formatted string that can be directly included in a R Markdown file for a dynamic report

Author(s)

John J. Curtin <jjcurtin@wisc.edu>

Description

Returns result of test if Numbers are Odd.

Usage

```r
varOdd(Numbers)
```

Arguments

- `Numbers`: Vector of numbers to test

Value

Returns vector of booleans to indicate result of test

Author(s)

John J. Curtin <jjcurtin@wisc.edu>

Examples

```r
varOdd(3)
varOdd(c(1,2,3,4,5))
```
varPadString  

Pads a string to fixed length

Description

Pads a string to fixed length (StringLen) with leading character (PadChar). If string length > StringLen, issues warning but returns original string.

Usage

varPadString(X, StringLen, PadChar = '0')

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>String to pad</td>
</tr>
<tr>
<td>StringLen</td>
<td>Fixed length of output strings</td>
</tr>
<tr>
<td>PadChar</td>
<td>Character to use for padding</td>
</tr>
</tbody>
</table>

Value

Returns string(s) with padding

Author(s)

John J. Curtin <jjcurtin@wisc.edu>

Examples

varPadString(c('1', '2', '300'),3,'0')

varParse  

Returns a subset of digits from a Number

Description

Returns a subset of digits from a Number.

Usage

varParse(Number, UpperDigit=1, LowerDigit=1)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Number to parse</td>
</tr>
<tr>
<td>UpperDigit</td>
<td>Location in base ten of upper end of digits to return</td>
</tr>
<tr>
<td>LowerDigit</td>
<td>Location in base ten of lower end of digits to return</td>
</tr>
</tbody>
</table>
**Value**

Returns a subset of the digits in Number

**Author(s)**

John J. Curtin <jjcurtin@wisc.edu>

**Examples**

```r
varParse(1234,100,10)
varParse(1234,1,1)
varParse(1234,1000,1000)
```

---

**Description**

Represents important aspects of a variable/vector both visually (histogram, rug or strip, and density plots) and with descriptive statistics of varying detail

**Usage**

```r
varPlot(TheVar, VarName = '', IDs = NULL, AddPoints = 'Strip',
        AddDensity = TRUE, Detail = 2)
```

**Arguments**

- **TheVar**: A variable/vector to visualize
- **VarName**: The variable name of TheVar as string. Default = ''
- **IDs**: Rownames for interactive identification of data points, Default is NULL with no identification done
- **AddPoints**: Strip (default), Rug, or None
- **AddDensity**: TRUE (default) or FALSE to include density plot
- **Detail**: 1-3 of increasing detail for descriptives using varDescribe()

**Value**

Prints descriptive statistics table and creates graphic as side effect. Returns list with Indices, Row-names, and Values if identify is not NULL

**Author(s)**

John J. Curtin <jjcurtin@wisc.edu>
See Also
hist(), rug(), varStripPlot(), density(), varDescribe(), describe(), identify()

Examples

```r
## NOT RUN
##data(BAC)
##varPlot(BAC$FPS, 'FPS')  # default use strip
##varPlot(BAC$FPS, AddPoints='RUG')
##varPlot(BAC$FPS, IDs=rownames(BAC))
```

---

**varRecode**

Recode levels of variable

Description

Recodes levels of variable from old values to new values. Levels in Old are recoded to levels in New by matching position in these two vectors.

Usage

`varRecode(Var, Old, New)`

Arguments

- `Var`: A variable to recode.
- `Old`: Vector with original levels of Var
- `New`: Vector with new levels

Value

Returns variable with new levels

Author(s)

John J. Curtin <jjcurtin@wisc.edu>

See Also

recode

Examples

```r
##d$rIV1 = varRecode(d$rIV1, c(-1,1), c(-.5, .5))
##d$rIV2 = varRecode(d$rIV2, c(1,2,3), c(-.667, .333, .333))
##d$rIV3 = varRecode(d$rIV3, c('A', 'B'), c('C', 'D'))
```
varRegressors

Description

 Adds actual numeric regressors for factor to dataframe as new variables

Usage

 varRegressors(Data, VarName, RegressorNames = NULL)

Arguments

 Data The dataframe to add regressors
 VarName Character string name of variable to code regressor for
 RegressorNames Optional variable names for regressors.

Value

 Returns original data frame (Data) with addition of new regressors.

Author(s)

 John J. Curtin <jjcurtin@wisc.edu>

Examples

 # NOT RUN
 # data(BAC)
 # BACSsex = factor(BAC$Sex)
 # BAC = varRegressors(BAC,'Sex')

varRename

 Rename Variable in Dataframe

Description

 Renames a variable in specified dataframe.

Usage

 varRename(Data, From, To)
Arguments

Arguments

Data a dataframe object
From vector of original name(s) of variable(s) as strings
To vector of new name(s) of variable(s) as strings

Value

Returns dataframe with new variable names for specified variable(s)

Author(s)

John J. Curtin <jjcurtin@wisc.edu>

Examples

```r
d <- data.frame(x=1:10, y=11:20)
names(d)
d <- varrename(d, c('x', 'y'), c('x1', 'y1'))
names(d)
```

Description

Reverse score an ordinal or boolean scored item/variable

Usage

```r
varReverse(Var, LowAnchor, HighAnchor)
```

Arguments

Var A variable to reverse score.
LowAnchor Absolute low value for variable
HighAnchor Absolute high value for variable

Value

Returns variable new (reversed) scores

Author(s)

John J. Curtin <jjcurtin@wisc.edu>

See Also

recode
varScore

Examples

```r
# d$item5r = varReverse(d$item5, 1, 5)
```

```
| varScore | Creates a total score from a sum of items |
```

Description

Creates a total score from a sum of items in a data frame. Can do range checking for items, reverse scoring of items, and prorating for missing data.

Usage

```
varScore(Data, Forward, Reverse=NULL, Range = NULL, Prorate = TRUE, MaxMiss = .20)
```

Arguments

- `Data`: A dataframe that contains item scores among other variables.
- `Forward`: A vector of variable names to indicate the items that should be summed as is (in contrast to reverse scored). All items should be listed in EITHER Forward or Reverse argument.
- `Reverse`: A vector of variable names to indicate the items that should be summed after reverse scoring the items. Range argument (see below) must also be specified to reverse score items. Default is NULL which indicates no items are reverse scored. All items should be listed in EITHER Forward or Reverse argument.
- `Range`: A numeric vector with two values for low and high anchor values for items. Must be specified if any items will be reverse scored. Used also to do range checking for all items. Default is NULL which indicates no range checking and no reverse scored items.
- `Prorate`: A boolean to indicate if total score should be prorated for missing data. Default is TRUE.
- `MaxMiss`: Maximum acceptable percentage of missing data before total score will be set to missing. Implemented regardless if Prorate is TRUE or FALSE. However, if Prorate is false, should probably be set to 0.

Details

This is a flexible routine to score measures that consist of sums of items.

Value

Returns vector of total scores for each participant

Author(s)

John J. Curtin <jjcurtin@wisc.edu>
Examples

## not run

```r
# varScore(d, c('I1', 'I3', 'I4'), Reverse= c('I2', 'I5'),
##       Range = c(1,5), Prorate=TRUE, MaxMiss = .25)
```
Index

*Topic **datasets**
  BAC, 2
*Topic **descriptives**
  varPlot, 39
*Topic **graphic**
  figAxis, 8
  figBarPlot, 9
  figConfidenceBand, 10
  figErrBars, 11
  figLabDefaults, 12
  figLayout, 12
  figLegend, 13
  figLines, 15
  figNewDevice, 16
  figPlotRegion, 17
  figPoints, 18
  figSetDefaults, 19
  figStripChart, 20
  figText, 20
*Topic **manip**
  dfMerge, 3
  dfReadDat, 4
  dfRemoveCases, 5
  dfRownames, 6
  dfWriteDat, 7
  modelErrors, 28
  modelPower, 28
  modelR2, 31
  varContrasts, 33
  varDescribe, 35
  varOdd, 37
  varPadString, 38
  varParse, 38
  varRecode, 40
  varRename, 41
  varReverse, 42
  varScore, 43
*Topic **regression**
  modelAssumptions, 22
  modelBoxCox, 23
  modelCaseAnalysis, 24
  modelCompare, 25
  modelCorrectSE, 26
  modelEffectSizes, 27
  modelPredictions, 30
  modelSummary, 32
  varContrasts, 33
  varRegressors, 41
*Topic **summary**
  varDescribeBy, 36

BAC, 2

contrasts, 34
dfMerge, 3, 22
dfReadDat, 4, 22
dfRemoveCases, 5, 22
dfRownames, 6, 22
dfWriteDat, 7, 22
figAxis, 8
figBarPlot, 9
figConfidenceBand, 10
figErrBars, 11
figLabDefaults, 12
figLayout, 12
figLegend, 13
figLines, 15
figNewDevice, 16
figPlotRegion, 17
figPoints, 18
figSetDefaults, 19
figStripChart, 20, 22
figText, 20
lm.boxCox (lmSupport-deprecated), 21
lm.codeRegressors (lmSupport-deprecated), 21
lm.correctSE (lmSupport-deprecated), 21
lm.deltaR2 (lmSupport-deprecated), 21
lm.describeData (lmSupport-deprecated), 21
lm.describeGroups
   (lmSupport-deprecated), 21
lm.figSum (lmSupport-deprecated), 21
lm.mergeData (lmSupport-deprecated), 21
lm.pointEstimates
   (lmSupport-deprecated), 21
lm.readDat (lmSupport-deprecated), 21
lm.removeCases (lmSupport-deprecated), 21
lm.renameVar (lmSupport-deprecated), 21
lm.setContrasts (lmSupport-deprecated), 21
lm.setRownames (lmSupport-deprecated), 21
lm.stripChart (lmSupport-deprecated), 21
lm.sumSquares (lmSupport-deprecated), 21
lm.writeDat (lmSupport-deprecated), 21
lmSupport-deprecated, 21

modelAssumptions, 22
modelBoxCox, 22, 23
modelCaseAnalysis, 24
modelCompare, 22, 25
modelCorrectSE, 22, 26
modelEffectSizes, 22, 27
modelErrors, 28
modelPower, 28
modelPredictions, 22, 30
modelR2, 31
modelRmd, 32
modelSummary, 32

varContrasts, 22, 33
varDescribe, 22, 35
varDescribeBy, 22, 36
varMarkdown, 36
varOdd, 37
varPadString, 38
varParse, 38
varPlot, 22, 39
varRecode, 40
varRegressors, 22, 41
varRename, 22, 41
varReverse, 42
varScore, 43