

Package ‘cg’

March 11, 2012

Type Package

Title Comparison of groups

Version 0.9-4

Date 2012-03-11

Author Bill Pikounis and John Oleynick <cg@billpikounis.net>

Maintainer Bill Pikounis and John Oleynick <cg@billpikounis.net>

Description cg is comprehensive data analysis software, and stands for “compare groups.” Its genesis and evolution are driven by common needs to compare administrations, conditions, etc. in medicine research & development. The current version provides comparisons of unpaired samples, i.e. a linear model with one factor of at least two levels. Good data graphs, modern statistical methods, and useful displays of results are emphasized.

License GPL (>= 2)

LazyLoad yes

LazyData yes

Depends R (>= 2.14.0), Hmisc, base, utils, methods, stats, graphics,grid, lattice, MASS, survival, multcomp, nlme

Imports VGAM (>= 0.8-6)

Repository CRAN

Date/Publication 2012-03-11 06:26:49

R topics documented:

cg-package	3
boxplot.cgOneFactorData	5
canine	7
canine.listfmt	8
cgInternalClasses	10
cgInternalUtilities	10
cgLineColors	11
cgValidity	12
comparisons	13
comparisonsGraph	17
comparisonsgraph	18
comparisonsGraph.cgOneFactorComparisonsTable	21
comparisonsTable	23
comparisonsTable.cgOneFactorFit	26
descriptiveTable	30
descriptiveTable.cgOneFactorData	31
downweightedTable	34
downweightedTable.cgOneFactorFit	35
errorBarGraph	37
errorbargraph	39
errorBarGraph.cgOneFactorFit	41
fit	44
fit.cgOneFactorData	45
globalTest	48
globalTest.cgOneFactorFit	49
gmcsfcens	51
gmcsfcens.listfmt	53
grpSummaryTable	55
grpSummaryTable.cgOneFactorFit	57
kmGraph	60
kmGraph.cgOneFactorData	62
pointGraph	64
pointGraph.cgOneFactorData	65
prepare	67
prepareCGOneFactorData	68
print.cgOneFactorComparisonsTable	73
print.cgOneFactorDescriptiveTable	75
print.cgOneFactorDownweightedTable	76
print.cgOneFactorFit	78
print.cgOneFactorGlobalTest	79
print.cgOneFactorGrpSummaryTable	81
print.cgOneFactorSampleSizeTable	82
qqGraph	84
qqGraph.cgOneFactorFit	85
samplesizeGraph	87
samplesizeGraph.cgOneFactorSampleSizeTable	89

samplesizeTable	91
samplesizeTable.cgOneFactorFit	93
show.cgOneFactorComparisonsTable	96
show.cgOneFactorDescriptiveTable	97
show.cgOneFactorDownweightedTable	99
show.cgOneFactorGlobalTest	100
show.cgOneFactorGrpSummaryTable	101
show.cgOneFactorSampleSizeTable	102
showObj	103
showObj.cgOneFactorFit	104
summary.cgOneFactorFit	105
varianceGraph	107
varianceGraph.cgOneFactorFit	108

Index**112**

cg-package	<i>Comparison of groups</i>
------------	-----------------------------

Description

cg is comprehensive data analysis software, and stands for "compare groups." Its genesis and evolution are driven by common needs to compare samples, administrations, conditions, etc. in medicine research & development. The current version provides comparisons of unpaired samples, i.e. a linear model with one factor of at least two levels. Good data graphs, modern statistical methods, and useful displays of results are emphasized.

Details

Package: cg
 Type: Package
 Version: 0.9.0
 Date: 2010-12-10
 License: GPL (>= 2)
 LazyLoad: yes
 LazyData: yes
 Depends: R (>= 2.11.0), Hmisc, base, utils, methods, stats, graphics,
 grid, lattice, MASS, survival, multcomp, nlme Imports: VGAM

Author(s)

Bill Pikounis and John Oleynick

Maintainers: Bill Pikounis <cg@billpikounis.net> and John Oleynick <cg@billpikounis.net>

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

## Exploratory methods
pointGraph(canine.data)

boxplot(canine.data)

descriptiveTable(canine.data)

## Fits and Comparisons
canine.fit <- fit(canine.data)

canine.comps0 <- comparisonsTable(canine.fit)

errorBarGraph(canine.fit)

canine.comps1 <- comparisonsTable(canine.fit, mcadjust=TRUE,
                                 type="allgroupstocontrol", refgrp="CC")

comparisonsGraph(canine.comps1)

grpSummaryTable(canine.fit)

## Diagnostics
varianceGraph(canine.fit)

qqGraph(canine.fit)

downweightedTable(canine.fit, cutoff=0.95)

## Sample Size calculations
canine.samplesize <- samplesizeTable(canine.fit, direction="increasing",
                                     mmdvec=c(10, 25, 50, 75, 100))

samplesizeGraph(canine.samplesize)

## Censored Data Set
data(gmcsfcens)
gmcsfcens.data <- prepareCGOneFactorData(gmcsfcens, format="groupcolumns",
                                         analysisname="cytokine",
                                         endptname="GM-CSF (pg/ml)",
                                         logscale=TRUE)

pointGraph(gmcsfcens.data)
boxplot(gmcsfcens.data)
descriptiveTable(gmcsfcens.data)

gmcsfcens.fit <- fit(gmcsfcens.data, type="aft")
```

 boxplot.cgOneFactorData

Box Plot Graph of Groups from a cgOneFactorData object

Description

Create graph of boxplots of groups in a cgOneFactorData object.

Usage

```
## S4 method for signature 'cgOneFactorData'
boxplot(x, ...)
```

Arguments

x A [cgOneFactorData](#) object.

... Additional arguments, both *optional*. Two are currently valid:

logscale A logical value, indicating whether or not the point graph should be plotted on the logarithmic scale. If `logscale` is not specified, its value is taken from the `cgOneFactorData` object, which [prepareCGOneFactorData](#) sets from its `logscale` argument.

ticklabels A list of two components:

- mod** Can be either of these two values,
 - "replace" Before graphing the data, remove any automatically generated tickmarks for the y-axis, and create the tickmarks specified in the marks component (see below).
 - "add" Before graphing the data, add tickmarks specified in the marks component to the automatically generated ones.

marks A vector of tickmarks to be placed on the y-axis. Any numeric representations will be coerced to character.

Details

For uncensored data, the boxplot for each group produced is a standard boxplot, similar to that produced by `graphics::boxplot.default`, but with the median shown as a "+" and the mean shown as a "o". A warning is added to the plot if any of the groups or all of the groups have 5 or fewer observations (in which case a plot from [pointGraph.cgOneFactorData](#) might be more suitable).

For censored data, Kaplan-Meier estimates are used for the quantiles, as proposed by Gentleman and Crowley (1991). The `survival::survfit` conventions are followed for interpolation of these quantiles. Extreme values that are censored are drawn as open arrow heads rather than open circles. Left-censored values are shown as a shallow "V", which is actually just a rotated downward ">" sign. Similarly, right-censored values are shown as a deeper "^", which is actually just a rotated upward ">" sign. Individual points are [jittered](#), and open circles are used for complete observations to alleviate potential overlap and the danger of representing multiple points as a single

point. Individual censored values are similarly jittered. With enough censored data observations in a group, certain quantiles may not be estimable, and thus a complete box would not appear.

If `logscale=TRUE`, the tick marks for the y-axis on the left side of the plot show original values, while the ticks mark for the y-axis on the right side of the graph show base 10 log values.

Tick marks are attempted to be chosen wisely. For log-scaled axes in particular, leading digits of 2, 5, and 10 for values are included if possible. Since the algorithm is empirical, the `ticklabels` argument is available for further refinement or complete replacement of tickmarks.

The heading for the graph is taken from the `cgOneFactorData` object, which `prepareCGOneFactorData` sets from its `analysisname` argument. The label for the y-axis is taken from the `cgOneFactorData` object, which `prepareCGOneFactorData` sets from its `endptname` argument. The number of decimal places printed in the ticks on the y-axis is taken from the `cgOneFactorData` object, which `prepareCGOneFactorData` sets from its `digits` argument.

The minimum and maximum values from the range of the data are respectively labeled in the bottom and top left corners of the graph region.

If group labels along the x-axis seem to overlap in the standard horizontal form, they will be rotated 45 degrees.

Value

`boxplot.cgOneFactorData` returns an invisible `NULL`. The main purpose is the side effect of graphing to the current device.

Note

Contact cg@billpikounis.net for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

References

Gentleman, R.C. and Crowley, J. (1991). "Graphical Methods for Censored Data", *Journal of the American Statistical Association*, Volume 86, 678-683.

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

boxplot(canine.data)

## Plot the data on the original scale instead of the log scale
boxplot(canine.data, logscale=FALSE)

## Censored Data
```

```
data(gmcsfcens)
gmcsfcens.data <- prepareCGOneFactorData(gmcsfcens, format="groupcolumns",
                                         analysisname="cytokine",
                                         endptname="GM-CSF (pg/mL)",
                                         logscale=TRUE)

boxplot(gmcsfcens.data)
```

canine

Canine Data Set in the cg package

Description

A data frame used to illustrate the **cg** package. It has a One Factor / One-Way / Unpaired Samples layout.

Usage

```
data(canine)
```

Format

A 5-by-5 data frame with 5 numeric observations from an experiment on the following 5 groups of beagle dogs.

AE castration plus estradiol and androstanediol
E1 castration plus low dose estradiol
E2 castration plus high dose estradiol
CC castration alone
NC no castration (normal controls)

Details

The canine data set that comes with the **cg** package is in `groupcolumns` format for the `prepareCGOneFactorData` call. Each column represents a group, and the observations in that group's column are the individual response or outcome values.

The 5 groups are regarded as levels of one factor in the `prepareCGOneFactorData`, `fit`, and other methods in the **cg** package.

An alternative format of this data set is contained in `canine.listfmt`. See that help file for details, including how it would be read and `prepared` by **cg**.

The purpose of this experiment was to evaluate the effect of a physiological dose of estradiol on prostate growth in dogs using ultrasound. See the reference below for details. Comparisons amongst all five groups are of interest.

Note

Contact `<cg@billpikounis.net>` for bug reports, questions, concerns, and comments.

References

Rhodes, L., Ding, V.D.H., Kemp, R.K., Khan, M.S., Nakhla, A.M., Pikounis, B., Rosner, W., Saunders, H.M. and Feeney, W.P. (2000). "Estradiol causes a dose dependent stimulation of prostate growth in castrate beagle dogs." *The Prostate*, Volume 44, 8-18.

See Also

[canine.listfmt](#), [prepareCGOneFactorData](#)

Examples

```
data(canine)
str(canine)
```

canine.listfmt

Canine Data Set in the cg package

Description

A data frame used to illustrate the **cg** package. It has a One Factor / One-Way / Unpaired Samples layout.

Usage

```
data(canine.listfmt)
```

Format

A 25-by-2 data frame with 5 numeric observations from an experiment on each of the following 5 groups of beagle dogs.

AE castration plus estradiol and androstenediol

E1 castration plus low dose estradiol

E2 castration plus high dose estradiol

CC castration alone

NC no castration (normal controls)

The above 5 items are the levels of the first column's factor, named `grp`. The second column size contains the numeric observations.

cgInternalClasses *cg package Internal Virtual Classes*

Description

cg package Internal Virtual Classes designed for polymorphic slots

Details

The virtual classes

characterOrExpression

characterOrNULL

dataframeMatrixOrNULL

dataframeOrNULL

numericOrNULL

olsfit

rrfit

aftfit

uvfit

are used internally by the cg package, and designed as polymorphic slots.

Note

Contact <cg@billpikounis.net> for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

cgInternalUtilities *cg package Internal Utilities*

Description

cg package Internal Utility functions and objects not intended for user-level calls

Details

The functions

stndErr geoMean pctToRatio makeZeroScore unwind unwrap graphStampCG setupAxisTicks
 setupLog10AxisTicks tryAgain seeHelpFile paragraphWrap cgMessage factorInSeq
 setupGrpNameTicks xTicksCex yTicksCex rmTicks minmaxTicks plotGrpNameTicks boxplotStamp
 errorBarGraphStamp
 comparisonsGraphStamp errorBarGraphApproximateStamp trimWhiteSpace chopZeroes fmtRatioToPercent
 fmtDifference fmtRatio fmtPercent fmtPvalue cgDevice contrastMatrix blockDiag
 rangeExtend getNumDigits makeCensored multcompInform multcompDone isAllEqual
 makeEndptLabel catCharExpr
 residualgrptrend.helper fround fround.charcens chop.matrix stripmiss makeTickMarks
 scaleVar makeContrastVec cg.largest.empty qminmin unpaste grpsummary samplesize
 samplesizegraph

are used internally by the **cg** package. See source code for details.

The blockDiag function is adapted from a Ben Bolker function contribution on R-help in 2002.

The cg.largest.empty function is a simplification of the Hmisc package function largest.empty function in that it does not check if it being called under R.

The factorInSeq function is exported since it may be useful for a user. It is a simple wrapper around [factor](#) with the order of its levels determined by first occurrence of each level in its x vector argument.

Note

Contact <cg@billpikounis.net> for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

cgLineColors

Color Choice order for Graph Lines

Description

A vector named cgLineColors that assigns colors to lines for graphing methods in the **cg** package.

Usage

cgLineColors

Format

A vector containing ten values in this order: black blue green red orange brown yellow darkblue darkgreen darkgreen

Details

This is a package internal convenience, and not intended for use or modification by the end user.

If more than ten values are needed then recycling will occur.

Note

Contact <cg@billpikounis.net> for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

cgValidity

cg package Validity Checks

Description

cg package Internal Validity functions not intended for user-level calls. These are used on input arguments for methods and functions.

Details

validAlpha validPower validDataFormat validBoolean validCharacter validNumeric
validNumericOrCensored validList validAtomicVec validArgMatch validDotsArg validDotsArgs
getDotsArgName parsePartialName reportInvalidArg validEqualLength validArgMatch
validArgDigits validArgModel validCGOneFacGroupColDfr validCGOneFacListedDfr validCensor
validZeroScore validAddConstant validAft validFitType validErrorDf validComparisonType
validEstimates validGrpNames validN validCutoffWt validDenDf

are used internally by the **cg** package.

Note

Contact <cg@billpikounis.net> for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

comparisons

*Create a table of comparisons amongst groups***Description**

Create a table of comparisons based on group estimates and variance-covariance matrix. Pairwise or custom specified contrasts are estimated and tested.

Usage

```
comparisons(estimates, varcovmatrix, errordf = Inf, endptscale,
            mcadjust = FALSE, alpha = 0.05, type = "pairwisereflect",
            contrastmatrix = NULL, n, offset = NULL, cnames = "derive",
            analysisname = "", endptname = "", digits = NULL, addpct = FALSE,
            display = "print", ...)
```

Arguments

estimates	A named vector of estimates. Each estimate element is a measure of the center of the group. The name of each group must be present in the names attribute of the vector.
varcovmatrix	The estimated variance-covariance matrix associated with the estimates. Must have the same number of columns and rows as the length of the estimates vector.
errordf	Can be one of three types of values: Inf The default, and will just use standard Gaussian (normal) distribution quantile for the critical points in each comparison; numeric A finite positive number that will be used for the degrees of freedom for the t-distribution quantile; "approx" Will try to apply a Satterthwaite approximation based on the variance-covariance matrix and n to estimate the degrees of freedom for each comparison. The variance-covariance matrix will need to be diagonal. Only will be accepted when mcadjust=FALSE.
endptscale	Must be specified as "log" or "original". If "log" then the estimates vector is assumed to be in the log scale, and calculations will need to transform to the original scale for the generated table.
mcadjust	Do a multiple comparisons adjustment, based on the simultaneous inference capabilities of the multcomp package. See Details below. The default value is FALSE. If mcadjust=TRUE is specified, there will be a delay, usually just for a few seconds, due to computing time of the critical point in order to conduct the adjusted comparisons.
alpha	Significance level, by default set to 0.05 to correspond to 95% confidence.
type	Can be one of four values:

	"pairwisereflect" The default value, it calculates and lists all possible pairwise comparison permutations, as each pair ordering is included. In other words, Groups A vs. B and B vs. A will be included.
	"pairwise" Calculates and lists all possible pairwise comparison combinations. Once a pair such as Groups A vs. B is specified, then the reflective B vs. A is not included. So the number of comparisons presented is half that produced by "pairwisereflect". The ordering of group levels in the estimates vector is used to determine which ordering is included and which is not. If all orderings are of interest, such as for endptscale=="log", use the "pairwisereflect" value above.
	"allgroupstocontrol" Takes the first element group of the estimate vector, deems it the "control" group, and constructs pairwise comparisons of all other groups to it.
	"custom" Indicates that a custom matrix of comparisons will be constructed, and that matrix needs to be specified in the contrastmatrix argument.
contrastmatrix	Only relevant if type=="custom" is specified. In that case, a numeric matrix with the number of rows equal to the number of comparisons of interest is needed. The number of columns must be equal to the number of groups in the estimate vector. Each row in the matrix is assumed to represent a contrast of coefficients amongst the groups that defines the comparison of interest.
n	Needs to be specified only when errorrdf="approx". In this case it needs to be a vector of group sample sizes, the same in length as the estimates vector.
offset	<i>Optional</i> , If for example a numeric constant was added to all response values before calculation of the estimates as means in the "log" scale, this could be used to adjust the estimates and comparisons appropriately. The default NULL.
cnames	If the default value of derive is used, row names are derived for the table that reflect the A vs. B type of comparison items in each row, using the names attributes of the estimates vector and the middle term of "vs." Otherwise, this can be explicitly specified and needs to be a character vector of the same length as estimates
analysisname	<i>Optional</i> , a character text that will be printed along with the results table. The default value is the empty "".
endptname	<i>Optional</i> , a character text that will be printed along with the results table. The default value is the empty "".
digits	<i>Optional</i> , For output display purposes, values will be rounded to this numeric value. Only the integers of 0, 1, 2, 3, and 4 are accepted. No rounding is done during any calculations. The default value is NULL, which will examine each individual estimates value and choose the one that has the maximum number of digits after any trailing zeroes are ignored. The max number of digits will be 4.
addpct	Only relevant if endptscale=="original". An column of percent differences is added for the comparisons, as a descriptive supplement to the original scale differences that are formally estimated.
display	One of three valid values:

"print" The default value, it calls a print method for the created object, which is a formatted text output of the table(s).

"none" Suppresses any printing. Useful, for example, when just assignment of the resulting object is desired.

"show" Calls the default `showDefault` method, which will just print out the comparisons return object.

... Additional arguments. None are currently used.

Details

Only two-sided Wald-type of confidence intervals are possible with this function.

When `mcadjust=TRUE`, a status message of "Some time may be needed as the critical point from the `multcomp::summary.glm` function call is calculated" is displayed at the console. This computed critical point is used for all subsequent p-value and confidence interval calculations.

The **multcomp** package provides a unified way to calculate critical points based on the comparisons of interest in a "family." Thus a user does not need to worry about choosing amongst the myriad names of multiple comparison procedures.

Value

Creates a return data frame object that specifies the comparison of the form A vs. B in each row, and with these columns:

`estimate` The difference in group estimates in the comparison: A vs. B. If `endptscale="log"`, this will back-transformed to a percent difference scale.

`se` The estimated standard error of the difference estimate. If `endptscale="log"`, this estimate will be based on the Delta method, and will particularly begin to be a poor approximation when the standard error in the logscale exceeds 0.50.

`lowerci` The lower $100 * (1-\alpha) \%$ confidence limit of the difference estimate. With the default `alpha=0.05`, this is 95%. If `endptscale="log"`, the confidence limit is first computed in the logarithmic scale of analysis, and then back-transformed to a percent difference scale.

`upperci` The upper $100 * (1-\alpha) \%$ confidence limit of the difference estimate. With the default `alpha=0.05`, this is 95%. If `endptscale="log"`, the confidence limit is first computed in the logarithmic scale of analysis, and then back-transformed to a percent difference scale.

`pval` The computed p-value from the test of the difference estimate.

`meanA` **or** `geomeanA` The estimated "mean" for the left hand side "A" of the A vs. B comparison. If `endptscale="log"`, this is a back-transform to the original scale, and therefore is a "geometric" mean, and will be labelled `geomeanA`. Otherwise it is the arithmetic mean and labelled `meanA`.

`seA` The estimated standard error of the `meanA` estimate. If `endptscale="log"`, this estimate will be based on the Delta method, and will particularly begin to be a poor approximation when the standard error in the logscale exceeds 0.50.

`meanB` **or** `geomeanB` The estimated "mean" for the right hand side "B" of the A vs. B comparison. If `endptscale="log"`, this is a back-transform to the original scale, and therefore is a "geometric" mean, and will be labelled `geomeanB`. Otherwise it is the arithmetic mean and labelled `meanB`.

seB The estimated standard error of the meanB estimate. If endptscale="log", this estimate will be based on the Delta method, and will particularly begin to be a poor approximation when the standard error in the logscale exceeds 0.50.

An additional column addpct of percent differences is added if endptscale=="original" and addpct=TRUE, as a descriptive supplement to the original scale differences that are formally estimated.

Warning

This function was created for internal use in the **cg** package as its use can be seen in the [comparisonsTable](#) methods code. Therefore any direct use of it needs to be done cautiously.

Note

Contact <cg@billpikounis.net> for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

References

Hothorn, T., Bretz, F., Westfall, P., Heiberger, R.M., and Schuetzenmeister, A. (2010). The **mult-comp** package.

Hothorn, T., Bretz, F., and Westfall, P. (2008). "Simultaneous Inference in General Parametric Models", *Biometrical Journal*, 50, 3, 346-363.

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(canine.data)

## Easier way: notice the comparisonsTable call

comparisonsTable(canine.fit, model="olsonly")

## Manual way
## Instead of comparisonsTable(canine.fit, model="olsonly")

comparisons(estimates=canine.fit@olsfit$coef,
            varcovmatrix=vcov(canine.fit@olsfit),
            errorrdf=canine.fit@olsfit$df.residual,
            endptscale="log",
```

```
analysisname="Canine",
digits=1,
endptname="Prostate Volume")
```

comparisonsGraph	<i>Graph comparisons specified amongst groups</i>
------------------	---

Description

Generic function to create a Comparisons Graph based on a Comparisons Table created in turn by the **cg** package.

Usage

```
comparisonsGraph(compstable, cgtheme=TRUE, device="single",
  wraplength=20, cex.comps=0.7, ...)
```

Arguments

compstable	A comparisonsTable object created by a comparisonsTable method from the cg package. The only class of object currently available is cgOneFactorComparisonsTable , which is prepared by the comparisonsTable.cgOneFactorFit method.
cgtheme	When set to the default TRUE, ensures a trellis device is active with limited color scheme. Namely, background, strip.shingle and strip.background are each set to "white".
device	Can be one of three values: "single" The default, which will put all graph panels on the same device page. "multiple" Relevant only when more than one panel of graphs is possible. In that case, a new graphics device is generated each newly generated single-paneled graph. "ask" Relevant only when more than one panel of graphs is possible. In that case, each are portrayed as a single-paneled graph, with the ask=TRUE argument specified in par so that the user input confirmation is needed before the graphs are drawn.
wraplength	On the left hand vertical axis are each A vs. B comparison label from the compstable object. An attempt at sensible formatting when a newline is needed is made, but adjustment by this argument may be needed. The default is 20 characters before wrapping to a newline.
cex.comps	Similar to wraplength, adjustment of this argument parameter can be made to fit the comparison labels on the left hand vertical axis.
...	Additional arguments, depending on the specific method written for the compstable object. Currently, there is only one such specific method; see comparisonsGraph.cgOneFactorComparisonsTable for any additional arguments that can be specified.

Value

The main purpose is the side effect of graphing to the current device. See the specific methods for discussion of any return values.

Note

Contact <cg@billpikounis.net> for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

See Also

[comparisonsGraph.cgOneFactorComparisonsTable](#)

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(canine.data)

canine.comps1 <- comparisonsTable(canine.fit, mcadjust=TRUE,
                                 type="allgroupstocontrol", refgrp="CC")

comparisonsGraph(canine.comps1)
```

comparisonsgraph *Create an graph of comparisons*

Description

Creates a graph to see comparisons based on group estimates and variance-covariance matrix

Usage

```
comparisonsgraph(compstable, diffstype, analysisname = "",
                 endptname = "", alpha = 0.05, digits = NULL,
                 titlestamp = TRUE, explanation = TRUE,
                 wraplength = 20, cex.comps = 0.7,
                 ticklabels = NULL, ...)
```

Arguments

compstable	A data frame object of form like that created by the <code>comparisons</code> function.
difftype	Must be specified as one of the following: "percent" Presumes the estimated differences in <code>compstable</code> are Percent, and thus will space the x-axis logarithmically; "amount" Presumes the estimated differences in <code>compstable</code> are Simple Amounts, and thus will space the x-axis in untransformed scale; "simple" Synonym for amount, presumes the estimated differences in <code>compstable</code> are Simple Amounts, and thus will space the x-axis in untransformed scale.
analysisname	<i>Optional</i> , a character text or math-valid expression that will used in the graph title. The default value is the empty "".
endptname	<i>Optional</i> , a character text or math-valid expression that that will be used as the y-axis label of the graph. The default value is the empty "".
alpha	Significance level, by default set to 0.05. This is only used for labelling purposes.
digits	<i>Optional</i> , For output display purposes in the graph, values will be rounded to this numeric value. Only the integers of 0, 1, 2, 3, and 4 are accepted. No rounding is done during any calculations. The default value is NULL, which will examine each individual data value and choose the one that has the maximum number of digits after any trailing zeroes are ignored. The max number of digits will be 4.
explanation	If TRUE, which is the default, add explanatory message to the graph rendering about "Error bars that do not cross the zero line indicate statistically significant difference(s)" along with the confidence level derived from alpha.
titlestamp	Specify text to the graph in the top of graph area, otherwise a default description of "Comparisons Graph" and <code>analysisname</code> will be constructed.
wraplength	On the left hand axis are each A vs. B comparison label from the <code>compstable</code> data frame. An attempt at sensible formatting when a newline is needed is made, but adjustment by this argument may be needed. The default is 20 characters before wrapping to a newline.
cex.comps	Similar to <code>wraplength</code> , adjustment of this argument parameter can be made to fit the comparison labels on the left hand axis.
ticklabels	<i>Optional</i> , before graphing the data, remove any automatically generated tickmarks for the y-axis, and use these tickmarks instead. A vector of tickmarks to be placed on the y-axis. Any numeric representations will be coerced to character.
...	Additional arguments. None are currently used.

Details

The minimum and maximum values across all the bar ends are added inside the plot region in blue, flush against the x-axis. In two panel cases, there is a tendency to fall outside the panel area even though right justified is used for the `adj` parameter of functions like `panel.text`.

Value

comparisonsgraph returns an invisible NULL. The main purpose is the side effect of graphing to the current device.

Warning

This function was created for internal use in the **cg** package as its use can be seen in the [comparisonsGraph](#) methods source code. Therefore any direct use of it needs to be done cautiously.

Note

Contact <cg@billpikounis.net> for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

See Also

[comparisons](#)

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(canine.data)
canine.comps <- comparisonsTable(canine.fit)

## Easier way: notice the camel-case of the comparisonsGraph call

comparisonsGraph(canine.comps, model="olonly")

## Manual way
## Instead of comparisonsGraph(canine.comps, model="olonly")

canine.compstable <- comparisons(estimates=canine.fit@olsfit$coef,
                                varcovmatrix=vcov(canine.fit@olsfit),
                                errordf=canine.fit@olsfit$df.residual,
                                endptscale="log",
                                analysisname="Canine",
                                digits=1,
                                endptname="Prostate Volume")

comparisonsgraph(canine.compstable,
                 difftype="percent",
                 analysisname="Canine",
                 digits=1,
```

```

endptname=expression(paste( plain('Prostate Volume'),
                           ' (', plain(cm)^3 , ')' ))
)

```

comparisonsGraph.cgOneFactorComparisonsTable

Create an graph of the comparisons in a cgOneFactorComparisonsTable object

Description

Creates a graph to see comparisons in a cgOneFactorComparisonsTable object

Usage

```

## S4 method for signature 'cgOneFactorComparisonsTable'
comparisonsGraph(compstable, cgtheme=TRUE, device="single",
  wraplength = 20, cex.comps = 0.7, ...)

```

Arguments

compstable	A comparisonsTable object created by a comparisonsTable method from the cg package. The only class of object currently available is cgOneFactorComparisonsTable , which is prepared by the comparisonsTable.cgOneFactorFit method.
cgtheme	When set to the default TRUE, ensures a trellis device is active with limited color scheme. Namely, background, strip.shingle, and strip.background are each set to "white".
device	Can be one of three values: "single" The default, which will put all graphs on the same device page. For example, when resistant & robust and classical least squares are present and model=both (the default), a 2 x 1 paneled graph will be created. "multiple" Relevant only when resistant & robust and classical least squares are present and model=both (the default). In that case, a new graphics device is generated to hold the resistant & robust version, as a single-paneled graph. The classical least squares version is on the previous device. "ask" Relevant only when resistant & robust and classical least squares are present and model=both (the default). In that case, each are portrayed as a single-paneled graph, with the ask=TRUE argument specified in par so that the user input confirmation is needed before the graphs are drawn.
wraplength	On the left hand axis are each A vs. B comparison label from the compstable object. An attempt at sensible formatting when a newline is needed is made, but adjustment by this argument may be needed. The default is 20 characters before wrapping to a newline.
cex.comps	Similar to wraplength, adjustment of this argument parameter can be made to fit the comparison labels on the left hand axis.

... Additional arguments. Two are currently valid:

model For `cgOneFactorOneFactorComparisonsTable` objects that have classical least squares `lm()` or resistant & robust `rlm()` table slots, the following argument values are possible:

- "both" Graphs of Comparisons Tables based on both the ordinary classical least squares and resistant & robust slots are populated. This is the default when both slots are present in the `cgOneFactorComparisonsTable` object specified in the `compstable` argument. If the resistant & robust fit is not available, this value is not relevant.
- "olsonly" Only an Comparisons Graph based on the ordinary classical least squares table slot is performed.
- "rronly" Only a Comparisons Graph based on the resistant and robust table slot is performed.

For other possible `cgOneFactorComparisonsTable` table slots such as accelerated failure time or unequal variance models, the `model` argument is not relevant, and the appropriate comparisons graph will be calculated for these model types.

ticklabels A list of two components:

- mod** Can be either of these two values,
 - "replace" Before graphing the data, remove any automatically generated tickmarks for the x-axis, and create the tickmarks specified in the `marks` component below.
 - "add" Before graphing the data, add tickmarks specified in the `marks` component to the automatically generated ones.
- marks** A vector of tickmarks to be placed on the x-axis. Any numeric representations will be coerced to character.

Details

The minimum and maximum values across all the bar ends are added inside the plot region in blue, flush against the x-axis. In two panel cases, there is a tendency to fall outside the panel area even though right justified is used for the `adj` parameter of functions like `panel.text`. The number of decimal places are determined by the `digits` and `endptscale` values in the `compstable@settings` slot.

Value

`comparisonsGraph.cgOneFactorComparisonsTable` returns an invisible NULL. The main purpose is the side effect of graphing to the current device.

Note

Contact <cg@billpikounis.net> for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

See Also

[cgOneFactorComparisonsTable](#)

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(canine.data)

## Comparisons Tables
canine.comps0 <- comparisonsTable(canine.fit)

canine.comps1 <- comparisonsTable(canine.fit, mcadjust=TRUE,
                                 type="allgroupstocontrol", refgrp="CC")

## Comparisons Graphs
comparisonsGraph(canine.comps0)

comparisonsGraph(canine.comps1)

comparisonsGraph(canine.comps1, cex.comps=0.9,
                 ticklabels=list(mod="add", marks=c(300, 700)))
```

<code>comparisonsTable</code>	<i>Create a Table of Comparisons amongst Groups</i>
-------------------------------	---

Description

Create a table of comparisons based on a fit by the **cg** package.

Usage

```
comparisonsTable(fit, mcadjust = FALSE, type = "pairwisereflect",
                 contrastmatrix = NULL, refgrp = NULL, alpha = 0.05, adpct = FALSE,
                 display = "print", ...)
```

Arguments

<code>fit</code>	A fit object created with a <code>fit</code> method from the cg package. The only class of object currently available is <code>cgOneFactorFit</code> , which is prepared by the <code>fit.cgOneFactorData</code> method.
------------------	---

mcadjust	Do a multiple comparisons adjustment, based on the simultaneous inference capabilities of the multcomp package. See Details below. The default value is FALSE. If mcadjust=TRUE is specified, there will be a delay, usually just for a few seconds, due to computing time of the critical point in order to conduct the adjusted comparisons.
type	Can be one of four values: <p>"pairwisereflect" The default value, it calculates and lists all possible pairwise comparison permutations, as each pair order is included. In other words, Groups A vs. B and B vs. A will be included.</p> <p>"pairwise" Calculates and lists all possible pairwise comparison combinations. Once a pair such as Groups A vs. B is specified, then the reflective B vs. A is not included. So the number of comparisons is half that produced by "pairwisereflect". The ordering of group levels in the fit object is used to determine which ordering is included and which is not. If all orderings are of interest, such as for settings\$endptscale=="log" in the fit object, use the "pairwisereflect" value above.</p> <p>"allgroupstocontrol" Takes the value of settings\$refgrp in the cg fit object, deems it the "control" group, and constructs pairwise comparisons of all other groups to it.</p> <p>"custom" Indicates that a custom matrix of comparisons will be constructed, and that matrix needs to be specified in the contrastmatrix argument.</p>
contrastmatrix	Only relevant if type="custom" is specified. In that case, a numeric matrix with the number of rows equal to the number of comparisons of interest is needed. The number of columns must be equal to the number of estimated means. Each row in the matrix is assumed to represent a contrast of coefficients amongst the group means that defines the comparison of interest.
refgrp	If left at the default value of NULL, it will be set to the settings\$refgrp value in the cg fit object. When set, it is deemed the "reference", or "control" group, so that pairwise comparisons of all other groups to it will be constructed when type="allgroupstocontrol".
alpha	Significance level, by default set to 0.05.
addpct	Only relevant if settings\$endptscale=="original" in the fit object. An column of percent differences is added for the comparisons, as a descriptive supplement to the original scale differences that are formally estimated.
display	One of three valid values: <p>"print" The default value, it calls a print method for the created ComparisonsTable object, which is a formatted text output of the table(s).</p> <p>"none" Suppresses any printing. Useful, for example, when just assignment of the resulting object is desired.</p> <p>"show" Calls the default showDefault method, which will just print out the ComparisonsTable object components.</p>
...	Additional arguments, depending on the specific method written for the object. See the method-specific documentation for additional details.

Details

When `mcadjust=TRUE`, a status message of "Some time may be needed as the critical point from the `multcomp::summary.glt` function call is calculated" is displayed at the console. This computed critical point is used for all subsequent p-value and confidence interval calculations.

The **multcomp** package provides a unified way to calculate critical points based on the comparisons of interest in a "family". Thus a user does not need to worry about choosing amongst the myriad names of multiple comparison procedures.

Value

A method-specific `comparisonsTable` object is returned. See the specific methods for discussion of return values.

Note

Contact cg@billpikounis.net for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

References

Hothorn, T., Bretz, F., Westfall, P., Heiberger, R.M., and Schuetzenmeister, A. (2010). The `multcomp` package.

Hothorn, T., Bretz, F., and Westfall, P. (2008). "Simultaneous Inference in General Parametric Models", *Biometrical Journal*, 50, 3, 346-363.

See Also

[comparisonsTable.cgOneFactorFit](#)

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(canine.data)

canine.comps0 <- comparisonsTable(canine.fit)

canine.comps1 <- comparisonsTable(canine.fit, mcadjust=TRUE,
                                 type="allgroupstocontrol", refgrp="CC")

data(gmcsfcens)
```

```
gmcsfcens.data <- prepareCGOneFactorData(gmcsfcens, format="groupcolumns",
                                         analysisname="cytokine",
                                         endptname="GM-CSF (pg/mL)",
                                         logscale=TRUE)

gmcsfcens.fit <- fit(gmcsfcens.data, type="aft")

gmcsfcens.comps <- comparisonsTable(gmcsfcens.fit)
```

```
comparisonsTable.cgOneFactorFit
```

Create a table of comparisons amongst groups with the cgOneFactorFit object

Description

Create a table of comparisons based on the `cgOneFactorFit` object. Pairwise or custom specified contrasts are estimated and tested. A `cgOneFactorComparisonsTable` class object is created.

Usage

```
## S4 method for signature 'cgOneFactorFit'
comparisonsTable(fit, mcadjust=FALSE, type="pairwisereflect",
                 contrastmatrix=NULL, refgrp=NULL, alpha=0.05, addpct=FALSE,
                 display="print", ...)
```

Arguments

<code>fit</code>	An object of class <code>cgOneFactorFit</code> .
<code>mcadjust</code>	Do a multiple comparisons adjustment, based on the simultaneous inference capabilities of the multcomp package. See Details below. The default value is <code>FALSE</code> . If <code>mcadjust=TRUE</code> is specified, there will be a delay, usually just for a few seconds, due to computing time of the critical point in order to conduct the adjusted comparisons.
<code>type</code>	Can be one of four values: <ul style="list-style-type: none"> "pairwisereflect" The default value, it calculates and lists all possible pairwise comparison permutations, as each pair order is included. In other words, Groups A vs. B and B vs. A will be included. "pairwise" Calculates and lists all possible pairwise comparison combinations. Once a pair such as Groups A vs. B is specified, then the reflective B vs. A is not included. So the number of comparisons is half that produced by "pairwisereflect". The ordering of group levels in the fit object is used to determine which ordering is included and which is not. If all orderings are of interest, such as for <code>settings\$endptscale="log"</code> in the fit objects, use the "pairwisereflect" value above.

	"allgroupstocontrol" Takes the value of settings\$refgrp in the cg fit object, deems it the "control" group, and constructs pairwise comparisons of all other groups to it.
	"custom" Indicates the a custom matrix of comparisons will be constructed, and that matrix needs to be specified in the contrastmatrix argument.
contrastmatrix	Only relevant if type="custom" is specified. In that case, a numeric matrix with the number of rows equal to the number of comparisons of interest. The number of columns must be equal to the number of group means. Each row in the matrix is assumed to represent a contrast of coefficients amongst the groups that defines the comparison of interest.
refgrp	If left at the default value of NULL, it will be set to the settings\$refgrp value in the cg fit object. When set, it is deemed the "reference", or "control" group, so that pairwise comparisons of all other groups to it will be constructed when type="allgroupstocontrol".
alpha	Significance level, by default set to 0.05.
addpct	Only relevant if settings\$endptscale=="original" in the fit object. An column of percent differences is added for the comparisons, as a descriptive supplement to the original scale differences that are formally estimated.
display	One of three valid values: "print" The default value, it calls a print method for the created cgOneFactorComparisonsTable object, which is a formatted text output of the table(s). "none" Suppresses any printing. Useful, for example, when just assignment of the resulting object is desired. "show" Calls the default showDefault method, which will just print out the cgOneFactorComparisonsTable components.
...	Additional arguments. Only one is currently valid:
model	For cgOneFactorFit fit objects that have classical least squares lm or resistant & robust rlm fits, the following argument values are possible: "both" Comparison tables based on both the ordinary classical least squares and resistant & robust fits are created. This is the default when both fits are present in the cgOneFactorFit object specified in the fit argument. If the resistant & robust fit is not available, this value is not relevant. "olsonly" Only a comparison table based on the ordinary classical least squares olsfit fit slot is performed. "rronly" Only a comparison table based on the resistant and robust rrfit fit slot is performed.

For other possible cgOneFactorFit fit components such as accelerated failure time or unequal variance models, the model argument is not relevant, and the appropriate comparisons table will be calculated for these model types.

Details

When `mcadjust=TRUE`, a status message of "Some time may be needed as the critical point from the `multcomp::summary.glht` function call is calculated" is displayed at the console. This computed critical point is used for all subsequent p-value and confidence interval calculations.

The **multcomp** package provides a unified way to calculate critical points based on the comparisons of interest in a "family". Thus a user does not need to worry about choosing amongst the myriad names of multiple comparison procedures.

Value

Creates an object of class `cgOneFactorComparisonsTable`, with the following slots:

`ols.comprs` The table of comparisons based on the `olsfit` component of the `cgOneFactorFit`, unless `model="rronly"` is specified. In that case the slot value is `NULL`. Will not be appropriate in the case where a valid `aftfit` component is present in the `cgOneFactorFit` object. See below for the data frame structure of the table.

`rr.comprs` The table of comparisons based on the `rrfit` component of the `cgOneFactorFit` object, if a valid resistant & robust fit object is present. If `rrfit` is a simple character value of "No fit was selected.", or `model="olonly"` was specified, then the value is `NULL`. See below for the data frame structure of the table.

`aft.comprs` The table of comparisons based on the `aftfit` component of the `cgOneFactorFit` object if a valid accelerated failure time fit object is present. If `aftfit` is a simple character value of "No fit was selected.", then the value is `NULL`. See below for the data frame structure of the table.

`uv.comprs` The table of comparisons based on the `uvfit` component of the `cgOneFactorFit` object if a valid unequal variances fit object is present. The error degrees of freedom for each comparison estimate and test is individually estimated with a Satterthwaite approximation. See below for the data frame structure of the table.

`settings` A list of settings carried from the `cgOneFactorFit` fit object, and the addition of some specified arguments in the method call above: `alpha`, `mcadjust`, `type`, and `addpct`. These are used for the `print.cgOneFactorComparisonsTable` method, invoked for example when `display="print"`.

The data frame structure of the comparisons table in a `*.comprs` slot consists of `row.names` that specify the comparison of the form A vs. B, and these columns:

`estimate` The difference in group means in the comparison: A vs. B. If `settings$endptscale=="log"` in the fit object, this will back-transformed to a percent difference scale.

`se` The estimated standard error of the difference estimate. If `settings$endptscale=="log"` in the fit object, this estimate will be based on the Delta method, and will particularly begin to be a poor approximation when the standard error in the logscale exceeds 0.50.

`lowerci` The lower $100 * (1-\alpha) \%$ confidence limit of the difference estimate. With the default `alpha=0.05`, this is 95%. If `settings$endptscale=="log"` in the fit object, the confidence limit is first computed in the logarithmic scale of analysis, and then back-transformed to a percent difference scale.

upperci The upper $100 * (1-\alpha)$ % confidence limit of the difference estimate. With the default $\alpha=0.05$, this is 95%. If `settings$endptscale=="log"` in the fit object, the confidence limit is first computed in the logarithmic scale of analysis, and then back-transformed to a percent difference scale.

pval The computed p-value from the test of the difference estimate.

meanA **or** **geomeanA** The estimated mean for the left hand side "A" of the A vs. B comparison. If `settings$endptscale=="log"` in the fit object, this is a back-transform to the original scale, and therefore is a geometric mean, and will be labelled **geomeanA**. Otherwise it is the arithmetic mean and labelled **meanA**.

seA The estimated standard error of the meanA estimate. If `settings$endptscale=="log"` in the fit object, this estimate will be based on the Delta method, and will particularly begin to be a poor approximation when the standard error in the logscale exceeds 0.50.

meanB **or** **geomeanB** The estimated mean for the right hand side "B" of the A vs. B comparison. If `settings$endptscale=="log"` in the fit object, this is a back-transform to the original scale, and therefore is a geometric mean, and will be labelled **geomeanB**. Otherwise it is the arithmetic mean and labelled **meanB**.

seB The estimated standard error of the meanB estimate. If `settings$endptscale=="log"` in the fit object, this estimate will be based on the Delta method, and will particularly begin to be a poor approximation when the standard error in the logscale exceeds 0.50.

An additional column **addpct** of percent differences is added if `endptscale=="original"` and `addpct=TRUE`, as a descriptive supplement to the original scale differences that are formally estimated.

Note

Contact <cg@billpikounis.net> for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

References

Hothorn, T., Bretz, F., Westfall, P., Heiberger, R.M., and Schuetzenmeister, A. (2010). The `multcomp` package.

Hothorn, T., Bretz, F., and Westfall, P. (2008). "Simultaneous Inference in General Parametric Models", *Biometrical Journal*, 50, 3, 346-363.

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(canine.data)
```

```

canine.comps0 <- comparisonsTable(canine.fit)

canine.comps1 <- comparisonsTable(canine.fit, mcadjust=TRUE,
                                  type="allgroupstocontrol", refgrp="CC")

data(gmcsfcens)
gmcsfcens.data <- prepareCGOneFactorData(gmcsfcens, format="groupcolumns",
                                          analysisname="cytokine",
                                          endptname="GM-CSF (pg/ml)",
                                          logscale=TRUE)

gmcsfcens.fit <- fit(gmcsfcens.data, type="aft")

gmcsfcens.comps <- comparisonsTable(gmcsfcens.fit)

```

descriptiveTable	<i>Compute Descriptive Summary Statistics of Groups</i>
------------------	---

Description

Create a table of quantiles and other summary statistics of the data in a **cg** data object.

Usage

```
descriptiveTable(data, display = "print", ...)
```

Arguments

data	A data object created and prepared (see prepare) using the cg package. The only class of object currently valid is cgOneFactorData , which is created by the prepareCGOneFactorData function.
display	One of three valid values: "print" The default value, it calls a print method for the created descriptiveTable object, which is a formatted text output of the table. "none" Suppresses any printing. Useful, for example, when just assignment of the resulting object is desired. "show" Calls the default showDefault method, which will just print out the descriptiveTable components.
...	Additional arguments. Currently only one is valid: logscale A logical value, indicating whether or not the geometric means and standard errors should be included in the summary. If logscale is not specified (default), its value is taken from the cgOneFactorData object, which prepareCGOneFactorData sets from its logscale argument.

Value

A method-specific descriptiveTable object is returned. See the specific methods for discussion of return values.

Note

Contact <cg@billpikounis.net> for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

See Also

[descriptiveTable.cgOneFactorData](#)

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

descriptiveTable(canine.data)

## Censored Data
data(gmcsfcens)
gmcsfcens.data <- prepareCGOneFactorData(gmcsfcens, format="groupcolumns",
                                         analysisname="cytokine",
                                         endptname="GM-CSF (pg/ml)",
                                         logscale=TRUE)

descriptiveTable(gmcsfcens.data)
```

descriptiveTable.cgOneFactorData

Compute Descriptive Summary Statistics of Groups in a cgOneFactor-Data object

Description

Create a table of quantiles and other summary statistics of the data in a [cgOneFactorData](#) object.

Usage

```
## S4 method for signature 'cgOneFactorData'
descriptiveTable(data, display = "print", ...)
```

Arguments

data	A cgOneFactorData object, typically created by prepareCGOneFactorData .
display	One of three valid values: "print" The default value, it calls a print method for the created descriptiveTable object, which is a formatted text output of the table. "none" Suppresses any printing. Useful, for example, when just assignment of the resulting object is desired. "show" Calls the default showDefault method, which will just print out the descriptiveTable components.
...	Additional arguments. Currently only one is valid: logscale A logical value, indicating whether or not the geometric means and standard errors should be included in the summary. If logscale is not specified, its value is taken from the cgOneFactorData object, which prepareCGOneFactorData sets from its logscale argument.

Details

The returned table contains quantiles, means, sample sizes, and estimates of variability for each group. If censored data are present, the estimated quantiles accommodate that with the Kaplan-Meier method, following Gentleman and Crowley (1991). The number of censored / incomplete and number of complete observations are also included when censored data is present in any of the groups. If the logscale option is specified (either explicitly, or implicitly from the cgOneFactorData object), then the geometric mean and geometric standard error for each group are also included. See the Value section below for details.

Value

Creates an object of class cgOneFactorDescriptiveTable, with the following slots:

contents The table of descriptive summary statistics for each group. See below for the data frame structure of the table.

settings A list of settings carried from the [cgOneFactorData](#) data object. These are used for the [print.cgOneFactorDescriptiveTable](#) method, invoked for example when display="print".

The data frame structure of the descriptive table in a contents slot consists of row.names that specify the group, and these columns:

n The sample size of the group.

Min The minimum value of the group.

25%ile The 25th percentile of the group, estimated with the [quantile](#) function.

Median The median value of the group.

75%ile The 75th percentile of the group, estimated with the [quantile](#) function.

Max The maximum value of the group.

Mean The arithmetic mean value of the group.

StdDev The standard deviation value of the group.

StdErr The standard error value of the group.

If logscale=TRUE, then two additional columns are added:

GeoMean The geometric mean value of the group.

SEGeoMean The estimated standard error associated with the geometric mean. This is calculated with the Delta Method, and will particularly lose accuracy in its useful approximation once the standard error in the log scale exceeds 0.50. A warning message is issued when this occurs.

If censored data are present in the cgOneFactorData object, then two more columns are added:

ncensored The number of censored / incomplete observations.

ncomplete The number of complete observations.

These two ncensored and ncomplete quantities will add up to n above and be placed adjacent to it.

The presence of censored observations will convert columns such as the Min and Max to character values, with the appropriate ">" and "<" symbols for right-censoring and left-censoring, respectively.

For censored data, Kaplan-Meier estimates are used for the quantiles, as proposed by Gentleman and Crowley (1991). The survreg::survfit conventions are followed for interpolation of these quantiles. With enough censored data observations in a group, certain quantiles may not be estimable. If any censored observations are present, the mean, geometric mean, and associated standard errors will not be calculated. The <NA> character representation is used.

Note

Contact <cg@billpikounis.net> for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

References

Gentleman, R.C. and Crowley, J. (1991). "Graphical Methods for Censored Data", *Journal of the American Statistical Association*, Volume 86, 678-683.

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

descriptiveTable(canine.data)

## Remove the geometric mean and standard error columns
descriptiveTable(canine.data, logscale=FALSE)
```

```
## Censored Data
data(gmcsfcens)
gmcsfcens.data <- prepareCGOneFactorData(gmcsfcens, format="groupcolumns",
                                         analysisname="cytokine",
                                         endptname="GM-CSF (pg/ml)",
                                         logscale=TRUE)

descriptiveTable(gmcsfcens.data)
```

downweightedTable *Create a table of downweighted observations from a Resistant & Robust fit.*

Description

Create a table of downweighted observations in a resistant & robust fit with the **cg** package.

Usage

```
downweightedTable(fit, cutoffwt, display="print", ...)
```

Arguments

fit	An object created by calling a <code>fit</code> method from the cg package. The only class of object currently valid is <code>cgOneFactorDownweightedTable</code> , which is prepared by the <code>downweightedTable.cgOneFactorFit</code> method.
cutoffwt	It has no default and must be specified as a numeric between 0 and 1 exclusive. It is a threshold on which observations to identify. All observations that exceed this specified value will be identified. For example, a <code>cutoffwt=0.90</code> will yield those observations that were downweighted by at least 10%.
display	One of three valid values: " <code>print</code> " The default value, it calls a <code>print</code> method for the created <code>downweightedTable</code> object, which is a formatted text output of the table(s). " <code>none</code> " Suppresses any printing. Useful, for example, when just assignment of the resulting object is desired. " <code>show</code> " Calls the default <code>showDefault</code> method, which will just print out the <code>downweightedTable</code> components.
...	Additional arguments, depending on the specific method written for the object. See the method-specific documentation for additional details.

Value

A method-specific `downweightedTable` object is returned. See the specific methods for discussion of return values.

Note

Contact <cg@billpikounis.net> for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

References

Venables, W. N. and Ripley, B. D. (2002), *Modern Applied Statistics with S*. Fourth edition. Springer.

See Also

[downweightedTable.cgOneFactorFit](#), MASS::rlm

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(canine.data)

canine.dwtable <- downweightedTable(canine.fit, cutoff=0.95)

downweightedTable(canine.fit, cutoff=0.75) ## No observation
                                         ## downweighted at least 25%
```

downweightedTable.cgOneFactorFit

Create a table of downweighted observations from a Resistant & Robust fit.

Description

Create a table of downweighted observations based on a rrfit object within a cgOneFactorFit object. A cgOneFactorDownweightedTable class object is created.

Usage

```
## S4 method for signature 'cgOneFactorFit'
downweightedTable(fit, cutoffwt, display="print", ...)
```

Arguments

fit	A fit object of class <code>cgOneFactorFit</code> .
cutoffwt	It has no default and must be specified as a numeric between 0 and 1 exclusive. It is a threshold on which observations to identify. All observations that exceed this specified value will be identified. For example, a <code>cutoffwt=0.90</code> will yield those observations that were downweighted by at least 10%.
display	One of three valid values: "print" The default value, it calls a <code>print</code> method for the created <code>cgOneFactorDownweightedTable</code> object, which is a formatted text output of the table(s). "none" Suppresses any printing. Useful, for example, when just assignment of the resulting object is desired. "show" Calls the default <code>showDefault</code> method, which will just print out the <code>cgOneFactorDownweightedTable</code> components.
...	Additional arguments. None are currently defined for this method.

Details

If no observations meet the cutoff criteria, a text message of the `cgOneFactorDownweightedTable` content emptiness is output instead.

The reported weights are in the scale of the observation, not the sum of squared errors representation for the likelihood. Thus they are derived from the square root of the w component from a MASS::rlm fit object.

Value

An object of class `cgOneFactorDownweightedTable`, with the following slots:

`contents` A data frame where each row is an observation from the fitted data set that meets the cutoff criteria, and these columns:

`group` The group identified from the fitted data.

`endpoint` The observed response value.

`weight` The weight associated to the observation from the resistant / robust fit.

`pct down-weighted` An expression of the weight in terms of percent reduction from the maximum of 1.

If no observations meet the cutoff criteria, the `contents` slot is set to `NULL`.

`cutoffwt` Taken from the specified `cutoffwt` argument value.

`settings` A list of settings carried from the `cgOneFactorFit` object, and the addition of the specified `cutoffwt` argument in the method call above. These are used for the `print.cgOneFactorDownweightedTable` method, invoked for example when `display="print"`.

Note

Contact <cg@billpikounis.net> for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

References

Venables, W. N. and Ripley, B. D. (2002), *Modern Applied Statistics with S*. Fourth edition. Springer.

See Also

[cgOneFactorFit](#), [MASS::rlm](#)

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(canine.data)

canine.dwttable <- downweightedTable(canine.fit, cutoff=0.95)

downweightedTable(canine.fit, cutoff=0.75) ## No observation
                                         ## downweighted at least 25%
```

errorBarGraph

Create an Error Bar graph amongst groups

Description

Generic function to create a Error Bar graph based on a fit by the **cg** package.

Usage

```
errorBarGraph(fit, mcadjust=FALSE, alpha = 0.05,
              cgtheme = TRUE, device="single", ...)
```

Arguments

fit	A fit object created by a fit method from the cg package. The only class of object currently available is cgOneFactorFit , which is prepared by the fit.cgOneFactorData method.
mcadjust	Do a multiple comparisons adjustment, based on the simultaneous inference capabilities of the multcomp package. See Details below. The default value is FALSE. If mcadjust=TRUE is specified, there will be a delay, usually just for a few seconds, due to computing time of the critical point in order to create the error bar intervals.

alpha	Significance level, by default set to 0.05 so that confidence levels are 95%.
cgtheme	When set to the default TRUE, ensures a trellis device is active with limited color scheme. Namely background, strip.shingle, and strip.background are each set to "white".
device	Can be one of three values: "single" The default, which will put all graph panels on the same device page. "multiple" Relevant only when more than one panel of graphs is possible. In that case, a new graphics device is generated each newly generated single-paneled graph. "ask" Relevant only when more than one panel of graphs is possible. In that case, each are portrayed as a single-paneled graph, with the ask=TRUE argument specified in par so that the user input confirmation is needed before the graphs are drawn.
...	Additional arguments, depending on the specific method written for the object. Currently, there is only one such specific method; see errorBarGraph.cgOneFactorFit for any additional arguments that can be specified.

Details

When `mcadjust=TRUE`, a status message of "Some time may be needed as the critical point from the `multcomp::summary.glht` function call is calculated" is displayed at the console. This computed critical point is used for all subsequent p-value and confidence interval calculations.

Value

The main purpose is the side effect of graphing to the current device. See specific methods for discussion of any return values.

Note

Contact cg@billpikounis.net for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

References

Hothorn, T., Bretz, F., Westfall, P., Heiberger, R.M., and Schuetzenmeister, A. (2010). The `multcomp` R package.

Hothorn, T., Bretz, F., and Westfall, P. (2008). "Simultaneous Inference in General Parametric Models", *Biometrical Journal*, 50, 3, 346-363.

See Also

[errorBarGraph.cgOneFactorFit](#)

Examples

```

data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(canine.data)

errorBarGraph(canine.fit)

errorBarGraph(canine.fit, mcadjust=TRUE, model="olsonly")

```

errorbargraph

Create an error bar graph based on pairwise multiple comparisons

Description

Creates a graph to see pairwise comparisons amongst groups. The method of Andrews, Sarnier, and Snee (1980) is applied to visualizes significant differences via non-overlapping error bars.

Usage

```

errorbargraph(estimates, centralvar, critpoint, endptscale="log",
              analysisname="", endptname="", alpha=0.05, digits=NULL, approxstamp=FALSE,
              titlestamp=TRUE, offset=NULL, ticklabels=NULL, ...)

```

Arguments

estimates	A named vector of estimates. Each estimate element is a measure that will be the center of the error bar of the group. The name of each group must be present in the names attribute of the vector.
centralvar	A single variance value to be used for each group's error bar construction. In the canonical case it is the <i>square of</i> the estimated standard error of the mean estimate of the group, where each group also has the same standard error (and sample size). If the standard errors / variances are similar enough across the groups, the visualization may still be effective even though the error bar lengths will be approximations.
critpoint	The single critical value of the theoretical reference distribution. In the canonical case it is the t-distribution quantile for estimates derived from a standard linear model with homoscedastic variance. It could also reflect a multiplicity adjustment, or like the centralvar discussion in the previous item, it may serve as part of a visually useful approximation for other cases.
endptscale	Must be specified as "log" or "original". If the default "log" then the y-axis will be created with a logarithmic spacing. The tick marks will be calculated accordingly and expressed in the original scale of the estimates. In other words, the estimates vector must not be already transformed.

analysisname	<i>Optional</i> , a character text or math-valid expression that will be set for default use in graph title and table methods. The default value is the empty "".
endptname	<i>Optional</i> , a character text or math-valid expression that will be set for default use as the y-axis label of graph methods, and also used for table methods. The default value is the empty "".
alpha	Significance level, by default set to 0.05, which equates to a 95% confidence level. This is just used for labelling purposes.
digits	<i>Optional</i> , for output display purposes in graphs and table methods, values will be rounded to this numeric value. Only the integers of 0, 1, 2, 3, and 4 are accepted. No rounding is done during any calculations. The default value is NULL, which will examine each individual estimates value and choose the one that has the maximum number of digits after any trailing zeroes are ignored. The max number of digits will be 4.
approxstamp	Add text to the graph that acknowledges that the error bar method is approximate.
titlestamp	Add text to the top margin above the graph area.
offset	<i>Optional</i> , if for example a numeric constant was added to all response values before calculation of the estimate as a mean, this could be used to shift the axis marks appropriately. The default value is NULL.
ticklabels	<i>Optional</i> , before graphing the data, remove any automatically generated tickmarks for the y-axis, and use these tickmarks instead. A vector of tickmarks to be placed on the y-axis. Any numeric representations will be coerced to character.
...	Additional arguments. None are currently used.

Details

The statistical method of Andrews, Sarner, and Snee (1980) is applied to visualizes significant differences via non-overlapping error bars. The method is exact when there are equal standard errors amongst the groups, and approximate otherwise. The method's usefulness declines as the standard errors become more disparate.

When two groups are compared, nonoverlapping error bars indicate a statistically significant pairwise difference. Conversely, if the error bars overlap, there is no such significant difference. In cases of approximation, or borderline overlap that is seen, the actual comparison needs to be consulted to judge significance with a p-value.

The minimum and maximum values across all the bar ends are added inside the plot region in blue, flush against the y-axis. The number of decimal places are determined by the `digits` value.

Value

`errorbargraph` returns an invisible NULL. The main purpose is the side effect of graphing to the current device.

Warning

This function was created for internal use in the `cg` package as its use can be seen in the [errorBarGraph](#) methods code. Therefore any direct use of it needs to be done cautiously.

Note

Contact <cg@billpikounis.net> for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

References

Andrews, H.P., Snee, R.D., Sarnier, M.H. (1980). "Graphical Display of Means," *The American Statistician*, 34, 195-199.

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(canine.data)

## Easier way: notice the camel case of the errorBarGraph call
errorBarGraph(canine.fit, model="olonly")

## Manual way
## Instead of errorBarGraph(canine.fit, model="olonly")
errorbargraph(estimate=canine.fit@olsfit$coef,
              centralvar=((summary(canine.fit@olsfit)$sigma^2) /
                          unique(sapply(canine, length))),
              critpoint=qt(0.975, df=canine.fit@olsfit$df.residual),
              endptscale="log",
              analysisname="Canine",
              digits=1,
              endptname=expression(paste( plain('Prostate Volume'),
                                         ' (', plain(cm)^3 , ') ' ))
              )
```

errorBarGraph.cgOneFactorFit

Create an Error Bar graph amongst groups in a cgOneFactorFit object

Description

Creates a graph to see comparisons amongst groups based on the cgOneFactorFit object. The method of Andrews, Sarnier, and Snee (1980) is applied to visualizes significant differences via non-overlapping error bars.

Usage

```
## S4 method for signature 'cgOneFactorFit'
errorBarGraph(fit, mcadjust = FALSE, alpha =0.05,
  cgtheme = TRUE, device = "single", ...)
```

Arguments

fit	A fit object of class <code>cgOneFactorFit</code> .
mcadjust	Do a multiple comparisons adjustment, based on the simultaneous inference capabilities of the multcomp package. See Details below. The default value is FALSE. If <code>mcadjust=TRUE</code> is specified, there will be a delay, usually just for a few seconds, due to computing time of the critical point in order to conduct the adjusted comparisons.
alpha	Significance level, by default set to 0.05, which equates to a 95% confidence level.
cgtheme	When set to the default TRUE, ensures a trellis device is active with limited color scheme. Namely <code>background</code> , <code>strip.shingle</code> , and <code>strip.background</code> are each set to "white".
device	Can be one of three values: "single" The default, which will put all graphs on the same device page. For example, when resistant & robust and classical least squares are present and <code>model=both</code> (the default), a 2 x 1 paneled graph will be created. "multiple" Relevant only when resistant & robust and classical least squares are present and <code>model=both</code> (the default). In that case, a new graphics device is generated to hold the resistant & robust version, as a single-paneled graph. The classical least squares version is on the previous device. "ask" Relevant only when resistant & robust and classical least squares are present and <code>model=both</code> (the default). In that case, each are portrayed as a single-paneled graph, with the <code>ask=TRUE</code> argument specified in <code>par</code> so that the user input confirmation is needed before the graphs are drawn.
...	Additional arguments. Two are currently valid:
model	For <code>cgOneFactorFit</code> objects that have classical least squares <code>lm()</code> or resistant & robust <code>r1m()</code> fits, the following argument values are possible: "both" Error Bar graphs based on both the ordinary classical least squares and resistant & robust fits are performed. This is the default when both fits are present in the <code>cgOneFactorFit</code> object specified in the <code>fit</code> argument. If the resistant & robust fit is not available, this value is not relevant. "olsonly" Only an Error Bar graph based on the ordinary classical least squares <code>olsfit</code> fit is performed. "rronly" Only a Error Bar Graph based on the resistant and robust <code>rrfit</code> fit is performed. For other possible <code>cgOneFactorFit</code> fit slots such as accelerated failure time or unequal variance models, the <code>model</code> argument is not relevant, and the appropriate comparisons table will be calculated for these model types.

ticklabels A list of two components:

- mod Can be either of these two values,
 - "replace" Before graphing the data, remove any automatically generated tickmarks for the y-axis, and create the tickmarks specified in the marks component below.
 - "add" Before graphing the data, add tickmarks specified in the marks component below, to the automatically generated ones.
- marks A vector of tickmarks to be placed on the y-axis. Any numeric representations will be coerced to character.

Details

When `mcadjust=TRUE`, a status message of "Some time may be needed as the critical point from the `multcomp::summary.glsht` function call is calculated" is displayed at the console. This computed critical point is used for all interval calculations.

The **multcomp** package provides a unified way to calculate critical points based on the comparisons of interest in a "family". Thus a user does not need to worry about choosing amongst the myriad names of multiple comparison procedures.

The `errorBarGraph.cgOneFactorFit` method is only relevant for classical least squares and resistant & robust fits in the `cgOneFactorFit` object. There is an `errorbargraph` core function that could be used for approximations in other cases like accelerated failure time or unequal variance fits.

The statistical method of Andrews, Sarner, and Snee (1980) is applied to visualize significant differences via non-overlapping error bars. The method is exact when there are equal sample sizes amongst the groups for the classical least squares case. When there are unequal group sample sizes or a resistant & robust fit is used to create the graph, the method is approximate, and this is noted in the main title section of the graph. For the unequal sample sizes, the harmonic mean is calculated to use for all the groups. The method's usefulness declines as the sample sizes become more disparate.

When two groups are compared, nonoverlapping error bars indicate a statistically significant pairwise difference. Conversely, if the error bars overlap, there is no such significant difference. In cases of approximation, or borderline overlap that is seen, the `cgOneFactorComparisonsTable` object created with `type="pairwisereflect"` or `type="pairwise"` needs to be consulted to judge significance with a p-value.

The minimum and maximum values across all the bar ends are added inside the plot region in blue, flush against the y-axis. The number of decimal places are determined by the `digits` value in the `fit$settings` slot.

If group labels along the x-axis seem to overlap in the standard horizontal form, they will be rotated 45 degrees.

Value

`errorBarGraph.cgOneFactorFit` returns an invisible `NULL`. The main purpose is the side effect of graphing to the current device.

Note

Contact cg@billpikounis.net for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

References

- Andrews, H.P., Snee, R.D., Sarnier, M.H. (1980). "Graphical Display of Means," *The American Statistician*, 34, 195-199.
- Hothorn, T., Bretz, F., Westfall, P., Heiberger, R.M., and Schuetzenmeister, A. (2010). The multcomp R package.
- Hothorn, T., Bretz, F., and Westfall, P. (2008). "Simultaneous Inference in General Parametric Models", *Biometrical Journal*, 50, 3, 346-363.

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(canine.data)

errorBarGraph(canine.fit)

errorBarGraph(canine.fit, mcadjust=TRUE, model="olsonly")
```

 fit

Fit models to data

Description

Fit data objects prepared by the **cg** package.

Usage

```
fit(data, type, ...)
```

Arguments

- | | |
|------|---|
| data | A data object prepared with a prepare call. This will involve a data frame and additional settings. The only class of object currently available is cgOneFactorData , which is prepared by the prepareCGOneFactorData method. |
| type | Type of model to fit, represented by a character string. |
| ... | Additional arguments, depending on the specific method written for the object. Currently, there is only one such specific method; see fit.cgOneFactorData . |

Value

A method-specific fit object is returned. See the specific methods for discussion of return values.

Note

Contact <cg@billpikounis.net> for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

See Also

[fit.cgOneFactorData](#)

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")
canine.fit <- fit(canine.data, type="rr")
```

```
data(gmcsfcens)
gmcsfcens.data <- prepareCGOneFactorData(gmcsfcens, format="groupcolumns",
                                         analysisname="cytokine",
                                         endptname="GM-CSF (pg/ml)",
                                         logscale=TRUE)
gmcsfcens.fit <- fit(gmcsfcens.data, type="aft")
```

fit.cgOneFactorData *Fit models to a cgOneFactorData object*

Description

Fits a one-factor model based on the cgOneFactorData object. The created object is designed for one-factor / one-way / unpaired samples collected data, and is of class [cgOneFactorFit](#).

Usage

```
## S4 method for signature 'cgOneFactorData'
fit(data, type="rr", ...)
```

Arguments

data	A data object of class <code>cgOneFactorData</code> .
type	Type of model to fit, represented by a character value. The default value is "rr". The four current possibilities are: <ul style="list-style-type: none"> "ols" Only fit an ordinary, classical least squares model with the base <code>lm</code> linear model function. "rr" Fit a Resistant / Robust model based on M- and S-estimation, using the <code>rlm</code> function from the MASS package (Venables and Ripley, 2002). This is the default character value for the <code>type</code> argument. See the ... argument and Details below for what options are available when <code>rlm</code> is used in this wrapper method. "aft" Fit an accelerated failure time model, using the <code>survreg</code> function from the survival package. If the data object has censored data and a slot state of <code>has.censored=TRUE</code>, then <code>type="aft"</code> will be set. See the ... argument and Details below for what options are used when <code>survreg</code> is used in this wrapper method; in particular, the robust argument. "uv" Fit an unequal variances model, with a simple wrapper method around the <code>gls</code> from the nlme package. No optional arguments are passed down to <code>gls</code> through the ... argument.
...	Additional arguments, both <i>optional</i> , that are allowed to be specified dependent on the choice of the <code>type</code> argument. Otherwise they have no effect on the fit: <ul style="list-style-type: none"> <code>maxIter</code> If <code>type="rr"</code> or <code>type="aft"</code>, then <code>maxIter</code> can be specified as a numeric positive integer. The default value of <code>maxIter</code> is 100. For "rr", this gets passed to the <code>maxit</code> argument in the <code>rlm</code> method. For "aft", this gets passed to the <code>maxiter</code> argument in the <code>survreg</code> function. <code>sandaft</code> If <code>type="aft"</code>, then <code>sandaft</code> is passed to the <code>robust</code> argument of the <code>survreg</code> function. The default value of <code>sandaft</code> is <code>TRUE</code> when <code>type="aft"</code>, which applies the Huber-type (1967) sandwich estimator to the variance-covariance matrix of the group estimates.

Details

In the current version of the **cg** package, most default settings for `rlm` are kept for the `fit.cgOneFactorData` method wrapper call when `type="rr"`, with no capability to choose another value for an arguments such as `psi`, `scale.est`, and `k2`. The `method` argument is set to "MM".

Analogously most `survreg` default settings are kept for the `fit.cgOneFactorData` method wrapper call when `type="aft"`, with no capability to modify the arguments. Most notably the `dist` argument is set to "lognormal" or "gaussian", depending on whether a log scale analysis request is evident in the `cgOneFactorData` object or not, respectively.

Value

Creates an object of class `cgOneFactorFit`, with the following slots:

`olsfit` The contents of a `lm` fit to the data. This is always populated with an `lm` object no matter the choice of the `type` argument, even though it is certainly inappropriate in the `type="aft"` case.

- `rrfit` The contents of a `rlm` fit to the data, housed as a `rrfit` class object. If `type="rr"` is not selected, then this is set to a simple character value of "No fit was selected."
- `aftfit` The contents of a `survreg` fit to the data, with some annotations, to be a `aftfit` class object. If `type="aft"` is not selected, then this is set to a simple character value of "No fit was selected."
- `uvfit` The contents of a `gls` fit to the data, housed as a `uvfit` class object. If `type="uv"` is not selected, then this is set to a simple character value of "No fit was selected."
- `settings` A list of properties carried as-is from the data argument object of class `cgOneFactorData`. In particular, if `zeroscore` is specified as a non-NULL number in the `cgOneFactorData` object in the data argument, then a score value near zero was derived to replace all zeroes for subsequent log-scale analyses. Alternatively, if `addconstant` is specified as a non-NULL number in the `cgOneFactorData` object in the data argument, then a value was added to shift up all observations for subsequent log-scale analyses.

Note

Contact <cg@billpikounis.net> for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

References

- Huber, P.J (1967), "The Behavior of Maximum Likelihood Estimates Under Nonstandard Conditions", *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, Volume 1, 221-233.
- Venables, W. N. and Ripley, B. D. (2002), *Modern Applied Statistics with S*. Fourth edition. Springer.

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(data=canine.data, type="rr")

data(gmcsfcens)
gmcsfcens.data <- prepareCGOneFactorData(gmcsfcens, format="groupcolumns",
                                         analysisname="cytokine",
                                         endptname="GM-CSF (pg/mL)",
                                         logscale=TRUE)

gmcsfcens.fit <- fit(gmcsfcens.data, type="aft")
```

globalTest	<i>Perform a global test of significance</i>
------------	--

Description

Generic function to perform a global test of significance on a fit by the **cg** package.

Usage

```
globalTest(fit, display="print", ...)
```

Arguments

fit	A fit object created by a <code>fit</code> method from the cg package. The only class of object currently available is <code>cgOneFactorFit</code> , which is prepared by the <code>fit.cgOneFactorData</code> method.
display	One of three valid values: "print" The default value, it calls a <code>print</code> method for the created <code>globalTest</code> object, which is formatted text output of the test p-values. "none" Suppresses any printing. Useful, for example, when just assignment of the resulting object is desired. "show" Calls the default <code>showDefault</code> method, which will just print out the <code>globalTest</code> components.
...	Additional arguments, depending on the specific method written for the object. See the method-specific documentation for additional details.

Value

A method-specific `globalTest` object is returned. See the specific methods for discussion of return values.

Note

Contact <cg@billpikounis.net> for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

See Also

[globalTest.cgOneFactorFit](#)

Examples

```

data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(canine.data)

canine.globalTest <- globalTest(canine.fit)

data(gmcsfcens)
gmcsfcens.data <- prepareCGOneFactorData(gmcsfcens, format="groupcolumns",
                                         analysisname="cytokine",
                                         endptname="GM-CSF (pg/ml)",
                                         logscale=TRUE)

gmcsfcens.fit <- fit(gmcsfcens.data, type="aft")

gmcsfcens.globalTest <- globalTest(gmcsfcens.fit)

```

```
globalTest.cgOneFactorFit
```

Perform a global Test of significance with cgOneFactorFit object

Description

Performs a global test based on the `cgOneFactorFit` object, to assess whether there are any significant differences amongst levels of the factor, i.e. amongst the groups. A `cgOneFactorGlobalTest` class object is created.

Usage

```

## S4 method for signature 'cgOneFactorFit'
globalTest(fit, display="print", ...)

```

Arguments

<code>fit</code>	A fit object of class <code>cgOneFactorFit</code> .
<code>display</code>	One of three valid values: <ul style="list-style-type: none"> "print" The default value, it calls a print method for the created <code>globalTest.cgOneFactorFit</code> object, which is formatted text output of the test p-values. "none" Suppresses any printing. Useful, for example, when just assignment of the resulting object is desired.

"show" Calls the default `showDefault` method, which will just print out the `globalTest.cgOneFactorFit` components.

...

Additional arguments. Only one is currently valid:

`model` For `cgOneFactorFit` objects that have classical least squares `lm()` or resistant & robust `r1m()` fits, the following argument values are possible:

"both" Global tests on both the ordinary classical least squares and resistant robust fits are performed. This is the default when both fits are present in the `cgOneFactorFit` object specified in the `fit` argument.

"olsonly" Only a global test on the ordinary classical least squares `olsfit` fit is performed.

"rronly" Only a global test on the resistant and robust `rrfit` fit is performed.

For other possible `cgOneFactorFit` fit components such as accelerated failure time or unequal variance models, the `model` argument is not relevant, and the appropriate global test will be detected and performed for these model types.

Details

The notion of a global F test, or equivalently, of R^2 , for resistant & robust linear models is murky, as no clear theoretical analogue to the ordinary classical least squares approach exists. The approach taken here is ad-hoc, which is essentially to re-fit a linear model with `lm()` and weights from the resistant & robust fit. This ad-hoc approach is taken when there are 3 or more groups.

If there are only 2 groups, then the `comparisonsTable.cgOneFactorFit` method is used with the `r1m()` model component.

Value

Creates an object of class `cgOneFactorGlobalTest`, with the following slots:

`ols.gpval` The p-value of a global F test applied to the `olsfit` component of the `cgOneFactorFit` object, unless `model="rronly"` is specified. Will not be appropriate in the case where a valid `aftfit` component is present in the `cgOneFactorFit` object.

`rr.gpval` The p-value of an ad-hoc global test applied to the `rrfit` component of the `cgOneFactorFit` object, if a valid resistant & robust fit object is present. See the Details section above. If `rrfit` is a simple character value of "No fit was selected.", or `model="olsonly"` was specified, then the value is NULL.

`aft.gpval` The p-value of a global chi-square test applied to the `aftfit` component of the `cgOneFactorFit` object if a valid accelerated failure time fit object is present. If `aftfit` is a simple character value of "No fit was selected.", then the value is NULL.

`uv.gpval` The p-value of a global F test applied to the `uvfit` component of the `cgOneFactorFit` object if a valid unequal variances fit object is present.

Note

Contact <cg@billpikounis.net> for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(canine.data)

canine.globalTest <- globalTest(canine.fit)

globalTest(canine.fit, model="both")

globalTest(canine.fit, model="olonly")

globalTest(canine.fit, model="rronly")

data(gmcsfcens)
gmcsfcens.data <- prepareCGOneFactorData(gmcsfcens, format="groupcolumns",
                                         analysisname="cytokine",
                                         endptname="GM-CSF (pg/ml)",
                                         logscale=TRUE)

gmcsfcens.fit <- fit(gmcsfcens.data, type="aft")

globalTest(gmcsfcens.fit)
```

gmcsfcens

GM-CSF censored data set in the cg package

Description

A data frame used to illustrate the **cg** package. It has a One Factor / One-Way / Unpaired Samples layout. It also contains left-censored values of varying degree in each of the six groups.

Usage

```
data(gmcsfcens)
```

Format

A 8-by-6 data frame with up to 8 numeric observations per group from an experiment on the following 6 groups.

PBS/Tg 197 phosphate buffered saline control group

1mg/kg/Tg 197 1 mg/kg dose

3mg/kg/Tg 197 3 mg/kg dose

10/mg/kg/Tg 197 10 mg/kg dose

30/mg/kg/Tg 197 30 mg/kg dose

PBS/WT phosphate buffered saline control group of wild-type mice

The first five groups have transgenic (Tg197) mice subjects, a well established model to induce arthritis. The sixth group are "wild-type" mice that did not have arthritis induced. The various doses of the inner four groups are administrations of golimumab, a monoclonal antibody therapy.

The individual group values are of mode character, since some of them are represented as left-censored values such as <82.5 . Note that two of the groups have less than 8 observations, and the corresponding cells in the data frame actually contain empty quote "" values.

Details

The gmcsfcens data set that comes with the **cg** package is in `groupcolumns` format. Each column represents a group, and the observations in that group's column are the individual response values. As described above, they are character valued potentially left-censored representations.

The 6 groups are regarded as levels of one factor in the `prepareCGOneFactorData`, `fit`, and other methods in the **cg** package.

Alternative formats of this data set is contained in `gmcsfcens.listfmt`. See that help file for details, including how such formats would be read and prepared by **cg**

GM-CSF stands for Granulocyte macrophage colony stimulating factor, a type of cytokine that is important in the growth of white blood cells. It is one of the outcomes measured in the experiment described in the references section below. Therapeutic inhibition of it may be beneficial in cases where too many white blood cells are produced, such as arthritis. In other situations where white blood cell counts are low, stimulation of it is desired. In the referenced study below, GM-CSF is evaluated in the context of inflammation.

Note

Contact `<cg@billpikounis.net>` for bug reports, questions, concerns, and comments.

References

Shealy, D., Cai, A., Staquet, K., Baker, A., Lacy, E., Johns, L., Vafa, O., Gunn, G., Tam, S., Sague, S., Wang, D., Brigham-Burke, M., Dalmonte, P., Emmell, E., Pikounis, B., Bugelski, P., Zhou, H., Scallon, B., Giles-Komar, J. (2010). "Characterization of Golimumab (CNTO148), a human monoclonal antibody specific for human tumor necrosis factor ", *mAbs*, Volume 2, Issue 4, 428-439.

See Also

[gmcsfcens.listfmt](#), [prepareCGOneFactorData](#)

Examples

```
data(gmcsfcens)
str(gmcsfcens)
```

gmcsfcens.listfmt	<i>GM-CSF censored data set in the cg package</i>
-------------------	---

Description

A data frame used to illustrate the **cg** package. It has a One Factor / One-Way / Unpaired Samples layout. It also contains left-censored values of varying degree in each of the six groups. There are three equivalent data frame versions documented here.

Usage

```
data(gmcsfcens.listfmt1)
  data(gmcsfcens.listfmt2)
  data(gmcsfcens.listfmt3)
```

Format

A 45 row data frame with up to 8 observations per group from an experiment on the following 6 groups.

```
PBS/Tg 197  phosphate buffered saline control group
1mg/kg/Tg 197  1 mg/kg dose
3mg/kg/Tg 197  3 mg/kg dose
10/mg/kg/Tg 197  10 mg/kg dose
30/mg/kg/Tg 197  30 mg/kg dose
PBS/WT  phosphate buffered saline control group of wild-type mice
```

The first five groups have transgenic (Tg197) mice subjects, a well established model to induce arthritis. The sixth group are "wild-type" mice that did not have arthritis induced. The various doses of the inner four groups are administrations of golimumab, a monoclonal antibody therapy.

There can be either 2, 3, or 4 columns in the data frame. The above 6 items are the levels of the first column's factor, named grp.

2 columns The data frame name is `gmcsfcens.listfmt1`. The second column `endpt` contains the character observations that can represent complete observations, and also left- or right-censored ones, in the same way that `gmcsfcens` does.

- 3 columns The data frame name is `gmcsfcens.listfmt2`. The second column `endpt` contains numeric observations, and the third column `status` indicates whether the observation is complete/not censored (1), and 0 if left-censored. See [prepareCGOneFactorData](#) for the explanation of why the value of 0 and not 2 is required. In the example code below, the `leftcensor=TRUE` argument needs to be specified when this format version is used.
- 4 columns The data frame name is `gmcsfcens.listfmt3`. The second and third columns `endpt1` and `endpt2` contain numeric observations, and the fourth column `status` indicates whether the observation is complete/not censored (1), and 2 if left-censored. See [prepareCGOneFactorData](#) for the explanation of this format, and the example code below.

Details

The `gmcsfcens.listfmt*` data sets that comes with the **cg** package are in listed format.

The 6 groups are regarded as levels of one factor in the [prepareCGOneFactorData](#), `fit`, and other methods in the **cg** package.

The `gmcsfcens.listfmt` data sets are alternative formats of the `gmcsfcens` data set. See that help file for details. Once a `gmcsfcens.listfmt` data set is [prepared](#) into a `cgOneFactorData` object, all the subsequent methods work on the object in the same way.

GM-CSF stands for Granulocyte macrophage colony stimulating factor, a type of cytokine that is important in the growth of white blood cells. It is one of the outcomes measured in the experiment described in the references section below. Therapeutic inhibition of it may be beneficial in cases where too many white blood cells are produced, such as arthritis. In other situations where white blood cell counts are low, stimulation of it is desired. In the referenced study below, GM-CSF is evaluated in the context of inflammation.

Note

Contact cg@billpikounis.net for bug reports, questions, concerns, and comments.

References

Shealy, D., Cai, A., Staquet, K., Baker, A., Lacy, E., Johns, L., Vafa, O., Gunn, G., Tam, S., Sague, S., Wang, D., Brigham-Burke, M., Dalmonte, P., Emmell, E., Pikounis, B., Bugelski, P., Zhou, H., Scallon, B., Giles-Komar, J. (2010). "Characterization of Golimumab (CNT0148), a human monoclonal antibody specific for human tumor necrosis factor ", *mAbs*, Volume 2, Issue 4, 428-439.

See Also

[gmcsfcens](#), [prepareCGOneFactorData](#)

Examples

```
data(gmcsfcens.listfmt1)
str(gmcsfcens.listfmt1)
```

```
data(gmcsfcens.listfmt2)
str(gmcsfcens.listfmt2)
```

```

data(gmcsfcens.listfmt3)
str(gmcsfcens.listfmt3)

## Analogous to prepareCGOneFactorData call on gmcsfcens data frame format,
## subsequent methods will work for gmcsfcens.listfmt.data objects below:

## leftcensor argument can be left as default NULL
gmcsfcens.listfmt1.data <- prepareCGOneFactorData(gmcsfcens.listfmt1, format="listed",
                                                analysisname="cytokine",
                                                endptname="GM-CSF (pg/ml)",
                                                logscale=TRUE)

## leftcensor=TRUE argument needs to be set
gmcsfcens.listfmt2.data <- prepareCGOneFactorData(gmcsfcens.listfmt2, format="listed",
                                                analysisname="cytokine",
                                                endptname="GM-CSF (pg/ml)",
                                                logscale=TRUE,
                                                leftcensor=TRUE)

## leftcensor argument can be left as default NULL
gmcsfcens.listfmt3.data <- prepareCGOneFactorData(gmcsfcens.listfmt3, format="listed",
                                                analysisname="cytokine",
                                                endptname="GM-CSF (pg/ml)",
                                                logscale=TRUE)

## as they do on gmcsfcens.data:

gmcsfcens.data <- prepareCGOneFactorData(gmcsfcens, format="groupcolumns",
                                        analysisname="cytokine",
                                        endptname="GM-CSF (pg/ml)",
                                        logscale=TRUE)

```

grpSummaryTable *Create a table of estimated group means and variability*

Description

Create a table of estimated group means based on a fit by the **cg** package.

Usage

```
grpSummaryTable(fit, mcadjust = FALSE, alpha = 0.05, display = "print", ...)
```

Arguments

fit An fit object created with a **fit** method from the **cg** package. The only class of object currently available is **cgOneFactorFit**, which is prepared by the **fit.cgOneFactorData** method.

mcadjust	Do a multiple comparisons adjustment, based on the simultaneous inference capabilities of the multcomp package. See Details below. The default value is FALSE. If mcadjust=TRUE is specified, there will be a delay, usually just for a few seconds, due to computing time of the critical point in order to conduct the adjusted comparisons.
alpha	Significance level, by default set to 0.05.
display	One of three valid values: "print" The default value, it calls a print method for the created cgOneFactorGrpSummaryTable object, which is a formatted text output of the table(s). "none" Suppresses any printing. Useful, for example, when just assignment of the resulting object is desired. "show" Calls the default showDefault method, which will just print out the grpSummaryTable object components.
...	Additional arguments, depending on the specific method written for the object. Currently, there is only one such specific method; see grpSummaryTable.cgOneFactorFit for any additional arguments that can be specified.

Details

When mcadjust=TRUE, a status message of "Some time may be needed as the critical point from the multcomp::summary.glt function call is calculated" is displayed at the console. This computed critical point is used for all subsequent p-value and confidence interval calculations.

Value

A method-specific grpSummaryTable object is returned. See the specific methods for discussion of return values.

Note

Contact <cg@billpikounis.net> for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

References

- Hothorn, T., Bretz, F., Westfall, P., Heiberger, R.M., and Schuetzenmeister, A. (2010). The multcomp package.
- Hothorn, T., Bretz, F., and Westfall, P. (2008). "Simultaneous Inference in General Parametric Models", *Biometrical Journal*, 50, 3, 346-363.

See Also

[grpSummaryTable.cgOneFactorFit](#)

Examples

```

data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(canine.data)

canine.grpsumm0 <- grpSummaryTable(canine.fit)

data(gmcsfcens)
gmcsfcens.data <- prepareCGOneFactorData(gmcsfcens, format="groupcolumns",
                                         analysisname="cytokine",
                                         endptname="GM-CSF (pg/mL)",
                                         logscale=TRUE)

gmcsfcens.fit <- fit(gmcsfcens.data, type="aft")

gmcsfcens.grpsumm0 <- grpSummaryTable(gmcsfcens.fit)

```

```
grpSummaryTable.cgOneFactorFit
```

Create a table of estimated group means and variability with a cgOneFactorFit object.

Description

Create a table of estimated group means based on the cgOneFactorFit object. Standard errors and confidence intervals are added. A cgOneFactorGrpSummaryTable class object is created.

Usage

```
## S4 method for signature 'cgOneFactorFit'
grpSummaryTable(fit, mcadjust=FALSE, alpha=0.05, display="print", ...)
```

Arguments

fit	A fit object of class <code>cgOneFactorFit</code> .
mcadjust	Do a multiple comparisons adjustment, based on the simultaneous inference capabilities of the multcomp package. See Details below. The default value is FALSE. If mcadjust=TRUE is specified, there will be a delay, usually just for a few seconds, due to computing time of the critical point in order to calculate the confidence intervals.
alpha	Significance level, by default set to 0.05.

`display` One of three valid values:

- "print" The default value, it calls a print method for the created `cgOneFactorGrpSummaryTable` object, which is a formatted text output of the table(s).
- "none" Suppresses any printing. Useful, for example, when just assignment of the resulting object is desired.
- "show" Calls the default `showDefault` method, which will just print out the `cgOneFactorGrpSummaryTable` components.

...

`model` Additional arguments. Only one is currently valid:

- For `cgOneFactorFit` fit objects that have classical least squares `lm()` or resistant & robust `rlm()` fits, the following argument values are possible:
 - "both" Group summary tables based on both the ordinary classical least squares and resistant & robust fits are performed. This is the default when both fits are present in the `cgOneFactorFit` object specified in the `fit` argument. If the resistant & robust fit is not available, this value is not relevant.
 - "olsonly" Only a group summary table based on the ordinary classical least squares `olsfit` fit is performed.
 - "rronly" Only a group summary table based on the resistant and robust `rrfit` fit is performed.

For other possible `cgOneFactorFit` fit components such as accelerated failure time or unequal variance models, the `model` argument is not relevant, and the appropriate group summary table will be calculated for these model types.

Details

When `mcadjust=TRUE`, a status message of "Some time may be needed as the critical point from the `multcomp::summary.glsht` function call is calculated" is displayed at the console. This computed critical point is used for all subsequent confidence interval calculations.

The **multcomp** package provides a unified way to calculate critical points based on the comparisons of interest in a "family". Thus a user does not need to worry about choosing amongst the myriad names of multiple comparison procedures.

Value

Creates an object of class `cgOneFactorGrpSummaryTable`, with the following slots:

`ols.grps` The table of group estimates based on the `olsfit` component of the `cgOneFactorFit`, unless `model="rronly"` is specified. In that case the slot value is `NULL`. Will not be appropriate in the case where a valid `aftfit` component is present in the `cgOneFactorFit` object. See below for the data frame structure of the table.

`rr.grps` The table of group estimates based on the `rrfit` component of the `cgOneFactorFit` object, if a valid resistant & robust fit object is present. If `rrfit` is a simple character value of "No fit was selected.", or `model="olsonly"` was specified, then the value is `NULL`. See below for the data frame structure of the table.


```

                                endptname="Prostate Volume",
                                endptunits=expression(plain(cm)^3),
                                digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(canine.data)

grpSummaryTable(canine.fit)

grpSummaryTable(canine.fit, mcadjust=TRUE, model="olonly")

data(gmcsfcens)
gmcsfcens.data <- prepareCGOneFactorData(gmcsfcens, format="groupcolumns",
                                         analysisname="cytokine",
                                         endptname="GM-CSF (pg/ml)",
                                         logscale=TRUE)

gmcsfcens.fit <- fit(gmcsfcens.data, type="aft")

grpSummaryTable(gmcsfcens.fit)

```

kmGraph

Graph Distribution Functions of Groups

Description

Create non-parametric survival or cumulative distribution graphs based on a data object in the **cg** package.

Usage

```
kmGraph(data, cgtheme = TRUE, distfcn = "survival", ylab = NULL,
        title = NULL, ...)
```

Arguments

data	A data object created using the cg package. The only class of object currently available is cgOneFactorData , which is created by the prepareCGOneFactorData function.
cgtheme	When set to the default TRUE, ensures a trellis device is active with limited color scheme, namely background, strip.shingle, and strip.background are each set to "white".
distfcn	A character, which may be either "survival" to graph the survival function, or "cumulative" to graph the cumulative distribution function.
ylab	Specify a character value for the y-axis label. The default value is NULL.
title	Specify a character value for the main title at the top of the graph. The default value is NULL.

... Additional arguments, depending on the specific method written for the object. Currently, there is only one such specific method; see [kmGraph.cgOneFactorData](#) for any additional arguments that can be specified.

Details

Color assignments of the graphed step functions lines for the groups match the order of the group name factor levels. The color order is given in [cgLineColors](#). The line widths are set to be thicker (`lwd=2`), and the group name label is placed near the line using `label` methodology for the **Hmisc** package.

The x-axis represents response values, and y-axis represents estimated probabilities. Minimum and maximum values from ranges of data are respectively labeled in the bottom left and right corners of graph regions.

Value

The main purpose is the side effect of graphing to the current device. See the specific methods for discussion of any return values.

Note

Contact cg@billpikounis.net for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

See Also

[kmGraph.cgOneFactorData](#)

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

kmGraph(canine.data, distfcn="cumulative")
kmGraph(canine.data, distfcn="cumulative",
        ticklabels=list(mod="add", marks=c(2)))

## Censored Data
data(gmcsfcens)
gmcsfcens.data <- prepareCGOneFactorData(gmcsfcens, format="groupcolumns",
                                         analysisname="cytokine",
                                         endptname="GM-CSF (pg/ml)",
                                         logscale=TRUE)

kmGraph(gmcsfcens.data, distfcn="cumulative")
kmGraph(gmcsfcens.data, distfcn="cumulative", logscale=FALSE)
```

 kmGraph.cgOneFactorData

Graph Distribution Functions of Groups in a cgOneFactorData object

Description

Create a non-parametric survival or cumulative distribution graph of groups of data in a cgOneFactorData object.

Usage

```
## S4 method for signature 'cgOneFactorData'
kmGraph(data, cgtheme = TRUE, distfcn = "survival",
        ylab = NULL, title = NULL, ...)
```

Arguments

data	A cgOneFactorData object.
cgtheme	A logical value, indicating whether to use the cg limited color scheme (white backgrounds, including shingles) or the current Trellis color scheme. The default is TRUE.
distfcn	A character value, which may be either the default "survival" to graph the survival function, or "cumulative" to graph the cumulative distribution function.
ylab	<i>Optional</i> , a character value to label the y-axis of the graph. If NULL, it defaults to either "Probability of Survival" or "Cumulative Probability", depending on the value of distfcn.
title	<i>Optional</i> , a character to show as the title of the graph. If NULL, it defaults to either "Nonparametric Survival Curve Estimates" or "Nonparametric Cumulative Distribution Function Estimates", depending on the value of distfcn.
...	Additional arguments. One is currently valid:
ticklabels	A list of two components: <ul style="list-style-type: none"> mod Can be either of these two values, <ul style="list-style-type: none"> "replace" Before graphing the data, remove any automatically generated tickmarks for the x-axis, and create the tickmarks specified in the marks component below. "add" Before graphing the data, add tickmarks specified in the marks component below, to the automatically generated ones. marks A vector of tickmarks to be placed on the x-axis. Any numeric representations will be coerced to character.

Details

Graph the estimated survival function or cumulative distribution for each group in a `cgOneFactorData` object. For censored data, Kaplan-Meier estimates are used. For uncensored data, the conventional step function empirical estimates are used.

Color assignments of the graphed step functions lines for the groups match the order of the group name factor levels. The color order is given in `cgLineColors`. The line widths are set to be thicker (`lwd=2`), and the group name label is placed near the line using `label` methodology for the **Hmisc** package.

The x-axis represents response values, and y-axis represents estimated probabilities. Minimum and maximum values from ranges of data are respectively labeled in the bottom left and right corners of graph regions.

The label for the x-axis is taken from the `cgOneFactorData` object, which `prepareCGOneFactorData` sets from its `endptname` and `endptunits` arguments.

The number of decimal places printed in the ticks on the x-axis is taken from the `cgOneFactorData` object, which `prepareCGOneFactorData` sets from its `digits` argument.

Value

`kmGraph.cgOneFactorFit` returns an invisible NULL. The main purpose is the side effect of graphing to the current device.

Note

Contact <cg@billpikounis.net> for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

kmGraph(canine.data, distfcn="cumulative")
kmGraph(canine.data, distfcn="cumulative",
        ticklabels=list(mod="add", marks=c(2)))

## Censored Data
data(gmcsfcens)
gmcsfcens.data <- prepareCGOneFactorData(gmcsfcens, format="groupcolumns",
                                         analysisname="cytokine",
                                         endptname="GM-CSF (pg/ml)",
                                         logscale=TRUE)

kmGraph(gmcsfcens.data, distfcn="cumulative")
kmGraph(gmcsfcens.data, distfcn="cumulative", logscale=FALSE)
```

pointGraph

Graph Individual Data Points of Groups

Description

Generic function to create point graphs (a.k.a. dot plot, strip plot, one-dimensional scatter plot) of a data object created by the **cg** package.

Usage

```
pointGraph(data, ...)
```

Arguments

data	A data object created with a prepare function or method from the cg package. The only class of object currently valid is cgOneFactorData , which is created by the prepareCGOneFactorData function.
...	Additional arguments, depending on the specific method written for the object. Currently, there is only one such specific method; see pointGraph.cgOneFactorData for any additional arguments that can be specified.

Details

Individual points are [jittered](#), and open circles are used to alleviate potential overlap and the danger of representing multiple points as a single point.

The point graph is a vertical dot plot or strip plot, with separate areas for each group in the data set, and light gray lines between the groups. For censored data, left-censored values are shown as a shallow "V", which is actually just a rotated downward "<" sign. Similarly, right-censored values are shown as a deeper "^", which is a rotated upward ">" sign.

Minimum and maximum values from ranges of data are respectively labeled in the bottom and top left corners of graph regions.

Tick marks are attempted to be chosen wisely. For log-scaled axes in particular, leading digits of 2, 5, and 10 for values are included if possible. Since the algorithm is empirical, the `ticklabels` argument is available for further refinement or complete replacement of tickmarks.

Value

The main purpose is the side effect of graphing to the current device. See the specific methods for discussion of any return values.

Note

Contact cg@billpikounis.net for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

See Also

[pointGraph.cgOneFactorData](#)

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

pointGraph(canine.data)

# Graph the data on the original scale instead of the log scale.
pointGraph(canine.data, logscale=FALSE)

## Censored Data
data(gmcsfcens)
gmcsfcens.data <- prepareCGOneFactorData(gmcsfcens, format="groupcolumns",
                                         analysisname="cytokine",
                                         endptname="GM-CSF (pg/ml)",
                                         logscale=TRUE)

pointGraph(gmcsfcens.data)
```

```
pointGraph.cgOneFactorData
```

Graph Individual Data Points in a cgOneFactorData object

Description

Create a point graph (a.k.a. dot plot, strip plot, one-dimensional scatter plot) of the data in a `cgOneFactorData` object.

Usage

```
## S4 method for signature 'cgOneFactorData'
pointGraph(data, ...)
```

Arguments

<code>data</code>	A <code>cgOneFactorData</code> object.
<code>...</code>	Additional arguments, both <i>optional</i> . Two are currently valid:
<code>logscale</code>	A logical value, indicating whether or not the point graph should be plotted on the logarithmic scale. If <code>logscale</code> is not specified, its value is taken from the <code>cgOneFactorData</code> object, which prepareCGOneFactorData sets from its <code>logscale</code> argument.
<code>ticklabels</code>	A list of two components:

`mod` Can be either of these two values,
 "replace" Before graphing the data, remove any automatically generated tickmarks for the y-axis, and create the tickmarks specified in the marks component below.
 "add" Before graphing the data, add tickmarks specified in the marks component below, to the automatically generated ones.
`marks` A vector of tickmarks to be placed on the y-axis. Any numeric representations will be coerced to character.

Details

If `logscale=TRUE`, the tick marks for the y-axis on the left side of the plot show original values, while the tick marks for the y-axis on the right side of the graph show base 10 log values.

Tick marks are attempted to be chosen wisely. For log-scaled axes in particular, leading digits of 2, 5, and 10 for values are included if possible. Since the algorithm is empirical, the `ticklabels` argument is available for further refinement or complete replacement of tickmarks.

Individual points are *jittered*, and open circles are used to alleviate potential overlap and the danger of representing multiple points as a single point.

The point graph is a vertical dot plot or strip plot, with separate areas for each group in the data set, and light gray lines between the groups. For censored data, left-censored values are shown as a shallow "V", which is actually just a rotated downward "<" sign. Similarly, right-censored values are shown as a deeper "^", which is a rotated upward ">" sign.

The heading for the graph is taken from the `cgOneFactorData` object, which [prepareCGOneFactorData](#) sets from its `analysisname` argument.

The label for the y-axis is taken from the `cgOneFactorData` object, which [prepareCGOneFactorData](#) sets from its `endptname` and `endptunit` arguments.

The number of decimal places printed in the ticks on the y-axis is taken from the `cgOneFactorData` object, which [prepareCGOneFactorData](#) sets from its `digits` argument.

The minimum and maximum values from the range of the data are respectively labeled in the bottom and top left corners of the graph region.

If group labels along the x-axis seem to overlap in the standard horizontal form, they will be rotated 45 degrees.

Value

`pointGraph.cgOneFactorData` returns an invisible NULL. The main purpose is the side effect of graphing to the current device.

Note

Contact cg@billpikounis.net for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

Examples

```

data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(canine.data)
pointGraph(canine.data)

# Graph the data on the original scale instead of the log scale.
pointGraph(canine.data, logscale=FALSE)

## Censored Data
data(gmcsfcens)
gmcsfcens.data <- prepareCGOneFactorData(gmcsfcens, format="groupcolumns",
                                         analysisname="cytokine",
                                         endptname="GM-CSF (pg/ml)",
                                         logscale=TRUE)

pointGraph(gmcsfcens.data)

```

```
prepare
```

Prepare a cg data object from a data frame

Description

Reads in a data frame and settings in order to create a cg Data object.

Usage

```
prepare(type, ...)
```

Arguments

<code>type</code>	Currently only accepts the value of "onefactor" or its synonym "unpairedgroups" to create a cgOneFactorData object.
<code>...</code>	Depends on the specific function that is called according to the <code>type</code> argument. Currently the prepareCGOneFactorData is the sole valid call, and no ... arguments are used.

Value

See [cgOneFactorData](#) for the only possible valid object currently returned, assuming that the `type` and `...` arguments are correctly specified.

Note

Contact <cg@billpikounis.net> for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

See Also

[prepareCGOneFactorData](#)

Examples

```
data(canine)
canine.data <- prepare(type="unpairedgroups", dfr=canine,
                      format="groupcolumns",
                      analysisname="Canine",
                      endptname="Prostate Volume",
                      endptunits=expression(plain(cm)^3),
                      digits=1, logscale=TRUE, refgrp="CC")

## Censored Data
data(gmcsfcens)
gmcsfcens.data <- prepare(type="onefactor",
                         dfr=gmcsfcens, format="groupcolumns",
                         analysisname="cytokine",
                         endptname="GM-CSF (pg/ml)",
                         logscale=TRUE)
```

prepareCGOneFactorData

Prepare data object from a data frame for One Factor / One-Way / Unpaired Samples evaluations

Description

The function `prepareCGOneFactorData` reads in a data frame and settings in order to create a `cgOneFactorData` object. The created object is designed to have exploratory and fit methods applied to it.

Usage

```
prepareCGOneFactorData(dfr, format = "listed", analysisname = "",
                       endptname = "", endptunits = "", logscale = TRUE, zeroscore = NULL,
                       addconstant = NULL, rightcensor = NULL, leftcensor = NULL, digits = NULL,
                       refgrp = NULL, stamps = FALSE)
```

Arguments

dfr	A valid data frame, see the format argument.
format	<p>Default value of "listed". Either "listed" or "groupcolumns" must be used. Abbreviations of "l" or "g", respectively, or otherwise sufficient matching values can be used:</p> <p>"listed" At least two columns, with the factor levels in the first column and response values in the second column. If there is censored data, then two or three more columns are required, see the Details Input Data Frame section below.</p> <p>"groupcolumns" Each column must represent a group. Each group is a unique level of the one factor, so the levels of the factor make up the column headers. The values in the data frame are for the response. If the groups have unequal sample sizes, the empty cells within the data.frame can have NA's or be left blank. Censored values can be represented; see the Details Input Data Frame section below. Otherwise, any character data will be coerced to numeric data with possibly undesirable results.</p>
analysisname	<i>Optional</i> , a character text or math-valid expression that will be set for default use in graph title and table methods. The default value is the empty "".
endptname	<i>Optional</i> , a character text or math-valid expression that will be set for default use as the y-axis label of graph methods, and also used for table methods. The default value is the empty "".
endptunits	<i>Optional</i> , a character text or math-valid expression that can be used in combination with the endptname argument. Parentheses are automatically added to this input, which will be added to the end of the endptname character value or expression. The default value is the empty "".
logscale	Apply a log-transformation to the data for evaluations. The default value is TRUE.
zeroscore	<i>Optional</i> , replace response values of zero with a derived or specified numeric value, as an approach to overcome the presence of zeroes when evaluation in the logarithmic scale (logscale=TRUE) is specified. The default value is NULL. To derive a score value to replace zero, "estimate" can be specified, see Details below on the algorithm used.
addconstant	<i>Optional</i> , add a numeric constant to all response values, as an approach to overcome the presence of zeroes when evaluation in the logarithmic scale logscale=TRUE is desired. The default value is NULL.
rightcensor	<i>Optional</i> , can be specified with a numeric value where any value equal to or greater will be regarded as right censored in the evaluation. The value of TRUE can be used to coerce a binary status variable in the data frame to be right censored for its values. The default value is NULL. See the Details Input Data Frame section below for specifications and consequences.
leftcensor	<i>Optional</i> , can be specified with a numeric value where any value equal to or lesser will be regarded as left censored in the evaluation. The value of TRUE can be used to coerce a binary status variable in the data frame to be right censored for its values. The default value is NULL. See the Details Input Data Frame section below for specifications and consequences.

digits	<i>Optional</i> , for output display purposes in graphs and table methods, values will be rounded to this numeric value. Only the integers of 0, 1, 2, 3, and 4 are accepted. No rounding is done during any calculations. The default value is NULL, which will examine each individual data value and choose the one that has the maximum number of digits after any trailing zeroes are ignored. The max number of digits will be 4.
refgrp	<i>Optional</i> , specify one of the factor levels to be the “reference group”, such as a “control” group. The default value is NULL, which will just use the first level determined in the data frame.
stamps	<i>Optional</i> , specify a time stamp in graphs, along with cg package version identification. The default value is FALSE.

Details

Input Data Frame The input data frame `dfr` can be of the format “`listed`” or “`groupcolumns`”. Another distinguishing characteristic is whether or not it contains censored data representations.

Censored observations can be represented by `<` for left-censoring and `>` for right-censoring. The `<` value refers to values less than or equal to a numeric value. For example, `<0.76` denotes a left-censored value of 0.76 or less. Similarly, `>2.02` denotes a value of 2.02 or greater for a right-censored value. There must be no space between the direction indicator and the numeric value. These representations can be used in either the `listed` or `groupcolumns` formats for `dfr`.

No interval-censored representations are currently handled when `format="groupcolumns"`.

If `format="groupcolumns"` for `dfr` is specified, then the number of columns must equal the number of groups, and any censored values must follow the `<` and `>` representations. The individual group values are of mode character, since any censored values will be represented for example as `<0.76` or `>2.02`. If any of the groups have less number of observations than any others, i.e. there are unequal sample sizes, then the corresponding “no data” cells in the data frame need to contain empty quote “” values.

If `format="listed"` for `dfr` is specified, then there may be anywhere from two to four columns for an input data frame.

two columns The first column has the group levels to define the factor, and the second column contains the response values. Censored representations of `<` and `>` can be used here. One or both of `rightcensor` or `leftcensor` may also be specified as a number. If a number is specified for `rightcensor`, then all values in the second column equal to this value will be processed as right-censored. Analogously, if a number is specified for `leftcensor`, then all values in the second column equal to this value will be processed as left-censored. **WARNING:** This should be used cautiously to make sure the equality occurs as desired. This convention is designed for simple Type I censoring scenarios.

three columns Like the two column case, the first column has the group levels to define the factor, and the second column contains the response values, which will all be coerced to numeric. Any censoring information must be specified in the third column. Borrowing the convention of `Surv` from the **survival** package, `0`=right censored, `1`=no censoring, and `2`=left censored. If `rightcensor=NULL` and `leftcensor=NULL` are left as defaults in the call, and values of `0`, `1`, and `2` are all represented, then the processing will create a suitable data frame in `dfru` for modeling that the canonical `survreg` function understands.

However, if 0 and 1 are the only specified values in the third censoring status column, then one of `rightcensor=TRUE` or `leftcensor=TRUE` must be specified, but NOT both, or an error message will occur. A column of all 1's or all 0's will also raise an error message.

four columns Like the two column case, the first column has the group levels to define the factor. The second and third columns need to have numeric response information, and the fourth column needs to have censoring status. This is the most general representation, where any combination of left-censoring, right-censoring, and interval-censoring is permitted. The `rightcensor` and `leftcensor` input arguments are ignored and set to NULL. **IMPORTANT:** The convention of `Surv` from the **survival** package, 0=right censored, 1=no censoring, and 2=left censored, 3=interval censored, and `type="interval"`, is followed. For `status=0, 1, and 2`, the second and third columns match in value, so that the status variable in the fourth column distinguishes the lower and upper bounds for the right-censored (0) and left-censored (2) cases. For `status=3`, the two values differ to define the interval boundaries. The processing will create a suitable data frame in `dfr` for modeling that the canonical `survreg` and `survfit` functions from the **survival** package understand.

zeroscore If `zeroscore="estimate"` is specified, a number close to zero is derived to replace all zeroes for subsequent log-scale analyses. A spline fit (using `spline` and `method="natural"`) of the log of the response vector on the original response vector is performed. The zeroscore is then derived from the log-scale value of the spline curve at the original scale value of zero. This approach comes from the concept of arithmetic-logarithmic scaling discussed in Tukey, Ciminera, and Heyse (1985).

addconstant If `addconstant="simple"` or `addconstant="VR"` is specified, a number is derived and added to all response values.

"simple" Taken from the "white" book on S (Chambers and Hastie, 1992), page 68. The range (`max - min`) of the response values is multiplied by 0.0001 to derive the number to add to all the response values.

"VR" Based on the `logtrans` function discussed in Venables and Ripley (2002), pages 171-172 and available in the **MASS** package. The algorithm applies a Box-Cox profile likelihood approach with a log scale translation model.

Value

A `cgOneFactorData` object is returned, with the following slots:

<code>dfr</code>	The original input data frame that is the specified value of the <code>dfr</code> argument in the function call.
<code>dfru</code>	Processed version of the input data frame, which will be used for the various evaluation methods.
<code>fmt.dfru</code>	A list version of the input data frame, which will only differ from the <code>dfr</code> value if the input data frame was specified in the <code>groupcolumns</code> format.
<code>has.censored</code>	Boolean TRUE or FALSE on whether there are any censored data observations.
<code>settings</code>	A list of properties associated with the data frame: <ul style="list-style-type: none"> <code>analysisname</code> Drawn from the input argument value of <code>analysisname</code>. <code>endptname</code> Drawn from the input argument value of <code>endptname</code>.


```
## Censored Data
data(gmcsfcens)
gmcsfcens.data <- prepareCGOneFactorData(gmcsfcens, format="groupcolumns",
                                         analysisname="cytokine",
                                         endptname="GM-CSF (pg/ml)",
                                         logscale=TRUE)
```

```
print.cgOneFactorComparisonsTable
```

Print One Factor Comparisons Table object with some format options

Description

Print a `cgOneFactorComparisonsTable` object, which contains a table of comparisons based on the `cgOneFactorFit` object.

Usage

```
## S4 method for signature 'cgOneFactorComparisonsTable'
print(x, digits = NULL, title = NULL, endptname = NULL, ...)
```

Arguments

<code>x</code>	An <code>cgOneFactorComparisonsTable</code> object, typically created by <code>comparisonsTable.cgOneFactorFit</code> .
<code>digits</code>	The number of decimal places to use in the output. If <code>NULL</code> , then the number of decimal places is taken from the <code>digits</code> value in the <code>settings</code> slot of the <code>cgOneFactorComparisonsTable</code> object.
<code>title</code>	The title printed out with the table. If <code>NULL</code> , it is set to be "Comparisons Table of" the <code>analysisname</code> value from the <code>settings</code> slot of the <code>cgOneFactorComparisonsTable</code> object.
<code>endptname</code>	The endpoint name, printed out with the table. If <code>NULL</code> , it is set to the <code>endptname</code> value from the <code>settings</code> slot of the <code>cgOneFactorComparisonsTable</code> object.
<code>...</code>	Additional arguments. Only one is currently valid: <code>model</code> For <code>cgOneFactorComparisonsTable</code> objects that have tables derived from classical least squares <code>lm</code> or resistant & robust <code>r1m</code> fits, the following argument values are possible: <code>"both"</code> Both the ordinary classical least squares and resistant robust comparisons tables are printed. This is the default when both fits are present in the <code>cgOneFactorComparisonsTable</code> object specified in the <code>x</code> argument. <code>"olonly"</code> Only the ordinary classical least squares group comparisons table is printed. <code>"rronly"</code> Only the resistant & robust comparisons table is printed.

For other possible `cgOneFactorComparisonsTable` table components such as accelerated failure time or unequal variance models, the `model` argument is not relevant, and the single table will just be printed for these model types.

Details

The smallest actual p-value that will be printed is 0.001. Anything less than 0.001 will be displayed as < 0.001 . If you need more digits, see the `cgOneFactorComparisonsTable` object.

The object is printed using a mix of `cat` and `print` calls. See `cgOneFactorComparisonsTable` for details of the `*`.`comprs` and other object slots.

Value

`print.cgOneFactorComparisonsTable` returns `invisible`. The main purpose is the side effect of printing to the current output connection, which is typically the console.

Note

Contact `<cg@billpikounis.net>` for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(canine.data)

canine.comps0 <- comparisonsTable(canine.fit)

print(canine.comps0, digits=1)

canine.comps1 <- comparisonsTable(canine.fit, mcadjust=TRUE,
                                 type="allgroupstocontrol", refgrp="CC")

print(canine.comps1, model="olonly")
```

```
print.cgOneFactorDescriptiveTable
```

Print a One Factor Descriptive Table object with some format options

Description

Print a `cgOneFactorDescriptiveTable` object, which contains a table of quantiles and other summary statistics of the data from a `cgOneFactorData` object.

Usage

```
## S4 method for signature 'cgOneFactorDescriptiveTable'  
print(x, digits = NULL, title = NULL, endptname = NULL, ...)
```

Arguments

<code>x</code>	A <code>cgOneFactorDescriptiveTable</code> object, typically created by descriptiveTable.cgOneFactorData .
<code>digits</code>	The number of decimal places to use in the output. If <code>NULL</code> , then the number of decimal places is taken from the <code>digits</code> value in the <code>settings</code> slot of the <code>cgOneFactorDescriptiveTable</code> object.
<code>title</code>	The title printed out with the table. If <code>NULL</code> , it is set to be "Descriptive Table of" the <code>analysisname</code> value taken from the <code>settings</code> slot of the <code>cgOneFactorDescriptiveTable</code> object.
<code>endptname</code>	The endpoint name of the data summarized in the table. If <code>NULL</code> , it is set to the <code>endpointname</code> value taken from the <code>settings</code> slot of the <code>cgOneFactorDescriptiveTable</code> object.
<code>...</code>	Additional arguments. None are currently defined for this method.

Details

The object is printed using a mix of `cat` and `print` calls. See [cgOneFactorDescriptiveTable](#) for details of the contents and other object slots.

Value

`print.cgOneFactorDescriptiveTable` returns `invisible`. The main purpose is the side effect of printing to the current output connection, which is typically the console.

Note

Contact cg@billpikounis.net for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

See Also[cgOneFactorDescriptiveTable](#)**Examples**

```

data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

## Next two calls are equivalent
descriptiveTable(canine.data)

print(descriptiveTable(canine.data, display="none"))

print(descriptiveTable(canine.data, display="none"),
      title="Quantiles and Summary Statistics")

## Censored Data
data(gmcsfcens)
gmcsfcens.data <- prepareCGOneFactorData(gmcsfcens, format="groupcolumns",
                                         analysisname="cytokine",
                                         endptname="GM-CSF (pg/ml)",
                                         logscale=TRUE)

## Next two calls are equivalent
descriptiveTable(gmcsfcens.data, display="print")
print(descriptiveTable(gmcsfcens.data, display="none"))

```

```
print.cgOneFactorDownweightedTable
```

Print Downweighted Observations Table object with some format options

Description

Print a `cgOneFactorDownweightedTable` object, as a table of downweighted observations in a resistant & robust fit from a `cgOneFactorFit` object.

Usage

```
## S4 method for signature 'cgOneFactorDownweightedTable'
print(x, digits=NULL, title = NULL, endptname = NULL, ...)
```

Arguments

x	An object of class cgOneFactorDownweightedTable , typically created by downweightedTable.cgOneFactorDownweightedTable .
digits	The number of decimal places to use in the output. If NULL, then the number of decimal places is taken from the cgOneFactorDownweightedTable object.
title	The title printed out with the p-value. If NULL, it is set to be "Downweighted Observations Table from Resistant & Robust Fit" of the analysisname value in the settings of the cgOneFactorDownweightedTable object.
endptname	The endpoint name, printed out with the p-value. If NULL, it is set to the endptname value in the cgOneFactorDownweightedTable object.
...	Additional arguments. None are currently defined for this method.

Details

The object is printed using a mix of cat and print calls. See [cgOneFactorDownweightedTable](#) for details of the contents and other object slots.

Value

print.cgOneFactorDownweightedTable returns [invisible](#). The main purpose is the side effect of printing to the current output connection, which is typically the console. If any observations meet the cutoff criteria, a table is displayed.

If no observations meet the cutoff criteria, a text message of table emptiness is displayed instead.

Note

Contact cg@billpikounis.net for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

See Also

[cgOneFactorDownweightedTable](#)

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(canine.data)

canine.dwtable <- downweightedTable(canine.fit, cutoff=0.95)

downweightedTable(canine.fit, cutoff=0.75) ## No observation
## downweighted at least 25%
```

```
print.cgOneFactorFit Print One Factor Model Fit object with some format options
```

Description

Print a `cgOneFactorFit` object, which contains fitted model information.

Usage

```
## S4 method for signature 'cgOneFactorFit'
print(x, title = NULL, endptname = NULL, ...)
```

Arguments

<code>x</code>	An <code>cgOneFactorFit</code> object.
<code>title</code>	The title printed out with the fitted model information. If <code>NULL</code> , it is set to be "Fitted Models of" the <code>analysisname</code> value in the <code>settings</code> slot of the <code>cgOneFactorFit</code> object.
<code>endptname</code>	The endpoint name, printed out with the fitted model information. If <code>NULL</code> , it is set to the <code>endptname</code> value in the <code>settings</code> slot of the <code>cgOneFactorFit</code> object.
<code>...</code>	Additional arguments. Only one is currently valid: <ul style="list-style-type: none"> <code>model</code> For <code>cgOneFactorFit</code> objects that have output derived from classical least squares <code>lm</code> or resistant & robust <code>rlm</code> fits, the following argument values are possible: <ul style="list-style-type: none"> <code>"both"</code> Both the ordinary classical least squares and resistant & robust model fits are printed. This is the default when both fits are present in the <code>cgOneFactorFit</code> object specified in the <code>x</code> argument. <code>"olonly"</code> Only the ordinary classical least squares model fit is printed. <code>"rronly"</code> Only the resistant & robust model fit is printed.

For other possible `cgOneFactorFit` components such as accelerated failure time or unequal variance model fits, the `model` argument is not relevant, and the single model fit will just be printed for these model types.

Details

The object is printed using a mix of `cat` and `print` calls. See `cgOneFactorFit` for details of the `*fit` and other object slots.

This method simply echoes print methods for individual fit classes, such as `lm` and `rlm`.

Note that `show` is an alias for `print` for this method. A `showObj.cgOneFactorFit` method is defined to display the raw form of the object.

Value

`print.cgOneFactorFit` returns `invisible`. The main purpose is the side effect of printing to the current output connection, which is typically the console.

Note

Contact <cg@billpikounis.net> for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

See Also

[cgOneFactorFit](#)

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(canine.data)

print(canine.fit)
```

```
print.cgOneFactorGlobalTest
```

Print One Factor Global F-test object with some format options

Description

Print a `cgOneFactorGlobalTest` object, which contains global F-test p-value information taken from a `cgOneFactorFit` object.

Usage

```
## S4 method for signature 'cgOneFactorGlobalTest'
print(x, title = NULL, endptname = NULL, ...)
```

Arguments

<code>x</code>	An cgOneFactorGlobalTest object, typically created by globalTest.cgOneFactorFit .
<code>title</code>	The title printed out with the p-value. If NULL, it is set to be "Group Test P-value of" the analysisname value in the settings slot of the cgOneFactorGlobalTest object.
<code>endptname</code>	The endpoint name, printed out with the p-value. If NULL, it is set to the endptname value in the settings slot of the cgOneFactorGlobalTest object.
<code>...</code>	Additional arguments. Only one is currently valid:

`model` For `cgOneFactorGlobalTest` objects that have p-values derived from classical least squares `lm` or resistant & robust `r1m` fits, the following argument values are possible:

"both" Both the ordinary classical least squares and resistant & robust p-values are printed. This is the default when both fits are present in the `cgOneFactorGlobalTest` object specified in the `x` argument.

"olsonly" Only the ordinary classical least squares p-value is printed.

"rronly" Only the resistant & robust approximated p-value is printed.

For other possible `cgOneFactorGlobalTest` p-value components such as accelerated failure time or unequal variance models, the `model` argument is not relevant, and the single p-value will just be printed for these model types.

Details

The smallest actual p-value that will be printed is 0.001. Anything less than 0.001 will be displayed as `< 0.001`. If you need more digits, see the `cgOneFactorGlobalTest` object.

The notion of a global F test, or equivalently, of R^2 , for resistant & robust linear models is murky, as no clear theoretical analogue to the ordinary classical least squares approach exists. See `cgOneFactorGlobalTest` for details, and regard the output p-value here as ad-hoc.

The object is printed using a mix of `cat` and `print` calls. See `cgOneFactorGlobalTest` for details of the `*.gpval` and other object slots.

Value

`print.cgOneFactorGlobalTest` returns `invisible`. The main purpose is the side effect of printing to the current output connection, which is typically the console.

Note

Contact `<cg@billpikounis.net>` for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

See Also

`cgOneFactorGlobalTest`

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(canine.data)
```

```
canine.global <- globalTest(canine.fit)

print(canine.global)
```

```
print.cgOneFactorGrpSummaryTable
```

Print One Factor Group Summary Table object with some format options

Description

Print a `cgOneFactorGrpSummaryTable` object, which contains a table of group means and variability based on the `cgOneFactorFit` object.

Usage

```
## S4 method for signature 'cgOneFactorGrpSummaryTable'
print(x, digits = NULL, title = NULL, endptname = NULL, ...)
```

Arguments

<code>x</code>	An <code>cgOneFactorGrpSummaryTable</code> object, typically created by <code>grpSummaryTable.cgOneFactorFit</code> .
<code>digits</code>	The number of decimal places to use in the output. If <code>NULL</code> , then the number of decimal places is taken from the <code>digits</code> value in the <code>settings</code> slot of the <code>cgOneFactorGrpSummaryTable</code> object.
<code>title</code>	The title printed out with the table. If <code>NULL</code> , it is set to be "Group Summary Table of" the <code>analysisname</code> value from the <code>settings</code> slot of the <code>cgOneFactorGrpSummaryTable</code> object.
<code>endptname</code>	The endpoint name, printed out with the table. If <code>NULL</code> , it is set to the <code>endptname</code> value from the <code>settings</code> slot of the <code>cgOneFactorGrpSummaryTable</code> object.
<code>...</code>	Additional arguments. Only one is currently valid: <ul style="list-style-type: none"> <code>model</code> For <code>cgOneFactorGrpSummaryTable</code> objects that have tables derived from classical least squares <code>lm</code> or resistant & robust <code>r1m</code> fits, the following argument values are possible: <ul style="list-style-type: none"> "both" Both the ordinary classical least squares and resistant robust comparisons tables are printed. This is the default when both fits are present in the <code>cgOneFactorGrpSummaryTable</code> object specified in the <code>x</code> argument. "olonly" Only the ordinary classical least squares comparisons table is printed. "rronly" Only the resistant and robust comparisons table is printed. For other possible <code>cgOneFactorGrpSummaryTable</code> table components such as accelerated failure time or unequal variance models, the <code>model</code> argument is not relevant, and the single table will just be printed for these <code>model</code> types.

Details

The object is printed using a mix of cat and print calls. See [cgOneFactorGrpSummaryTable](#) for details of the *.grps and other object slots.

Value

print.cgOneFactorGrpSummaryTable returns `invisible`. The main purpose is the side effect of printing to the current output connection, which is typically the console.

Note

Contact <cg@billpikounis.net> for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

See Also

[cgOneFactorGrpSummaryTable](#)

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(canine.data)

canine.grpsumm <- grpSummaryTable(canine.fit)

print(canine.grpsumm, digits=2)
```

```
print.cgOneFactorSampleSizeTable
```

Print a One Factor Sample Size Table object with some format options

Description

Print a `cgOneFactorSampleSizeTable` object, which contains a table of sample size estimates based on a `cgOneFactorFit` object.

Usage

```
## S4 method for signature 'cgOneFactorSampleSizeTable'
print(x, title=NULL, endptname=NULL, ...)
```

Arguments

x	A cgOneFactorSampleSizeTable object, typically created by samplesizeTable.cgOneFactorFit .								
title	The title for the table. If NULL, it is set to be "Sample Size Table from" concatenated to planningname value in the settings slot of the cgOneFactorSampleSizeTable object.								
endptname	The endpoint name, printed out with the table. If NULL, it is set to the endptname value from the settings slot of the cgOneFactorSampleSizeTable object.								
...	Additional arguments. Currently one is valid: <table> <tr> <td>model</td> <td>For cgOneFactorComparisonsTable objects that have tables derived from classical least squares lm or resistant & robust r1m fits, the following argument values are possible: <table> <tr> <td>"both"</td> <td>Both the ordinary classical least squares and resistant robust sample size tables are printed. This is the default when both fits are present in the cgOneFactorSampleSizeTable object specified in the x argument. If only one of the two tables were computed, then only that single table is printed.</td> </tr> <tr> <td>"olonly"</td> <td>Only the ordinary classical least squares group summary table is printed.</td> </tr> <tr> <td>"rronly"</td> <td>Only the resistant & robust group summary table is printed.</td> </tr> </table> </td> </tr> </table>	model	For cgOneFactorComparisonsTable objects that have tables derived from classical least squares lm or resistant & robust r1m fits, the following argument values are possible: <table> <tr> <td>"both"</td> <td>Both the ordinary classical least squares and resistant robust sample size tables are printed. This is the default when both fits are present in the cgOneFactorSampleSizeTable object specified in the x argument. If only one of the two tables were computed, then only that single table is printed.</td> </tr> <tr> <td>"olonly"</td> <td>Only the ordinary classical least squares group summary table is printed.</td> </tr> <tr> <td>"rronly"</td> <td>Only the resistant & robust group summary table is printed.</td> </tr> </table>	"both"	Both the ordinary classical least squares and resistant robust sample size tables are printed. This is the default when both fits are present in the cgOneFactorSampleSizeTable object specified in the x argument. If only one of the two tables were computed, then only that single table is printed.	"olonly"	Only the ordinary classical least squares group summary table is printed.	"rronly"	Only the resistant & robust group summary table is printed.
model	For cgOneFactorComparisonsTable objects that have tables derived from classical least squares lm or resistant & robust r1m fits, the following argument values are possible: <table> <tr> <td>"both"</td> <td>Both the ordinary classical least squares and resistant robust sample size tables are printed. This is the default when both fits are present in the cgOneFactorSampleSizeTable object specified in the x argument. If only one of the two tables were computed, then only that single table is printed.</td> </tr> <tr> <td>"olonly"</td> <td>Only the ordinary classical least squares group summary table is printed.</td> </tr> <tr> <td>"rronly"</td> <td>Only the resistant & robust group summary table is printed.</td> </tr> </table>	"both"	Both the ordinary classical least squares and resistant robust sample size tables are printed. This is the default when both fits are present in the cgOneFactorSampleSizeTable object specified in the x argument. If only one of the two tables were computed, then only that single table is printed.	"olonly"	Only the ordinary classical least squares group summary table is printed.	"rronly"	Only the resistant & robust group summary table is printed.		
"both"	Both the ordinary classical least squares and resistant robust sample size tables are printed. This is the default when both fits are present in the cgOneFactorSampleSizeTable object specified in the x argument. If only one of the two tables were computed, then only that single table is printed.								
"olonly"	Only the ordinary classical least squares group summary table is printed.								
"rronly"	Only the resistant & robust group summary table is printed.								

Details

The object is printed using a mix of cat and print calls. See [cgOneFactorSampleSizeTable](#) for details of the *.sstable and other object slots.

Value

print.cgOneFactorSampleSizeTable returns [invisible](#). The main purpose is the side effect of printing to the current output connection, which is typically the console.

Note

Contact <cg@billpikounis.net> for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

See Also

[cgOneFactorSampleSizeTable](#)

Examples

```

data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(canine.data)

canine.samplesize <- samplesizeTable(canine.fit, direction="increasing",
                                     model="olsonly",
                                     mmdvec=c(10, 25, 50, 75, 100), display="none")

print(canine.samplesize)

```

qqGraph

Quantile-Quantile Graphs

Description

Create a Quantile-Quantile (Q-Q) Gaussian graph of the residuals of a fitted object from the **cg** package.

Usage

```
qqGraph(fit, line = TRUE, cgtheme = TRUE, device = "single", ...)
```

Arguments

fit	A fit object, typically created by the fit generic function.
line	Add a line to help assess the distribution of the residuals. See specific method written for the <code>fit</code> argument.
cgtheme	When set to the default TRUE, ensures a trellis device is active with limited color scheme. Namely <code>background</code> , <code>strip.shingle</code> , and <code>strip.background</code> are each set to "white".
device	Can be one of three values: "single" The default, which will put all graphs on the same device page. "multiple" Relevant only when multiple fitted models are present in the <code>fit</code> object and requested to be plotted. In those cases, a new graphics device is generated to hold each additional plot beyond the first. "ask" Relevant only when multiple fitted models are present in the <code>fit</code> object and requested to be plotted. In these cases, each plot is portrayed as a single-paneled graph, with the <code>ask=TRUE</code> argument specified in par so that the user input confirmation is needed before the graphs are drawn.
...	Additional arguments, depending on the specific method written for the object. See the method-specific documentation for additional details.

Value

qqGraph returns an invisible NULL. The main purpose, of course, is the side effect of graphing to the current device.

Note

Contact <cg@billpikounis.net> for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis

See Also

[qqGraph.cgOneFactorFit](#)

qqGraph.cgOneFactorFit

Quantile-Quantile (QQ) Graphs of a cgOneFactorFit object

Description

Create a Q-Q Gaussian graph of the residuals of a cgOneFactorFit object

Usage

```
## S4 method for signature 'cgOneFactorFit'
qqGraph(fit, line=NULL, cgtheme = TRUE, device = "single", ...)
```

Arguments

fit	A fit object of class cgOneFactorFit .
line	Add a line through the estimated 25th and 75th percentiles. When set to the default NULL, the addition of a line depends on the following: When there is no censored data, the line will be added with the qqline algorithm. If any censored data residuals are present, no line is added unless line=TRUE is explicitly specified below.
cgtheme	When set to the default TRUE, ensures a trellis device is active with limited color scheme. Namely, background, strip.shingle, and strip.background are each set to "white".
device	Can be one of three values: "single" The default, which will put all graphs on the same device page. For example, when resistant & robust and classical least squares are present and model="both" (the default), a 2 x 1 paneled graph will be created.

"multiple" Relevant only when resistant & robust and classical least squares are present and model="both" (the default) or model="extended". In those cases, a new graphics device is generated to hold the resistant & robust version, as a single-paneled graph, and the classical least squares version is on the previous device. If model="extended", then a second new graphics device is generated to hold the unweighted resistant & robust residuals, as another single-paneled graph.

"ask" Relevant only when resistant & robust and classical least squares are present and model="both" (the default) or model="extended". In that case, each are portrayed as a single-paneled graph, with the ask=TRUE argument specified in `par` so that the user input confirmation is needed before the graphs are drawn.

...

Additional arguments. One is currently valid:

`model` For `cgOneFactorFit` fit objects that have classical least squares `lm()` or resistant & robust `rlm()` fits, the following argument values are possible:

"both" Q-Q graphs of the residuals from both the ordinary classical least squares and resistant & robust fits are plotted. This is the default when both fits are present in the `cgOneFactorFit` object specified in the `fit` argument. If the resistant & robust fit is not available, this value is not relevant.

"olsonly" Only a Q-Q graph of the residuals from the ordinary classical least squares `olsfit` fit is performed.

"rronly" or "rrwtdonly" Only a Q-Q graph of the weighted residuals from the resistant and robust `rrfit` fit is plotted.

"rrunwtdonly" Only a Q-Q graph of the unweighted residuals from the resistant and robust `rrfit` fit is plotted.

For other possible `cgOneFactorFit` fit slots such as accelerated failure time or unequal variance models, the `model` argument is not relevant, and the appropriate Q-Q graph will be plotted for these model types.

Details

For censored data residuals, left-censored values are shown as a shallow "V", which is actually just a rotated downward "<" sign. Similarly, right-censored values are shown as a deeper "^", which is a rotated upward ">" sign.

For the `line` argument, an added line when censored data residuals are present needs to be interpreted very cautiously. If "too many" censored data values are present, the line will appear nonsensical if indeed it can even be estimated with 25th and 75th percentiles in the presence of the censored data residuals. These percentiles are estimated via the Kaplan-Meier method as proposed by Gentleman and Crowley (1991), with the `survival::survfit` function.

The heading for the graph is taken from the `cgOneFactorData` object, which `prepareCGOneFactorData` sets from its `analysisname` argument.

The label for the Y-axis is taken from the `cgOneFactorData` object, which `prepareCGOneFactorData` sets from its `endptname` argument.

The number of decimal places printed in the ticks on the Y-axis is taken from the `cgOneFactorData` object, which `prepareCGOneFactorData` sets from its `digits` argument.

The minimum and maximum values from the range of the residuals are respectively labeled in the bottom and top left corners of the graph region.

Value

qqGraph.cgOneFactorFit returns an invisible NULL. The main purpose, of course, is the side effect of graphing to the current device.

Note

Contact <cg@billpikounis.net> for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

References

Gentleman, R.C. and Crowley, J. (1991). "Graphical Methods for Censored Data", *Journal of the American Statistical Association*, Volume 86, 678-683.

See Also

[qqline](#)

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(canine.data)

qqGraph(canine.fit)

qqGraph(canine.fit, model="olonly")
```

samplesizeGraph

Graph Estimated Sample Sizes

Description

Generic function to graph a table of estimated sample sizes, using a Sample Size table created by the **cg** package.

Usage

```
sampleSizeGraph(sstable, Nscale = "log", mmdscale = "log",
  cgtheme=TRUE, device = "single", ...)
```

Arguments

sstable	A sampleSizeTable created by a sampleSizeTable method from the cg package.
Nscale	A character indicating whether the y-axis, which shows the estimated samples sizes, should be drawn on the log scale ("log") or the original scale ("original").
mmdscale	A character indicating whether the x-axis, which shows the minimum meaningful differences to be detected, should be drawn on the log scale ("log") or the original scale ("original").
cgtheme	When set to the default TRUE, ensures a trellis device is active with limited color scheme. Namely background, strip.shingle, and strip.background are each set to "white".
device	Can be one of three values: "single" The default, which will put all graph panels on the same device page. "multiple" Relevant only when more than one panel of graphs is possible. In that case, a new graphics device is generated each newly generated single-paneled graph. "ask" Relevant only when more than one panel of graphs is possible. In that case, each are portrayed as a single-paneled graph, with the ask=TRUE argument specified in par so that the user input confirmation is needed before the graphs are drawn.
...	Additional arguments, depending on the specific method written for the object. See the specific methods for additional details.

Value

The main purpose is the side effect of graphing to the current device. See the specific methods for discussion of any return values.

Note

Contact <cg@billpikounis.net> for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

See Also

[sampleSizeGraph.cgOneFactorSampleSizeTable](#)

Examples

```

data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(canine.data)

canine.sampleSize <- sampleSizeTable(canine.fit, direction="increasing",
                                     mmdvec=c(10, 25, 50, 75, 100))

sampleSizeGraph(canine.sampleSize)

```

```
sampleSizeGraph.cgOneFactorSampleSizeTable
```

Graph estimated sample sizes from a cgOneFactorSampleSizeTable object

Description

Creates a graph to see estimated sample sizes in a cgOneFactorSampleSizeTable object.

Usage

```

## S4 method for signature 'cgOneFactorSampleSizeTable'
sampleSizeGraph(sstable, Nscale="log", mmdscale = "log",
               cgtheme = TRUE, device = "single", ...)

```

Arguments

sstable	A sample size object of class <code>cgOneFactorSampleSizeTable</code> .
Nscale	A character indicating whether the Y-axis, which shows the estimated samples sizes, should be drawn on the log scale ("log") or the original scale ("original").
mmdscale	A character indicating whether the X-axis, which shows the minimum meaningful differences to be detected, should be drawn on the log scale ("log") or the original scale ("original").
cgtheme	When set to the default TRUE, ensures a trellis device is active with limited color scheme. Namely <code>background</code> , <code>strip.shingle</code> , and <code>strip.background</code> are each set to "white".
device	Can be one of three values: "single" The default, which will put all graphs on the same device page. For example, when resistant & robust and classical least squares sample size estimates are present and <code>model="both"</code> (the default), a 2 x 1 paneled graph will be created.

"multiple" Relevant only when resistant & robust and classical least squares sample size estimates are present and model="both". In those cases, a new graphics device is generated to hold the resistant & robust version, as a single-paneled graph, and the classical least squares version is on the previous device.

"ask" Relevant only when resistant & robust and classical least squares sample size estimates are present and model="both" (the default). In that case, each are portrayed as a single-paneled graph, with the ask=TRUE argument specified in `par` so that the user input confirmation is needed before the graphs are drawn.

...

Additional arguments. Two are currently valid:

`model` For `cgOneFactorOneSampleSizeTable` objects that have classical least squares `lm()` filled or resistant & robust `r1m()` filled slots, the following argument values are possible:

"both" Sample Size graphs based on both the ordinary classical least squares and resistant & robust slots are populated. This is the default when both slots are present in the `cgOneFactorSampleSizeTable` object specified in the `sstable` argument. If the resistant & robust fit is not available, this value is not relevant.

"olsonly" Only an Sample Size Table Graph based on the ordinary classical least squares table slot is created.

"rronly" Only a Sample Size Table Graph based on the resistant and robust table slot is created.

`mmdticklabels` A list of two components:

`mod` Can be either of these two values,

"replace" Before graphing the data, remove any automatically generated tickmarks for the x-axis, and create the tickmarks specified in the `marks` component below.

"add" Before graphing the data, add tickmarks specified in the `marks` component below, to the automatically generated ones.

`marks` A vector of tickmarks to be placed on the x-axis. Any numeric representations will be coerced to character.

`Nticklabels` A list of two components:

`mod` Can be either of these two values,

"replace" Before graphing the data, remove any automatically generated tickmarks for the x-axis, and create the tickmarks specified in the `marks` component below.

"add" Before graphing the data, add tickmarks specified in the `marks` component below, to the automatically generated ones.

`marks` A numeric vector of tickmarks to be placed on the y-axis. Any numeric representations will be coerced to character.

Details

The minimum and maximum sample size values are added inside the plot region in blue, flush against the y-axis in the top and bottom left corners.

Tick marks are attempted to be chosen wisely. For log-scaled axes in particular, leading digits of 2, 5, and 10 for values are included if possible. Since the algorithm is empirical, the `ticklabels` argument is available for further refinement or complete replacement of tickmarks.

Value

`sampleSizeGraph.cgOneFactorSampleSizeTable` returns an invisible NULL. The main purpose is the side effect of graphing to the current device.

Note

Contact <cg@billpikounis.net> for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

See Also

[cgOneFactorSampleSizeTable](#)

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(canine.data)

canine.sampleSize <- sampleSizeTable(canine.fit, direction="increasing",
                                    mmdvec=c(10, 25, 50, 75, 100))

sampleSizeGraph(canine.sampleSize)

sampleSizeGraph(canine.sampleSize, model="olsonly",
                mmdticklabels=list(mod="add", marks=100))
```

sampleSizeTable

Estimate Required Sample Sizes

Description

Estimate the sample size required to detect a specified difference in a future study. The estimate is based on the variability in a data fit with the **cg** package.

Usage

```
sampleSizeTable(fit, ngrps = 2, direction, mmdvec, power = 0.80,
  alpha = 0.05, nmax = 1000, display = "print", ...)
```

Arguments

<code>fit</code>	An object created by calling a <code>fit</code> method from the <code>cg</code> package. The only class of object currently available is <code>cgOneFactorSampleSizeTable</code> , which is prepared by the <code>sampleSizeTable.cgOneFactorFit</code> method.
<code>ngrps</code>	The number of groups that will be in the future study.
<code>direction</code>	A character indicating whether the sample size should be estimated to detect an "increase" or a "decrease". This only effects the sample size estimates if the previous study in <code>fit</code> was analyzed on the log scale, in which case the differences in <code>mmdvec</code> are relative differences instead of absolute differences. For detecting relative changes, the sample size required to detect a relative increase of 25% is not the same as the sample size to detect a relative decrease of 25%. But for detecting absolute changes, the sample size required to detect an absolute increase of 25 is the same as the sample size to detect an absolute decrease of 25.
<code>mmdvec</code>	A numeric vector specifying the minimum meaningful differences to be detected in the future study. If the previous study in <code>fit</code> was analyzed on the log scale, then the values in <code>mmdvec</code> are assumed to be relative percentage increases or decreases, depending on the value of <code>direction</code> . If the previous study in <code>fit</code> was not analyzed on the log scale, then the values in <code>mmdvec</code> are assumed to be absolute increases or decreases, depending on the value of <code>direction</code> . Each value in <code>mmdvec</code> needs to be positive.
<code>power</code>	The power for the future study, set by default to be 0.80. This is equivalent to $1 - \beta$, where β is the probability of committing a Type II error: accepting the null hypothesis of no difference when differences truly exist.
<code>alpha</code>	The significance level or alpha for the future study, set by default as 0.05.
<code>nmax</code>	The maximum number of subjects per group. If more subjects are estimated to be required, than the exact number required is not reported, only the fact that more than the maximum number would be required. This is in place to prevent long and likely unnecessary calculations.
<code>display</code>	One of three valid values: " <code>print</code> " The default value, it calls a <code>print</code> method for the created <code>sampleSizeTable</code> object, which is a formatted text output of the table(s). " <code>none</code> " Suppresses any printing. Useful, for example, when just assignment of the resulting object is desired. " <code>show</code> " Calls the default <code>showDefault</code> method, which will just print out the <code>sampleSizeTable</code> components.
<code>...</code>	Additional arguments, depending on the specific method written for the object. See the method-specific documentation for additional details.

Value

A method-specific SampleSizeTable object is returned. See the specific methods for discussion of return values.

Note

Contact <cg@billpikounis.net> for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

See Also

[sampleSizeTable.cgOneFactorFit](#)

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(canine.data)

canine.sampleSize <- sampleSizeTable(canine.fit, direction="increasing",
                                     mmdvec=c(10, 25, 50, 75, 100))

sampleSizeGraph(canine.sampleSize)
```

sampleSizeTable.cgOneFactorFit

Estimate Sample Sizes based on a cgOneFactorFit object

Description

Estimate the sample size that would be required to detect a specified difference in a one factor study. The estimate is based on the variability that was observed in a previous one factor study. A cgOneFactorSampleSizeTable class object is created.

Usage

```
## S4 method for signature 'cgOneFactorFit'
sampleSizeTable(fit, ngrps = 2, direction, mmdvec,
               power = 0.80, alpha = 0.05, nmax = 1000, display = "print", ...)
```

Arguments

<code>fit</code>	A <code>cgOneFactorFit</code> object from a previous one factor study.
<code>ngrps</code>	The number of groups that will be in the future one factor study.
<code>direction</code>	A character value indicating whether the sample size should be estimated to detect an "increase" or a "decrease". This only effects the sample size estimates if the previous study in <code>fit</code> was analyzed on the log scale, in which case the differences in <code>mmdvec</code> are relative differences instead of absolute differences. For detecting relative changes, the sample size required to detect a relative increase of 25% is not the same as the sample size to detect a relative decrease of 25%, for example. But for detecting absolute changes, the sample size required to detect an absolute increase of 25 is the same as the sample size to detect an absolute decrease of 25.
<code>mmdvec</code>	A numeric vector specifying the minimum meaningful differences to be detected in the future study. If the previous study in <code>fit</code> was analyzed on the log scale, then the values in <code>mmdvec</code> are assumed to be relative percentage increases or decreases, depending on the value of <code>direction</code> . If the previous study in <code>fit</code> was not analyzed on the log scale, then the values in <code>mmdvec</code> are assumed to be absolute increases or decreases, depending on the value of <code>direction</code> . Each value in <code>mmdvec</code> needs to be positive.
<code>power</code>	The power for the future study, set by default to be 0.80. This is equivalent to $1 - \beta$, where β is the probability of committing a Type II error: accepting the null hypothesis of no difference when differences truly exist.
<code>alpha</code>	The significance level or alpha for the future study, set by default as 0.05.
<code>nmax</code>	The maximum number of subjects per group. If more subjects are estimated to be required, than the exact number required is not reported, only the fact that more than the maximum number would be required. This is in place to prevent long and likely unnecessary calculations.
<code>display</code>	One of three valid values: <p>"print" The default value, it calls a <code>print</code> method for the created <code>cgOneFactorComparisonsTable</code> object, which is a formatted text output of the table(s).</p> <p>"none" Suppresses any printing. Useful, for example, when just assignment of the resulting object is desired.</p> <p>"show" Calls the default <code>showDefault</code> method, which will just print out the <code>cgOneFactorComparisonsTable</code> components.</p>
<code>...</code>	Additional arguments. Only one is currently valid:
<code>model</code>	A character value indicating which variability estimate in <code>fit</code> should be used to estimate the sample size: the robust model ("rroly"), the classical model ("olonly"), or both ("both"). If an estimate is requested for a model that was not fit, then no sample sizes are returned for that model but an error is not reported (e.g. if only the classical model was fitted but "both" are requested, only the classical model estimates will be returned): <p>"both" Sample Size tables based on both the ordinary classical least squares and resistant & robust fits are performed. This is the default when both</p>

fits are present in the `cgOneFactorFit` object specified in the `fit` argument. If the resistant & robust fit is not available, this value is not relevant.

"olonly" Only a sample size table based on the ordinary classical least squares `olsfit` fit is calculated.

"rronly" Only a sample size table based on the resistant and robust `rrfit` fit is calculated.

Details

This sample size method does not work for fitted models that allowed unequal variances or censored observations.

Sample sizes are estimated for detecting a minimum difference with a global F test. The algorithm is detailed in Fleiss (1986), Appendix A. When there are more than 2 groups, the lower bound of possible noncentrality parameter values is calculated from assuming only two of the `ngrps` number of groups differ by the `mmdvec/2` amount from the "grand mean" while the rest of the groups are equal to the grand mean.

For detecting an absolute difference, the sample size is the smallest group size n for which $1 - \text{pf}(\text{qf}(1 - \alpha, \text{numdf}, \text{denof}), \text{numdf}, \text{denof}, \text{ncp})$ exceeds power, where $\text{ncp} = (n * \text{mmdvec}^2) / (2 * \text{sigmaest}^2)$, and `sigmaest` is the residual mean square error from the model in `fit`. For detecting a relative difference, the calculations are the same except $\text{ncp} = (n * (\log(\text{sign} * \text{mmdvec} / 100 + 1))^2) / (2 * \text{sigmaest}^2)$, where `sign` = -1 if `direction="decreasing"`, and `sign` = 1 if `direction = "increasing"`.

Value

Creates an object of class `cgOneFactorSampleSizeTable`, with the following slots:

`ols.sstable` A matrix with the estimated sample sizes based on the classical model variance estimates, or NULL. The matrix has 3 columns and one row for each element of the `mmdvec` vector. The first column specifies the minimum meaningful difference ("`mmd`"). The second column gives the number of subjects required for each group ("`n`"), possibly truncated at `nmax`. The third column gives the total number of subjects required ("`N`"), also possibly truncated at `nmax`.

`rr.sstable` A matrix with the estimated sample sizes based on the robust model variance estimates, or else NULL if `model="olonly"` was specified. See the `ols.sstable` slot description above for the analogous layout of the matrix.

`settings` A list of properties mostly carried as-is from the data argument object of class `cgOneFactorData`, with the following additional members:

`sigmaest` A list with 2 members, `ols`, containing the estimated spread (sigma, standard deviation) from the classical model of `fit`, and `rr`, containing the estimated spread (sigma, standard deviation) from the robust model of `fit`, or NULL if the robust model was not fit.

`planningname` A character describing the study or purpose of the sample size analysis. Taken from the `settings$analysisname` of the `fit` object.

`ngrps` A saved copy of the `ngrps` argument.

`direction` A saved copy of the `direction` argument.

`alpha` A saved copy of the `alpha` argument.

`power` A saved copy of the `power` argument.

`nmax` A saved copy of the `nmax` argument.

Note

Contact <cg@billpikounis.net> for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

References

Fleiss, J.L. (1986). *The Design and Analysis of Clinical Experiments*, Appendix A, pages 371 - 376. New York: Wiley.

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(canine.data)

canine.samplesize <- samplesizeTable(canine.fit, direction="increasing",
                                     mmdvec=c(10, 25, 50, 75, 100))

samplesizeTable(canine.fit, direction="decreasing",
                mmdvec=c(25, 50, 75), model="olsonly")
```

```
show.cgOneFactorComparisonsTable
```

Show an One Factor Comparisons Table object from the cg package

Description

Show a `cgOneFactorComparisonsTable` object, which contains information of comparisons based on a fit in a `cgOneFactorFit` object.

Usage

```
## S4 method for signature 'cgOneFactorComparisonsTable'
show(object)
```

Arguments

`object` A `cgOneFactorComparisonsTable` object, typically created by [comparisonsTable.cgOneFactorFit](#).

Details

The object is shown using `showDefault`. See `cgOneFactorComparisonsTable` for details of the object slots.

Value

`show.cgOneFactorComparisonsTable` returns `invisible`. The main purpose is the side effect of printing the whole object to the current output connection, which is typically the console.

Note

Contact <cg@billpikounis.net> for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

See Also

`cgOneFactorComparisonsTable`, `showDefault`

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(canine.data)
show(comparisonsTable(canine.fit, display="none"))
```

```
show.cgOneFactorDescriptiveTable
```

Show an One Factor Descriptive Table object from the cg package

Description

Show a `cgOneFactorDescriptiveTable` object, which contains a table of quantiles and other summary statistics of the data from a `cgOneFactorData` object.

Usage

```
## S4 method for signature 'cgOneFactorDescriptiveTable'
show(object)
```

Arguments

object A `cgOneFactorDescriptiveTable` object, typically created by `descriptiveTable.cgOneFactorData`.

Details

The object is shown using `showDefault`. See `cgOneFactorDescriptiveTable` for details of the object slots.

Value

`show.cgOneFactorDescriptiveTable` returns `invisible`. The main purpose is the side effect of printing the whole object to the current output connection, which is typically the console.

Note

Contact <cg@billpikounis.net> for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

See Also

[cgOneFactorDescriptiveTable](#), [showDefault](#)

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")
show(descriptiveTable(canine.data, display="none"))

## Censored Data
data(gmcsfcens)
gmcsfcens.data <- prepareCGOneFactorData(gmcsfcens, format="groupcolumns",
                                         analysisname="cytokine",
                                         endptname="GM-CSF (pg/ml)",
                                         logscale=TRUE)
show(descriptiveTable(gmcsfcens.data, display="none"))
```



```
canine.fit <- fit(canine.data)
show(downweightedTable(canine.fit, cutoffwt=0.95, display="none"))
```

```
show.cgOneFactorGlobalTest
```

Show an Global Test object from the cg package

Description

Show a `cgOneFactorGlobalTest` object, which contains p-value information from a `cgOneFactorFit` object.

Usage

```
## S4 method for signature 'cgOneFactorGlobalTest'
show(object)
```

Arguments

`object` A `cgOneFactorGlobalTest` object, typically created by [globalTest.cgOneFactorFit](#).

Details

The object is shown using [showDefault](#). See [cgOneFactorGlobalTest](#) for details of the object slots.

Value

`show.cgOneFactorGlobalTest` returns [invisible](#). The main purpose is the side effect of printing the whole object to the current output connection, which is typically the console.

Note

Contact cg@billpikounis.net for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

See Also

[cgOneFactorGlobalTest](#), [showDefault](#)

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(canine.data)
globalTest(canine.fit, display="show")
show(globalTest(canine.fit, display="none"))
```

```
show.cgOneFactorGrpSummaryTable
```

Show an One Factor Group Summary Table object from the cg package

Description

Show a `cgOneFactorGrpSummaryTable` object, which contains information of group mean and standard error summaries based on a fit in a `cgOneFactorFit` object.

Usage

```
## S4 method for signature 'cgOneFactorGrpSummaryTable'
show(object)
```

Arguments

`object` A `cgOneFactorGrpSummaryTable` object, typically created by [grpSummaryTable.cgOneFactorFit](#).

Details

The object is shown using [showDefault](#). See [cgOneFactorGrpSummaryTable](#) for details of the object slots.

Value

`show.cgOneFactorGrpSummaryTable` returns `invisible`. The main purpose is the side effect of printing the whole object to the current output connection, which is typically the console.

Note

Contact cg@billpikounis.net for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

See Also

[cgOneFactorGrpSummaryTable](#), [showDefault](#)

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(canine.data)
show(grpSummaryTable(canine.fit, display="none"))
```

show.cgOneFactorSampleSizeTable

Show a One Factor Sample Size Table object from the cg package

Description

Show a `cgOneFactorSampleSizeTable` object, which contains a table of sample size estimates based on a `cgOneFactorFit` object.

Usage

```
## S4 method for signature 'cgOneFactorSampleSizeTable'
show(object)
```

Arguments

object A `cgOneFactorSampleSizeTable` object, typically created by [sampleSizeTable.cgOneFactorFit](#).

Details

The object is shown using [showDefault](#). See [cgOneFactorSampleSizeTable](#) for details of the object slots.

Value

`show.cgOneFactorSampleSizeTable` returns `invisible`. The main purpose is the side effect of printing the whole object to the current output connection, which is typically the console.

Note

Contact cg@billpikounis.net for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

See Also

[cgOneFactorSampleSizeTable](#), [showDefault](#)

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(canine.data)
show(samplesizeTable(canine.fit, direction="increasing",
                    mmdvec=c(25, 50, 75, 100), display="none"))
```

showObj

Show raw form of an object from the cg package

Description

Show the raw form of an object from the cg package.

Usage

```
showObj(object)
```

Arguments

object An object created by the **cg** package. The only class of object currently available is [cgOneFactorFit](#).

Details

The object raw form is shown using [showDefault](#). The name showObj is designed for use when the conventional show name is an alias for print in the cg package.

Value

A method-specific fit object is returned. See the specific methods for discussion of return values.

Note

Contact <cg@billpikounis.net> for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

See Also

[cgOneFactorFit](#), [showDefault](#)

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(canine.data)

showObj(canine.fit)
```

```
showObj.cgOneFactorFit
```

Show an Fitted Model object from the cg package

Description

Show the raw form of a `cgOneFactorFit` object, which contains model fit information.

Usage

```
## S4 method for signature 'cgOneFactorFit'
showObj(object)
```

Arguments

`object` A `cgOneFactorFit` object.

Details

The object is shown using [showDefault](#). See [cgOneFactorFit](#) for details of the object slots.

The name `showObj` is designed for use for cases like this when the conventional show name is an alias for `print`.

Value

`showObj.cgOneFactorFit` returns `invisible`. The main purpose is the side effect of printing the whole object to the current output connection, which is typically the console.

Note

Contact <cg@billpikounis.net> for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

See Also

[cgOneFactorFit](#), [showDefault](#)

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(canine.data)

showObj(canine.fit)

show(canine.fit) ## alias for print method on the object
```

summary.cgOneFactorFit

Summary display of a One Factor Model Fit object with some format options

Description

Summary printing of a cgOneFactorFit object, which contains fitted model information.

Usage

```
## S4 method for signature 'cgOneFactorFit'
summary(object, title = NULL, endptname = NULL, ...)
```

Arguments

object	An cgOneFactorFit object.
title	The title printed out with the summary of the fitted model(s). If NULL, it is set to be "Fitted Model Summaries of" the analysisname value in the settings slot of the cgOneFactorFit object.
endptname	The endpoint name, printed out with the fitted model information. If NULL, it is set to the endptname value in the settings slot of the cgOneFactorFit object.

... Additional arguments. Only one is currently valid:

`model` For `cgOneFactorFit` objects that have output derived from classical least squares `lm` or resistant & robust `rlm` fits, the following argument values are possible:

- "both" Both the ordinary classical least squares and resistant & robust model fit summaries are printed. This is the default when both fits are present in the `cgOneFactorFit` object specified in the object argument.
- "olonly" Only the ordinary classical least squares model fit summary is printed.
- "rronly" Only the resistant & robust model fit summary is printed.

For other possible `cgOneFactorFit` components such as accelerated failure time or unequal variance model fits, the `model` argument is not relevant, and the single model fit summary is printed for these model types.

Details

The object summary is printed using a mix of `cat` and `print` calls. See `cgOneFactorFit` for details of the `*fit` and other object slots.

This method simply echoes summary methods for individual fit classes, such as `lm` and `rlm`.

Value

`summary.cgOneFactorFit` returns `invisible`. The main purpose is the side effect of printing to the current output connection, which is typically the console.

Note

Contact `<cg@billpikounis.net>` for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

See Also

[cgOneFactorFit](#)

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(canine.data)

summary(canine.fit)
```

varianceGraph	<i>Variance Graphs</i>
---------------	------------------------

Description

Create an equal variance assessment graph of the residuals of a fitted object from the **cg** package

Usage

```
varianceGraph(fit, trend = NULL, cgtheme = TRUE,
  device = "single", ...)
```

Arguments

fit	A fit object, typically created by the <code>fit</code> generic function.
trend	Add a trend line to help assess the trend of the residuals. See specific method written for the <code>fit</code> argument.
cgtheme	When set to the default <code>TRUE</code> , ensures a trellis device is active with limited color scheme. Namely <code>background</code> , <code>strip.shingle</code> , and <code>strip.background</code> are each set to "white".
device	Can be one of three values: "single" The default, which will put all graphs on the same device page. "multiple" Relevant only when multiple fitted models are present in the <code>fit</code> object and requested to be plotted. In those cases, a new graphics device is generated to hold each additional plot beyond the first. "ask" Relevant only when multiple fitted models are present in the <code>fit</code> object and requested to be plotted. In these cases, each plot is portrayed as a single-paneled graph, with the <code>ask=TRUE</code> argument specified in par so that the user input confirmation is needed before the graphs are drawn.
...	Additional arguments, depending on the specific method written for the object. See the method-specific documentation for additional details.

Details

The graphs plot the square root of the absolute value of the residuals against the fitted value.

Value

`varianceGraph` returns an invisible `NULL`. The main purpose, of course, is the side effect of graphing to the current device.

Note

Contact <cg@billpikounis.net> for bug reports, questions, concerns, and comments.

Author(s)

Bill Pikounis and John Oleynick

See Also

[varianceGraph.cgOneFactorFit](#)

Examples

```
data(canine)
canine.data <- prepareCGOneFactorData(canine, format="groupcolumns",
                                     analysisname="Canine",
                                     endptname="Prostate Volume",
                                     endptunits=expression(plain(cm)^3),
                                     digits=1, logscale=TRUE, refgrp="CC")

canine.fit <- fit(canine.data)

varianceGraph(canine.fit)

varianceGraph(canine.fit, model="olonly")
```

varianceGraph.cgOneFactorFit

Equal Variance Graphs assessment of cgOneFactorFit object

Description

Graph residuals of a cgOneFactorFit object to assess equal variance assumption

Usage

```
## S4 method for signature 'cgOneFactorFit'
varianceGraph(fit, trend = NULL, cgtheme = TRUE,
              device = "single", ...)
```

Arguments

fit	A fit object of class <code>cgOneFactorFit</code> .
trend	Add a trend line. When set to the default NULL, the addition of a trend line depends on the following: When there is no censored data, a trend will be added with the <code>lowess</code> smoother. If any censored data residuals are present, then at least 7 groups are needed in the fit for a trend line to be added with the <code>VGAM::vgam</code> smoothing spline. Otherwise, group means are estimated and connected. The values of FALSE and TRUE override this. See Details section below.

cgtheme	When set to the default TRUE, ensures a trellis device is active with limited color scheme. Namely background, strip.shingle, and strip.background are each set to "white".
device	Can be one of three values: "single" The default, which will put all graphs on the same device page. For example, when resistant & robust and classical least squares are present and model="both" (the default), a 2 x 1 paneled graph will be created. "multiple" Relevant only when resistant & robust and classical least squares are present and model="both" (the default) or model="extended". In those cases, a new graphics device is generated to hold the resistant & robust version, as a single-paneled graph, and the classical least squares version is on the previous device. If model="extended", then a second new graphics device is generated to hold the unweighted resistant & robust residuals, as another single-paneled graph. "ask" Relevant only when resistant & robust and classical least squares are present and model="both" (the default) or model="extended". In that case, each are portrayed as a single-paneled graph, with the ask=TRUE argument specified in <code>par</code> so that the user input confirmation is needed before the graphs are drawn.
...	Additional arguments. Two are currently valid: model For cgOneFactorFit fit objects that have classical least squares <code>lm()</code> or resistant & robust <code>rlm()</code> fits, the following argument values are possible: "both" Graphs of the residuals from both the ordinary classical least squares and resistant & robust fits are plotted. This is the default when both fits are present in the cgOneFactorFit object specified in the fit argument. If the resistant & robust fit is not available, this value is not relevant. "olsonly" Only a graph of the residuals from the ordinary classical least squares <code>olsfit</code> fit is performed. "rronly" or "rrwtdonly" Only a graph of the weighted residuals from the resistant and robust <code>rrfit</code> fit is plotted. "rrunwtdonly" Only a graph of the unweighted residuals from the resistant and robust <code>rrfit</code> fit is plotted. For other possible cgOneFactorFit fit slots such as accelerated failure time or unequal variance models, the model argument is not relevant, and the appropriate graph will be plotted for these model types.

Details

The graph plots the square root of the absolute value of the residuals against the fitted value. The square root spacing on the y-axis has tick marks in the fitted scale.

The values are automatically jittered to minimize overlapping points. For censored data, left-censored values are shown as a shallow "V", which is actually just a rotated downward "<" sign. Similarly, right-censored values are shown as a deeper "^", which is a rotated upward ">" sign.

For the trend argument, an added trend line when censored data residuals are present needs to be interpreted cautiously. When there are 7 or more groups, a cubic smoothing spline based on


```
                                digits=1)

gmcsfcens.fit <- fit(gmcsfcens.data, type="aft")

varianceGraph(gmcsfcens.fit, trend=TRUE)

varianceGraph(gmcsfcens.fit) ## will yield a warning message why no line
                             ## is graphed

varianceGraph(gmcsfcens.fit, trend=FALSE)
```

Index

- *Topic **datagen**
 - prepare, [67](#)
 - prepareCGOneFactorData, [68](#)
- *Topic **datasets**
 - canine, [7](#)
 - canine.listfmt, [8](#)
 - cgLineColors, [11](#)
 - gmcsfcens, [51](#)
 - gmcsfcens.listfmt, [53](#)
- *Topic **models**
 - fit, [44](#)
 - fit.cgOneFactorData, [45](#)
- *Topic **package**
 - cg-package, [3](#)
- aftfit-class (cgInternalClasses), [10](#)
- blockDiag (cgInternalUtilities), [10](#)
- boxplot, cgOneFactorData-method
 - (boxplot.cgOneFactorData), [5](#)
- boxplot.cgOneFactorData, [5](#)
- boxplot.default, [5](#)
- boxplotcensoreddata
 - (cgInternalUtilities), [10](#)
- boxplotStamp (cgInternalUtilities), [10](#)
- canine, [7](#), [9](#), [72](#)
- canine.listfmt, [7](#), [8](#), [8](#)
- catCharExpr (cgInternalUtilities), [10](#)
- cg (cg-package), [3](#)
- cg-package, [3](#)
- cg.largest.empty (cgInternalUtilities), [10](#)
- cgDevice (cgInternalUtilities), [10](#)
- cgInternalClasses, [10](#)
- cgInternalUtilities, [10](#)
- cgLineColors, [11](#), [61](#), [63](#)
- cgMessage (cgInternalUtilities), [10](#)
- cgOneFactorComparisonsTable, [17](#), [21](#), [23](#), [43](#), [73](#), [74](#), [97](#)
- cgOneFactorComparisonsTable
 - (comparisonsTable.cgOneFactorFit), [26](#)
- cgOneFactorComparisonsTable-class
 - (comparisonsTable.cgOneFactorFit), [26](#)
- cgOneFactorData, [5](#), [6](#), [9](#), [30–32](#), [44](#), [46](#), [47](#), [54](#), [60](#), [64](#), [65](#), [67](#), [95](#), [110](#)
- cgOneFactorData
 - (prepareCGOneFactorData), [68](#)
- cgOneFactorData-class
 - (prepareCGOneFactorData), [68](#)
- cgOneFactorDescriptiveTable, [75](#), [76](#), [98](#)
- cgOneFactorDescriptiveTable
 - (descriptiveTable.cgOneFactorData), [31](#)
- cgOneFactorDescriptiveTable-class
 - (descriptiveTable.cgOneFactorData), [31](#)
- cgOneFactorDownweightedTable, [34](#), [77](#), [99](#)
- cgOneFactorDownweightedTable
 - (downweightedTable.cgOneFactorFit), [35](#)
- cgOneFactorDownweightedTable-class
 - (downweightedTable.cgOneFactorFit), [35](#)
- cgOneFactorFit, [23](#), [26](#), [28](#), [36](#), [37](#), [42](#), [43](#), [45](#), [48–50](#), [55](#), [57–59](#), [78](#), [79](#), [85](#), [94](#), [103–106](#), [108](#)
- cgOneFactorFit (fit.cgOneFactorData), [45](#)
- cgOneFactorFit-class
 - (fit.cgOneFactorData), [45](#)
- cgOneFactorGlobalTest, [79](#), [80](#), [100](#)
- cgOneFactorGlobalTest
 - (globalTest.cgOneFactorFit), [49](#)
- cgOneFactorGlobalTest-class
 - (globalTest.cgOneFactorFit), [49](#)
- cgOneFactorGrpSummaryTable, [81](#), [82](#), [101](#), [102](#)

- cgOneFactorGrpSummaryTable
(grpSummaryTable.cgOneFactorFit),
57
- cgOneFactorGrpSummaryTable-class
(grpSummaryTable.cgOneFactorFit),
57
- cgOneFactorSampleSizeTable, 83, 89, 91,
92, 102, 103
- cgOneFactorSampleSizeTable
(sampleSizeTable.cgOneFactorFit),
93
- cgOneFactorSampleSizeTable-class
(sampleSizeTable.cgOneFactorFit),
93
- cgValidity, 12
- characterOrExpression-class
(cgInternalClasses), 10
- characterOrNULL-class
(cgInternalClasses), 10
- chop.matrix (cgInternalUtilities), 10
- chopZeroes (cgInternalUtilities), 10
- comparisons, 13, 19, 20
- comparisonsGraph, 17, 20
- comparisonsgraph, 18
- comparisonsGraph, cgOneFactorComparisonsTable-method
(comparisonsGraph.cgOneFactorComparisonsTable),
21
- comparisonsGraph.cgOneFactorComparisonsTable,
17, 18, 21
- comparisonsGraphStamp
(cgInternalUtilities), 10
- comparisonsTable, 16, 17, 21, 23
- comparisonsTable, cgOneFactorFit-method
(comparisonsTable.cgOneFactorFit),
26
- comparisonsTable.cgOneFactorFit, 17, 21,
25, 26, 50, 73, 96
- contrastMatrix (cgInternalUtilities), 10
- dataframeMatrixOrNULL-class
(cgInternalClasses), 10
- dataframeOrNULL-class
(cgInternalClasses), 10
- descriptive.censoreddata
(cgInternalUtilities), 10
- descriptiveTable, 30
- descriptiveTable, cgOneFactorData-method
(descriptiveTable.cgOneFactorData),
31
- descriptiveTable.cgOneFactorData, 31,
31, 75, 98
- downweightedTable, 34
- downweightedTable, cgOneFactorFit-method
(downweightedTable.cgOneFactorFit),
35
- downweightedTable.cgOneFactorFit, 34,
35, 35, 77, 99
- errorBarGraph, 37, 40
- errorbargraph, 39, 43
- errorBarGraph, cgOneFactorFit-method
(errorBarGraph.cgOneFactorFit),
41
- errorBarGraph.cgOneFactorFit, 38, 41
- errorBarGraphApproximateStamp
(cgInternalUtilities), 10
- errorBarGraphStamp
(cgInternalUtilities), 10
- factor, 11
- factorInSeq (cgInternalUtilities), 10
- fit, 7, 9, 23, 34, 37, 44, 48, 52, 54, 55, 84, 92
- fit, cgOneFactorData-method
(fit.cgOneFactorData), 45
- fit.cgOneFactorData, 23, 37, 44, 45, 48,
55
- fmtDifference (cgInternalUtilities), 10
- fmtPercent (cgInternalUtilities), 10
- fmtPvalue (cgInternalUtilities), 10
- fmtRatio (cgInternalUtilities), 10
- fmtRatioToPercent
(cgInternalUtilities), 10
- fround (cgInternalUtilities), 10
- geoMean (cgInternalUtilities), 10
- getDotsArgName (cgValidity), 12
- getNumDigits (cgInternalUtilities), 10
- globalTest, 48
- globalTest, cgOneFactorFit-method
(globalTest.cgOneFactorFit), 49
- globalTest.cgOneFactorFit, 48, 49, 79,
100
- gls, 46, 47
- gmcsfcens, 51, 53, 54, 72
- gmcsfcens.listfmt, 52, 53, 53
- gmcsfcens.listfmt1 (gmcsfcens.listfmt),
53

- gmcsfcens.listfmt2 (gmcsfcens.listfmt),
53
- gmcsfcens.listfmt3 (gmcsfcens.listfmt),
53
- graphStampCG (cgInternalUtilities), 10
- grpsummary (cgInternalUtilities), 10
- grpSummaryTable, 55
- grpSummaryTable, cgOneFactorFit-method
(grpSummaryTable.cgOneFactorFit),
57
- grpSummaryTable.cgOneFactorFit, 56, 57,
81, 101
- invisible, 74, 75, 77, 78, 80, 82, 83, 97–102,
104, 106
- isAllEqual (cgInternalUtilities), 10
- jitter, 5, 64, 66
- kmGraph, 60
- kmGraph, cgOneFactorData-method
(kmGraph.cgOneFactorData), 62
- kmGraph.cgOneFactorData, 61, 62
- label, 61, 63
- lm, 27, 46, 73, 78, 80, 81, 83, 106
- logtrans, 71
- lowess, 108
- makeCensored (cgInternalUtilities), 10
- makeContrastVec (cgInternalUtilities),
10
- makeEndptLabel (cgInternalUtilities), 10
- makeTickMarks (cgInternalUtilities), 10
- makeZeroScore (cgInternalUtilities), 10
- minmaxTicks (cgInternalUtilities), 10
- multcompDone (cgInternalUtilities), 10
- multcompInform (cgInternalUtilities), 10
- numericOrNULL-class
(cgInternalClasses), 10
- olsfit-class (cgInternalClasses), 10
- pairwisecompsmatrix
(cgInternalUtilities), 10
- panel.text, 19, 22
- par, 17, 21, 38, 42, 84, 86, 88, 90, 107, 109
- paragraphWrap (cgInternalUtilities), 10
- parsePartialName (cgValidity), 12
- pctToRatio (cgInternalUtilities), 10
- plotGrpNameTicks (cgInternalUtilities),
10
- pointGraph, 64
- pointGraph, cgOneFactorData-method
(pointGraph.cgOneFactorData),
65
- pointGraph.cgOneFactorData, 5, 64, 65, 65
- prepare, 7, 9, 30, 44, 52, 54, 64, 67, 72
- prepareCGOneFactorData, 5–9, 30, 32, 44,
52–54, 60, 63–67, 68, 68, 86, 110
- print, cgOneFactorComparisonsTable-method
(print.cgOneFactorComparisonsTable),
73
- print, cgOneFactorDescriptiveTable-method
(print.cgOneFactorDescriptiveTable),
75
- print, cgOneFactorDownweightedTable-method
(print.cgOneFactorDownweightedTable),
76
- print, cgOneFactorFit-method
(print.cgOneFactorFit), 78
- print, cgOneFactorGlobalTest-method
(print.cgOneFactorGlobalTest),
79
- print, cgOneFactorGrpSummaryTable-method
(print.cgOneFactorGrpSummaryTable),
81
- print, cgOneFactorSampleSizeTable-method
(print.cgOneFactorSampleSizeTable),
82
- print.cgOneFactorComparisonsTable, 28,
73
- print.cgOneFactorDescriptiveTable, 32,
75
- print.cgOneFactorDownweightedTable, 36,
76
- print.cgOneFactorFit, 78
- print.cgOneFactorGlobalTest, 79
- print.cgOneFactorGrpSummaryTable, 59,
81
- print.cgOneFactorSampleSizeTable, 82
- qminmin (cgInternalUtilities), 10
- qqGraph, 84
- qqGraph, cgOneFactorFit-method
(qqGraph.cgOneFactorFit), 85
- qqGraph.cgOneFactorFit, 85, 85
- qqline, 85, 87

- quantile, 32
- rangeExtend (cgInternalUtilities), 10
- reportInvalidArg (cgValidity), 12
- residualgrptrend.helper
 - (cgInternalUtilities), 10
- rlm, 27, 35, 37, 46, 47, 73, 78, 80, 81, 83, 106
- rmTicks (cgInternalUtilities), 10
- rrfit-class (cgInternalClasses), 10
- samplesize (cgInternalUtilities), 10
- samplesizeGraph, 87
- samplesizegraph (cgInternalUtilities), 10
- samplesizeGraph, cgOneFactorSampleSizeTable-method
 - (samplesizeGraph.cgOneFactorSampleSizeTable), 89
- samplesizeGraph.cgOneFactorSampleSizeTable, 88, 89
- samplesizeTable, 91
- samplesizeTable, cgOneFactorFit-method
 - (samplesizeTable.cgOneFactorFit), 93
- samplesizeTable.cgOneFactorFit, 83, 92, 93, 93, 102
- scaleVar (cgInternalUtilities), 10
- seeHelpFile (cgInternalUtilities), 10
- setupAxisTicks (cgInternalUtilities), 10
- setupGrpNameTicks
 - (cgInternalUtilities), 10
- setupLog10AxisTicks
 - (cgInternalUtilities), 10
- show, cgOneFactorComparisonsTable-method
 - (show.cgOneFactorComparisonsTable), 96
- show, cgOneFactorDescriptiveTable-method
 - (show.cgOneFactorDescriptiveTable), 97
- show, cgOneFactorDownweightedTable-method
 - (show.cgOneFactorDownweightedTable), 99
- show, cgOneFactorGlobalTest-method
 - (show.cgOneFactorGlobalTest), 100
- show, cgOneFactorGrpSummaryTable-method
 - (show.cgOneFactorGrpSummaryTable), 101
- show, cgOneFactorSampleSizeTable-method
 - (show.cgOneFactorSampleSizeTable), 102
- show.cgOneFactorComparisonsTable, 96
- show.cgOneFactorDescriptiveTable, 97
- show.cgOneFactorDownweightedTable, 99
- show.cgOneFactorGlobalTest, 100
- show.cgOneFactorGrpSummaryTable, 101
- show.cgOneFactorSampleSizeTable, 102
- showDefault, 15, 24, 27, 30, 32, 34, 36, 48, 50, 56, 58, 92, 94, 97–105
- showObj, 103
- showObj, cgOneFactorFit-method
 - (showObj.cgOneFactorFit), 104
- showObj.cgOneFactorFit, 78, 104
- spline, 71
- stdErr (cgInternalUtilities), 10
- stripMiss (cgInternalUtilities), 10
- summary, cgOneFactorFit-method
 - (summary.cgOneFactorFit), 105
- summary.cgOneFactorFit, 105
- Surv, 70–72
- survfit, 5, 33
- survreg, 46, 47
- trimWhiteSpace (cgInternalUtilities), 10
- tryAgain (cgInternalUtilities), 10
- unpaste (cgInternalUtilities), 10
- unwind (cgInternalUtilities), 10
- unwrap (cgInternalUtilities), 10
- uvfit-class (cgInternalClasses), 10
- validAddConstant (cgValidity), 12
- validAft (cgValidity), 12
- validAlpha (cgValidity), 12
- validArgDigits (cgValidity), 12
- validArgMatch (cgValidity), 12
- validArgModel (cgValidity), 12
- validAtomicVec (cgValidity), 12
- validBoolean (cgValidity), 12
- validCensor (cgValidity), 12
- validCGOneFacGroupColDfr (cgValidity), 12
- validCGOneFacListedDfr (cgValidity), 12
- validCharacter (cgValidity), 12
- validComparisonType (cgValidity), 12
- validCutoffWt (cgValidity), 12
- validDataFormat (cgValidity), 12
- validDenDf (cgValidity), 12
- validDotsArg (cgValidity), 12

`validDotsArgs` (cgValidity), 12
`validEqualLength` (cgValidity), 12
`validErrorDf` (cgValidity), 12
`validEstimates` (cgValidity), 12
`validFitType` (cgValidity), 12
`validGrpNames` (cgValidity), 12
`validList` (cgValidity), 12
`validN` (cgValidity), 12
`validNumeric` (cgValidity), 12
`validNumericOrCensored` (cgValidity), 12
`validPower` (cgValidity), 12
`validZeroScore` (cgValidity), 12
`varianceGraph`, 107
`varianceGraph`, `cgOneFactorFit`-method
 (`varianceGraph.cgOneFactorFit`),
 108
`varianceGraph.cgOneFactorFit`, 108, 108
`vgam`, 110

`xTicksCex` (cgInternalUtilities), 10

`yTicksCex` (cgInternalUtilities), 10