Package

January 11, 2019

Type Package

Title Nice Plots for Data Exploration

Version 0.1.2

Author Zachary Hunter

Maintainer Zachary Hunter <zhunter7@gmail.com>

Description A series of S3 generic plotting functions to aid in data exploration and visualization for those just getting started in R
One of the fundamental problems in biomedical research is how interdependent bioinformatics specialists and wet lab biologist are and yet they have very few ways of effectively collaborating or sharing insights with each other. For a rigorous analysis or custom visualization, a lot of background knowledge is necessary. Not only does one need to learn R but the relevant statistics, specialized tools, then ggplot's grammar of graphics and Bioconductor's datatypes as well. With this project I hope to encourage biologists, physicians and other non-computational researchers to learn enough R to be able to interact with and visualize large datasets for the purposes of hypothesis generation. These plotting functions are S3 generics so that they can be easily adapted for use with complex next generation sequencing datasets from Bioconductor or any large dataset with a complex structure and make it invisible to the end user. The functions work like standard R functions allowing users to practice their basic R skills while being able to perform some basic exploratory analysis of their data.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

Depends dplyr,beeswarm,KernSmooth

Imports boot,tidyr,tibble,RColorBrewer,purrr

Suggests shiny,rgl

Collate 'np_utility.R'
'np_options_processing.R'
'niceThemes.R'
'np_categorical_plot_setup.R'
'np_data_processing.R'
'np_plotting_functions.R'
'np_niceBar.R'
'np_niceBox.R'
'np_niceDots.R'
'np_niceVio.R'

R topics documented:

---

addNicePoints   *Add a datapoint overlay to a box or violin plot*

Description

This function prepares data based on settings from `niceBox`, `niceDots`, or `niceVio` and passes the data on to `drawPoints`.

Usage

```
addNicePoints(prepedData, by, filter = TRUE, sidePlot = F,
  subGroup = F, plotAt, pointHighlights = F, pointMethod = "jitter",
  pointShape = 16, pointSize = 1, width = 1, pointLaneWidth = 0.9,
  plotColors = formatPlotColors(list(1)), drawPoints = T,
  outliers = F, dataCols = 1, swarmOverflow = "random")
```

Arguments

prepedData	list; a list object returned by [prepCategoryWindow](#)
by	factor or dataframe of factors; One or more factors that control how the data is grouped. The first column is the primary grouping factor and the second and thrid columns are used for sub-grouping and highlighting as needed.
filter	logical vector; Used to further filter the data if necissary.
sidePlot	logical; switches the axis to plot horizontally instead of vertically.
subGroup	logical; Should the data be faceted into subgroups within the primary factor levels. Ignored if by is a [factor](#).
plotAt	numeric; A vector of where to draw each set of points
pointHighlights	
	logical; will use additional factors in by to highlight points in the dot plot.
pointMethod	character; method to be used for ploting dots. Can be set to "jitter", "linear", "beeswarm" or "distribution".
pointShape	positive integer; sets pty for plotting data points. Can be a vector to support additional graphical customization.
pointSize	positive integer; sets the cex multiplier for point size.
width	numeric; A multiplier that controls how wide the ploting elements will be. Setting width=1.1 would result in plot elements being 10% wider.
pointLaneWidth	numeric; This controls how far data point dots can move along the categorical axis when plotting. Used for pointMethod options 'jitter', 'beeswarm', and 'distribution'.
plotColors	list; a named list of vectors of colors that set the color options for all NicePlot functions. Names left unspecified will be added and set to default values automatically.
drawPoints	logical; draws a dot plot overlay of the data for each box. Setting this to false causes just the outlier points to be ploted. Used in [niceBox](#).
outliers	positive numeric; number of interquartile ranges (IQR) past the Q1 (25%) and Q3 (75%) cumulative distribution values. Outliers are often defined as $1.5 \times IQR$ and extreme outliers are more than $3 \times IQR$ away from the inner 50% data range.
dataCols	numeric; A number of representing the number of data columns to be plotted. These is a combination of the dimentions of prepedData and/or the number of primary and secondary grouping factors. Used to determine the maximum ploting width for the points.
swarmOverflow	character; How to handle beeswarms that would normally overflow the pointLaneWidth argument. Valid options are "random", "gutter", "wrap", "omit", and "none".

Details

This funciton takes in cleaned data from [prepCategoryWindow](#) and reorganizes to to create a dot plot overlay for a graph. This code is used by both [niceBox](#) and [niceVio](#) and has been moved to an independant funciton to make the code more compact and easier to maintain. This code is also used to draw the outlier dots in a boxplot by setting drawPoints = [FALSE](#).

See Also

[drawPoints](#), [niceBox](#), [niceVio](#), [niceDots](#), [beeswarm](#), [jitter](#), [drawPoints](#)

Examples

```
#Add a beeswarm plot overlay to a boxplot in the iris dataset:
data(iris)
data<-list(data=iris$Sepal.Length)
boxplot(iris$Sepal.Length~iris$Species)
addNicePoints(data,by=iris$Species,pointMethod="beeswarm",plotAt=1:3)

#Add an outlier point to a boxplot:
boxplot(iris$Sepal.Length~iris$Species, outline=FALSE)
addNicePoints(data,by=iris$Species,pointMethod="linear",plotAt=1:3,
    drawPoints=FALSE,outliers=1.5)
```

---

basicTheme   *Nice Plots Theme: Basic*

Description

This is the default theme for nicePlots

Usage

```
basicTheme
```

Format

An object of class npTheme (inherits from list) of length 46.

Details

This default theme has uses transparent solid circles for point overlays with up to 8 colors. Fill and line options as constant.

---

boot95ci   *Calculate a basic bootstrap 95% confidence interval*

Description

Calculates a basic 95% boostrap confidence interval for an aggregator function

Usage

```
boot95ci(x, agg = "mean", upper = FALSE)
```

Arguments

x	numeric vector
agg	character: A string corresponding to the aggregator function to be modeled (eg. [median](), [mean](), etc.)
upper	logical:

Details

This function calculates a bootstrap 95% confidence interval for a aggregator function determined by the agg variable. The number of iterations is hard coded at 1000. The variable upper will cause to return the upper bound of the 95% confidence interval if set to TRUE or the lower bound if set to FALSE.

Value

a number corresponding to the upper or lower bootrap 95% confidence interval of the aggregator function given by agg.

See Also

prepBarData, niceBar, boot, boot.ci

Examples

```
library(boot)
data(iris)
median(iris$Sepal.Width)
#boot95ci(iris$Sepal.Width,agg="median",upper=FALSE)
#boot95ci(iris$Sepal.Width,agg="median",upper=TRUE)
```

---

calcStats	*calculate preliminary statistical significance analysis*

Description

calcStats takes a numeric vector and a factor and runs a preliminary statistical analysis. Output is printed to the screen and the p-value is returned as a character string.

Usage

```
calcStats(x, by, type = c("Wilcox", "Tukey", "T.Test", "ANOVA"),
  verbose = FALSE)
```

Arguments

x	numeric; numeric vector of data points to analyze.
by	factor; factor describing the groups within x to test.
type	character; determines which statistical test should be used. Accepted values are 'wilcox', 't.test', 'ttest', 'anova' and 'tukey'. Values not matching a valid input will produce a warning.
verbose	logical; will print statistical output to the screen if set TRUE. Calculations returned by the function either way.

Details

This is designed to be used in conjunction with data visualization plots to help with data exploration and should not be used for a robust statistical analysis. Normal distribution, variance and other data characteristics are not evaluated and there is no guarantee that the underling test assumptions are met. For two level factors `wilcox.test` or `t.test` is recommended. If the factor has more than two levels then `pairwise.wilcox.test` and `pairwise.t.test` are automatically selected. In this case `anova` and the optional follow-up `TukeyHSD` can also be used. All output it printed to the console and for the two level tests and `anova` the p-value is returned as a text string.

Value

a character string describing the test run and the p-value.

See Also

`wilcox.test`, `pairwise.wilcox.test`, `t.test`, `pairwise.t.test`, `anova`, `TukeyHSD`

Examples

```
data(iris)
pv<-calcStats(iris$Petal.Length,by=iris$Species,type="anova")
boxplot(iris$Petal.Length~iris$Species,main="Petal Length by Species",sub=pv)
```

---

ci	*Confidence interval helper function*

Description

Designed to be used by `boot95ci` to calculate a bootstrap model of an aggregator function given by agg.

Usage

```
ci(x, agg, indices)
```

Arguments

x	numeric vector
agg	character: A string corresponding to the aggregator function to be modeled (eg. `median`, `mean`, etc.)
indices	numeric vector: Indicies are passed to calculated individual bootstaps of x

Details

This is a convenience function internal to `NicePlots` and is not exported. The variable agg should be a string corresponding an aggregator function such as `mean` or `median`. The indicies designed to passed from the `boot` function from the package boot and are used to determine calculated the different bootstrap iterations of the agg functions on x.

Value

a number corresponding to a bootstrap iteration of the aggregator function given by agg

See Also

[boot](), [boot95ci]()

Examples

```
#Calculates the median of Sepal.Length from the iris data set
data(iris)
#ci(iris$Sepal.Length,"median",seq(1,length(iris$Sepal.Length)))
```

---

dataFlightCheck   *Check data formating for the NicePlots package*

Description

Formats and cleans data prior to setting up the plotting enviroment

Usage

```
dataFlightCheck(data, by, flipFacts, na.rm = FALSE)
```

Arguments

data	vector or dataframe; data to be plotted
by	factor or dataframe; factors to be used to format data
flipFacts	logical; If a dataframe is used for plotting data input, this will covert the data to a vector with the dataframe colums trasfered a factor in the second column of the by input.
na.rm	logical; Removes all data and factor rows were NA is presenst.

Details

This funciton makes sure the data input is a numeric vector or a data frame of numeric vectors. It will also check to make sure by is a factor or a dataframe of factors. If requested, it will also remove missing data and rearrage numeric dataframe inputs.

Value

A named list with d=data and b=by.

Examples

```
todo<-1
```

---

drawBar   *drawBar*

Description

Add a barplot with options error pars to the active ploting enviroment

Usage

```
drawBar(x, plotColors, errorBars = FALSE, errorCap = "ball",
  errorLineType = 1, width = 0.5, sidePlot = FALSE,
  stacked = FALSE, capSize = 2, lineWidth = 1)
```

Arguments

x	dataframe; number of interquartile ranges (IQR) past the Q1 (25%) and Q3 (75%) cumulative distribution values. Outliers are often defined as $1.5 \times IQR$ and extreme outliers are more than $3 \times IQR$ away from the inner 50% data range.
plotColors	list; a named list of vectors of colors that set the color options for all NicePlot functions. Names left unspecified will be added and set to default values automatically.
errorBars	Logical; Should error bars be drawn. Defaults to true but is ignored if stack=TRUE.
errorCap	character; Determines the style for the ends of the error bars. Valid options are ball, bar or none.
errorLineType	numeric; Sets lty line type for drawing the error bars.
width	numeric; cex like scaling factor controlling the width of the bars.
sidePlot	logical; Plots bar hight on the x axis if set to TRUE.
stacked	logical; draws a stacked barplot if set to TRUE.
capSize	numeric; cex like scaling value the controls the size of the caps on the error bars.
lineWidth	numeric; Sets the lwd options for controling line plotting thickness for the bar plot.

Details

This function draws a series of bars based on a dataframe. The expected columns include yt (locaion top of the bar), yb or bottom of the bar, at indicating where the bar should be drawn, Group which is a unique ID per row, fact which contains an optional stacking factor UpperError for the top of the error bar and LowerError for the location of the bottom of the error bar. The construction of the dataframe is handled automatically from input data by niceBar.

See Also

barplot, niceBar, errorBars

Examples

```
data(iris)
data<-iris  %>% group_by(Species) %>%
    summarize(yt=mean(Sepal.Length),yb=0,UpperError=sd(Sepal.Length),
    LowerError=sd(Sepal.Length)) %>%
    ungroup() %>% select(yt,yb,UpperError,LowerError,Group=Species) %>%
    bind_cols(at=1:3,fact=1:3)
plot(type="n",xlim=c(0,4),ylim=c(0,max(iris$Sepal.Length)),-1,xaxt="n")
drawBar(data,plotColors=list())
```

---

drawBoxPlot	*draw a custom box and whisker plot*

Description

takes a date frame with columns labeled 'at', 'q1', 'q3', 'min', 'max', 'median' and 'width' to draw a series of boxplots.

Usage

```
drawBoxPlot(x, col = "black", fill = NULL, drawBox = T,
  drawDot = F, whiskerLty = 2, side = FALSE, lWidth = 1,
  capWidth = 0.25)
```

Arguments

x	named list or data frame; xat, xq2, x$q4, x$median, xmin, xmax and x$width must all be defined as numeric vectors in a named list or data.frame object.
col	character; color vector that controls the line color.
fill	character; color vector that determines the interior color of the box.
drawBox	logical; draws the box and whiskers if set to TRUE. The median line will be drawn regardless.
drawDot	logical; draws a circle at the center of the median bar if set to TRUE.
whiskerLty	positive integer; sets the line type or lty option for plotting the wiskers.
side	logical; if set to TRUE, the box plots will be drawn horizontally.
lWidth	positive integer; corresponds to lwd line width setting in base R.
capWidth	numeric; size of the error bar cap relative to the box width.

Details

The input data frame x should include columns labels named 'at','q1',and 'q3', 'median', 'min', 'max' and 'width' in any order. Each row will draw a box and whisker plot. The columns 'q1' and 'q3' refer to the 25% and 75% cumulative distribution values that bound the interquartile range. If side=TRUE then the x and y axises are swapped to support horizontal plotting. The box and whiskers can be suppressed leaving only the median line and the optional center marker if so desired.

See Also

boxplot, niceBox

Examples

```
data(iris)
iData<-iris %>% group_by(Species) %>%
   summarize(median=median(Sepal.Length),min=min(Sepal.Length),max=max(Sepal.Length),
   q1=quantile(Sepal.Length)[2],q3=quantile(Sepal.Length)[4]) %>%
   bind_cols(at=c(1:3),width=c(.2,.3,.4))
plot(1,1,type="n",xlim=c(0,4),ylim=c(0,9))
drawBoxPlot(iData)
```

---

drawPoints	*draw dots for a dot plot*

Description

takes a data frame of locations, values and an optional subgrouping factor and adds the data points to the active plot

Usage

```
drawPoints(x, type = "jitter", col = "black", size = 1, shape = 1,
  highlight = FALSE, width = 0.2, sidePlot = FALSE,
  swarmOverflow = "random")
```

Arguments

x	named list or data frame; xat, xdata and x$pfact (optional) should all be defined. These vectors are used to place the the point on the chart and determine the point level grouping (highlighting)
type	character; determines how the points are arranged. Options are 'jitter', 'linear', 'beeswarm' and 'distribution'.
col	character; vector of color names for plotting points. If length is greater than one it will be used for subgroups or will iterate over the groups.
size	numeric; vector of cex values for point size. If length is greater than one it will be used for subgroups or will iterate over the groups.
shape	numeric; vector determining point shapes (pch). If length is greater than one it will be used for subgroups or will iterate over the groups.
highlight	logical; Should the point highlighting option be turned on (assumes that pfact is defined).
width	numeric; determines how far points can deviate from the center category label for type options other than 'linear'.
sidePlot	logical; plots dots for a horizontal rather than vertical axis.
swarmOverflow	character; How to handle beeswarms that would normally overflow the width argument. Valid options are "random", "gutter", "wrap", "omit", and "none".

Details

This function adds data points to a chart. These can be organized exactly as specified (linear), as a jitter cloud (jitter), as a waterfall plot (distribution) or as a swarm (beeswarm). A factor labeled pfact can be included in x and used to highlight individual data points by setting subGroup=TRUE. All graphic customization options can given as vectors and will be iterated over during plotting. Note that the size/cex option can not be used to highlight pfact levels in a beeswarm plot and only the first element of the vector will be used.

See Also

points, stripchart, beeswarm

Examples

```
data(iris)
boxplot(iris$Sepal.Length~iris$Species,ylab="Sepal Length")
iData<-data.frame(at=as.numeric(iris$Species),data=iris$Sepal.Length)
drawPoints(iData,type="jitter",col=c("red","blue","purple"))
```

---

drawViolinPlot   *draw a violin plot*

Description

Produce a violin plot with optional box plot and strip plot overlays

Usage

```
drawViolinPlot(x, groups, at = seq(1, length(levels(groups))),
  h = NULL, plotColors = basicTheme$plotColors, sidePlot = FALSE,
  borderCol = plotColors$lines, borderWidth = 1,
  fill = plotColors$fill, width = 1, trimViolins = TRUE,
  samplePoints = NULL)
```

Arguments

x	numeric; A vector of numeric values that will be subset and formated by the factor(s) in by.
groups	factor; A factor used to subset x to draw the violins.
at	numeric; a numeric vector of where each factor level should be plotted
h	numeric; Bandwidth for the kernal desnisty estimates. Will cycle over values if multiple bandwidths are given
plotColors	list; a named list of vectors of colors that set the color options for all NicePlot functions.
sidePlot	logical; If TRUE, the x and y ploting axis are swapped
borderCol	R color string; Color of the border of the violins
borderWidth	numeric; Thickness of the violin borders (lwd)
fill	R color string; Color of the interior of the violins

width	numeric; Relative width of the violins. A value of 1 will cause the violins to cover thier entire lane and potentially just touch.
trimViolins	logical; Should the violins be truncated at the edges of the data range.
samplePoints	integer; The number of points used to draw each side of the violin. This is generally obtained from theme$curvePoints.

Details

This uses [bkde](#) from the KernSmooth package to calculated kernal desnisty estimates and estimate the optimal bandwidth h setting. This data is then used to draw a violin plot with an optional boxplot drawn as an overlay to better charactarize the quartile distributon. Likewise, a strip chart of individual data points can be added on top of these two plots to full charactarize the data distribution. This uses [niceBox](#) to handle the box plot and strip chart overlays.

See Also

[niceVio](#), [bkde](#), [dpik](#)

Examples

```
todo<-1
```

| errorBars | *draw custom error bars* |

Description

Draws error bars with an optional cap at one end

Usage

```
errorBars(x, capType = c("none", "bar", "ball"), capSize = NULL,
  side = FALSE, col = "black", lType = 1, width = 1)
```

Arguments

x	named list or data frame; x$start, x$stop and x$at must all be defined as numeric vectors in a named list or data.frame object. In the case of a data frame, each row returns a single error bar.
capType	character; can be set to 'none', 'bar', 'ball'. If set to 'bar' or ball, a round point or a line segment will be used to cap the end of the error bar.
capSize	numeric; capSize is the distance that the cap extends away from the error bar. Set to [NULL](#) to suppress the cap regardless of the capType setting.
side	logical; if set to true, the error bars will be drawn horizontally.
col	color; a vector of line colors.
lType	positive integer; corresponds to lty line type in base R.
width	positive numeric; corresponds to lwd line width setting in base R.\#'

Details

The input data frame x should have columns labels 'at','start',and 'stop' with at determining the x-axis location and start and stop indicating the position of the segment on the y-axis. If side=TRUE then the x and y axises are swapped to support horizontal plotting. Each row of the data frame will produce one bar and an optional cap can be drawn at the 'stop' location.

Examples

```
data(iris)
iData<-iris %>% group_by(Species) %>%
    summarize(Average=mean(Sepal.Length),SD=sd(Sepal.Length))
barplot(iData$Average,ylim=c(0,10),names=levels(iris$Species),ylab="sepal length")
loc<-c(.7,1.9,3.1)
top<-iData$SD*2+iData$Average
bottom<-iData$SD*-2+iData$Average
errorBars(data.frame(at=loc,start=iData$Average,stop=top),capType="ball",capSize=2)
errorBars(data.frame(at=loc,start=iData$Average,stop=bottom),capType="ball",capSize=2)
```

---

facetSpacing   *Generate plotting locations for subgrouping data*

Description

facetSpacing generates a vector for the at= specification in functions for data sub-grouping

Usage

```
facetSpacing(subGroup, labels)
```

Arguments

subGroup   positive integer; number of levels in the subgrouping factor

labels   positive integer; number of levels in the primary factor

Details

facetSpacing takes the number factor levels from the primary and secodary grouping factors to generate a vector of positions for plotting subgrouped data for the nicePlots package. The spacing assumes that each primary factor levels is plot on positive integers 1, 2, 3 etc. For a primary factor at position i with f subgroup levels, the subgrouping comes from generating equally spaced intervals starting at $i - \frac{1}{2} + \frac{1}{f+1}$ and ending at $i + \frac{1}{2} - \frac{1}{f+1}$. Simply put:

$$Spacing = \frac{1}{NSubGroups - 1}$$

Value

a numeric vector of where to plot the subgrouped data. Can be supplied to that at= option in plotting functions

See Also

prepCategoryWindow

Examples

```
boxplot(CNA$BM~ CNA$Status,border="white")
stripchart(CNA$BM~factor(paste0(CNA$Status,CNA$Sex)),add=T,at=facetSpacing(2,2))
```

---

formatPlotColors   *format a NicePlots color list*

Description

To simplify code and user options, any color option not set by the user is added to the list and set to the default value.

Usage

```
formatPlotColors(plotColors, theme = NA)
```

Arguments

plotColors   list; a named list of vectors of colors that set the color options for all NicePlot functions. Names left unspecified will be added and set to default values automatically.

theme   list; A NicePlots plotColor list from a theme.

Details

The NicePlots plotColors object is a list of named color values/vectors. The options NicePlots colors include bg (background color), marginBg (color of area surrounding the plot), guides (guide lines for major tick-marks), minorGuides (guide lines for minor tick-marks) lines (lines for box/bar plots etc.), points (plotting data points), fill (fill for box/bar plots etc.), axis (axis colors), majorTick (major tick-mark color), minorTick (minor tick-mark color), labels (label colors), subGroupLabels (subgroup label colors), rectCol (inner quartile range box used in niceVio), and medianMarkerCol (the median value marker used in niceVio). Any option not set be the user will be added to the list and set to the default in order to insure compatibility with downstream NicePlot functions. If a theme is given, any option not set by the user will be set by the theme.

Value

a formated NicePlots color list.

Examples

```
myCols<-list(bg="lightgrey",fill=c("red","green","blue"),lines="darkgrey")
myCols<-formatPlotColors(myCols)
print(myCols)
```

---

makeColorMatrix *Create a matrix of increasingly transparent colors*

Description

makeColorMatrix is a convenience function for plotting with transparent colors.

Usage

```
makeColorMatrix()
```

Details

This function take no arguments, but generates rows corresponding to red, blue, green, gray, purple and gold with increasing transparency moving from left to right across the columns.

Value

A 6 x 5 matrix of colors.

See Also

[rainbow](), [col2rgb](), [rgb]().

Examples

```
plot(1,1,col="white",xlim=c(0,10),ylim=c(0,10))
for(n in 1:6){rect(0:4,rep(8-n,5),1:5,rep(9-n,5),col=as.matrix(makeColorMatrix())[n,])}

#An example how it can be used in practice:
myData<-rnorm(600)
fact<-factor(c(rep("a",100),rep("b",100),rep("c",100),rep("d",100),rep("e",100),rep("f",100)))
plot(myData,col=makeColorMatrix()[fact,3])
```

---

makeLogTicks *format a log scale axis*

Description

Generates the location and labels for the major tick marks for a given log base transformation along with optional minor tick mark location.

Usage

```
makeLogTicks(dataRange, minorCount = 10, logScale = 2,
  axisText = c(NULL, NULL), expLabels = TRUE)
```

Arguments

dataRange   numeric; a numeric vector with the min and max values for the data set prior to log transformation.

minorCount   positive integer; the number of minor tick marks to be drawn between each major tick.

logScale   numeric; the logarithm base to use for the log scale transformation.

axisText   character; a length two character vector containing text to be prepend or append to the major tick labels, respectively.

expLabels   logical; if set to TRUE, the major labels will written as $logbase^x$. Otherwise the labels will correspond to the non-transformed values at that point.

Details

Base R does not have great visual queues to indicate when data is being plotted in log scale. This is a simple function takes the min and max of the untransformed data and uses axisTicks from base R to determine the location of the major tick marks in the new scale. To better indicate that the graph is on a log scale, the major tick-marks are labeled in the untransformed values or expressed in as $logScale^x$ when expLabels=TRUE. The minor tick marks are drawn equidistant from each other between the major tick marks in the untransformed scale giving them shrinking appearance when rendered in log scale coordinates. This can help helps with the interpretation of data within the log scale and adds another visual indication that the data has been transformed. The value of minorCount gives number of minor ticks to be drawn between each pair of major tick-marks. axisText allows for symbols or units such as ' It is worth stressing again that the input values to dataRange are assumed to be raw values prior to log transformation. If log transformed values are given, the axis will be drawn correctly.

Value

a list with the following elements: major tick marks locations [[1]], major tick labels [[2]], minor tick mark locations [[3]].

See Also

axisTicks, axis, prepCategoryWindow

Examples

```
plot(1:10,log(1:10,2),yaxt="n",ylab="")
majorTicks<-makeLogTicks(c(0,10),minorCount= 4,logScale=2, axisText=c("","mg"), expLabels=TRUE)
axis(side=2,lab=majorTicks[[2]],at=majorTicks[[1]],las=2)
axis(side = 2, at = majorTicks[[3]], labels = FALSE, tcl = -0.2)
```

---

makeNiceLegend   *Draw a nice plot legened*

Description

Draws a customizable legend in the margins based on factor levels.

Usage

```
makeNiceLegend(labels, title = "Legend", fontCol = "black",
  border = NULL, lineCol = NA, bg = NA, col = makeColorMatrix()[,
  3], shape = "rect", size = 0.66, spacing = 0.2)
```

Arguments

`labels`	character vector; The names of the levels decribed in the legend. Typically factor levels.
`title`	character; The title of the legend. This defaults to "Legend" if unspecificed.
`fontCol`	R color; Color of the legend text.
`border`	R color; The color of the rectanglar border surrounding the legend. Defaults to NULL which supresses this feature
`lineCol`	R color; The color of the line colors for the color key. Optional. Defaults to NA.
`bg`	R color; Sets the background color for the legend aread. Note that this can be distinct the the margin background.
`col`	R color vector; A vector of colors determining the color of the color code boxes.
`shape`	character; Determins if the color code is rectangles or circles. Valid ptions are "rect", "rectangle", "circ", or "circle". Not there is no funcitonal difference between the synonyms.
`size`	numeric; Sets the legend font cex sizing.
`spacing`	numeric; Determins the total amount of padding (sum of upper and lower padding) surrounding each line. in the legend in units of font line hight.

Details

This functions works with plot enviroment initializing functions such as `prepCategoryWindow` to expand the right margin to accomodate a figure legend.

See Also

`legend`, `prepCategoryWindow`, `niceBox`, `niceDots`, `niceBar`, `niceVio`

Examples

```
ToDo<-1
```

---

niceBar	*draw a bar plot*

Description

Aggregates data from a numeric vector or dataframe using up to three factors to draw a barplot with optional error bars.

Usage

```
niceBar(x, by = NULL, groupNames = NULL, aggFun = c("mean", "median",
    "none"), errFun = c("sd", "se", "range"), theme = basicTheme,
    legend = FALSE, stack = FALSE, main = NULL, sub = NULL,
    ylab = NULL, minorTick = FALSE, guides = TRUE, outliers = FALSE,
    width = NULL, errorMultiple = 1, plotColors = list(bg = "open",
    fill = setAlpha("grey", 0.8)), logScale = FALSE, trim = FALSE,
    axisText = c(NULL, NULL), showCalc = FALSE, calcType = "none",
    yLim = NULL, rotateLabels = FALSE, rotateY = TRUE, add = FALSE,
    minorGuides = NULL, extendTicks = TRUE, subGroup = FALSE,
    subGroupLabels = NULL, expLabels = FALSE, sidePlot = FALSE,
    errorBars = TRUE, errorCap = "ball", errorLineType = 1,
    capWidth = 1.2, lWidth = 1.5, na.rm = FALSE, flipFacts = FALSE,
    verbose = FALSE, logAdjustment = 1, ...)
```

Arguments

x	numeric vector or data frame; The input to `prepCategoryWindow` can be a numeric vector a data frame of numeric vectors.
by	factor or data frame of factors; used as the primary grouping factor and the factor levels will be used as group names if `groupNames` is not specified. If `by` is a data frame and `subGroup=TRUE`, the second column is assumed to be a secondary grouping factor, breaking out the data into sub-categories within each major group determined by the levels of the first column.
groupNames	character vector; overrides the factor levels of `by` to label the groups
aggFun	character; Determines how the data is summarized by factor level. Valid options are `mean`, `median` or `none`.
errFun	character; How the data spread is charactarized by the error bars. Valid options are `sd` (standard deviation), `se` (standard error of the mean) or `range`.
theme	list object; Themes are are an optional way of storing graphical preset options that are compatible with all nicePlot graphing functions.
legend	logical/character; if not equal to `FALSE` with cause a legend to be drawn in the margins. If set to a character string instead of a logical value, the string will be used as the legend title insteas of the factor column name from `by`.
stack	logical; Should one of the factors in `by` be used make a stacked bar plot. Note that this sort of analysis is nonsensical for many data sets.
main	character; title for the graph which is supplied to the `main` argument.
sub	character; subtitle for the graph which is supplied to the `sub` argument. If `NULL` and `showCalc=TRUE` it will be used to display the output form `calcStats`.
ylab	character; y-axis label.
minorTick	positive integer; number of minor tick-marks to draw between each pair of major ticks-marks.
guides	logical; will draw guidelines at the major tick-marks if set to `TRUE`. Color of the guidelines is determined by `plotColors$guides`.
outliers	positive numeric; number of interquartile ranges (IQR) past the Q1 (25%) and Q3 (75%) cumulative distribution values. Outliers are often defined as $1.5 \times IQR$ and extreme outliers are more than $3 \times IQR$ away from the inner 50% data range.

width	numeric; cex-like scaling factor controlling the width of the bars.
errorMultiple	numeric; How many standard errors/deviations should be represented by the error bars.
plotColors	list; a named list of vectors of colors that set the color options for all NicePlot functions. Names left unspecified will be added and set to default values automatically.
logScale	positive numeric; the base for the for log scale data transformation calculated as log(x+1,logScale).
trim	positive numeric; passed to threshold argument of [quantileTrim](#) if any data points are so extreme that they should be removed before plotting and downstream analysis. Set to [FALSE](#) to disable.
axisText	character; a length two character vector containing text to be prepended or appended to the major tick labels, respectively.
showCalc	logical; if a p-value can be easily calculated for your data, it will be displayed using the sub annotation setting.
calcType	character; should match one of 'none', 'wilcox', 'Tukey','t.test','anova' which will determine which, if any statistical test should be performed on the data.
yLim	numeric vector; manually set the limits of the plotting area (eg. yLim=c(min,max)). Used to format the y-axis by default but will modify the x-axis if side=[TRUE](#).
rotateLabels	logical; sets las=2 for the x-axis category labels. Will affect y-axis if side=[TRUE](#). Note that this may not work well if long names or with subgrouped data.
rotateY	logical; sets las=2 for the y-axis major tick-mark labels. Will affect x-axis if side=[TRUE](#).
add	logical; causes plotting to be added to the existing plot rather the start a new one.
minorGuides	logical; draws guidelines at minor tick-marks
extendTicks	logical; extends minor tick-marks past the first and last major tick to the edge of the graph provided there is enough room. Works for both log-scale and regular settings.
subGroup	logical; use additional column in by to group the data within each level of the major factor.
subGroupLabels	character vector; sets the labels used for the subGroup factor. Defaults to the levels of the factor.
expLabels	logical; prints the major tick labels is $logScale^x$ instead of the raw value
sidePlot	logical; switches the axis to plot horizontally instead of vertically.
errorBars	Logical; Should error bars be drawn. Defaults to true but is ignored if stack=[TRUE](#).
errorCap	character; Determines the style for the ends of the error bars. Valid options are ball, bar or none.
errorLineType	numeric; Sets lty line type for drawing the error bars.
capWidth	numeric; Controls the cex like scaling of the ball or width of the cap if they are drawn at the end of the error bars for the bar plot.
lWidth	numeric; Line width (lwd) for drawing the bar plot.
na.rm	logical; Should NAs be removed from the data set? Both data input and the factor input from by with be checked.
flipFacts	logical; When a dataframe of values is given, column names are used as a secondary grouping factor by default. Setting flipFacts=[TRUE](#) makes the column names the primary factor and by the secondary factor.

verbose	logical; Prints summary and p-value calculations to the screen. All data is silently by the function returned either way.
logAdjustment	= numeric; This number is added to the input data prior to log transformation. Default value is 1.
...	additional options for S3 method variants.

Details

This bar plot function allows for standard barplot features but with error bars, the ability summaryize dataframes into bar plots with median/mean values, sort by bar hight for waterfall plots, color bars based on interquartile outlier detection and more. Barplots can be clustered by a secondary factor or if a dataframe is passed to x the input values of multiple measurments (dataframe columns) can be clustered together by the primary factor. As with [niceBox](#), [niceDots](#) and [niceVio](#), by can be a factor or a dataframe factors for forming subgroups.

For most data this would be nonsensical but if you data is say store profits by goods by region one could group by region (first)

See Also

[vioplot](#), [boxplot](#), [niceBox](#), [beeswarm](#), [prepCategoryWindow](#)

Examples

```
data(mtcars)
Groups<-data.frame(Cyl=factor(mtcars$cyl),Gear=factor(mtcars$gear))
niceBar(mtcars$mpg,by=Groups,subGroup=TRUE,yLim=c(0,45),main="MpG by Cylinders and Gear")
```

---

niceBox   *draw a box plot*

Description

draws a box plot with optional scatter plot overlays, subgrouping options and log scale support.

Usage

```
niceBox(x, by = NULL, groupNames = NULL, main = NULL, sub = NULL,
  ylab = NULL, theme = basicTheme, minorTick = FALSE,
  guides = TRUE, outliers = 1.5, pointSize = 1, width = 1,
  pointShape = 16, plotColors = list(bg = "open"), logScale = FALSE,
  trim = FALSE, pointMethod = "jitter", axisText = c(NULL, NULL),
  showCalc = FALSE, calcType = "none", drawBox = TRUE, yLim = NULL,
  rotateLabels = FALSE, rotateY = FALSE, add = FALSE,
  minorGuides = NULL, extendTicks = TRUE, subGroup = FALSE,
  subGroupLabels = NULL, expLabels = TRUE, sidePlot = FALSE,
  drawPoints = TRUE, pointHighlights = FALSE, pointLaneWidth = 0.7,
  flipFacts = FALSE, na.rm = FALSE, verbose = FALSE,
  legend = FALSE, logAdjustment = 1, ...)
```

Arguments

x	numeric vector or data frame; The input to prepCategoryWindow can be a numeric vector a data frame of numeric vectors.
by	factor or data frame of factors; used as the primary grouping factor and the factor levels will be used as group names if groupNames is not specified. If by is a data frame and subGroup=TRUE, the second column is assumed to be a secondary grouping factor, breaking out the data into sub-categories within each major group determined by the levels of the first column.
groupNames	character vector; overrides the factor levels of by to label the groups
main	character; title for the graph which is supplied to the main argument.
sub	character; subtitle for the graph which is supplied to the sub argument. If NULL and showCalc=TRUE it will be used to display the output form calcStats.
ylab	character; y-axis label.
theme	list object; Themes are are an optional way of storing graphical preset options that are compatible with all nicePlot graphing functions.
minorTick	positive integer; number of minor tick-marks to draw between each pair of major ticks-marks.
guides	logical; will draw guidelines at the major tick-marks if set to TRUE. Color of the guidelines is determined by plotColors$guides.
outliers	positive numeric; number of interquartile ranges (IQR) past the Q1 (25%) and Q3 (75%) cumulative distribution values. Outliers are often defined as $1.5 \times IQR$ and extreme outliers are more than $3 \times IQR$ away from the inner 50% data range.
pointSize	positive integer; sets the cex multiplier for point size.
width	numeric; scaling factor controlling the width of the boxes.
pointShape	positive integer; sets pty for plotting data points. Can be a vector to support additional graphical customization.
plotColors	list; a named list of vectors of colors that set the color options for all NicePlot functions. Names left unspecified will be added and set to default values automatically.
logScale	positive numeric; the base for the for log scale data transformation calculated as log(x+1,logScale).
trim	positive numeric; passed to threshold argument of quantileTrim if any data points are so extreme that they should be removed before plotting and downstream analysis. Set to FALSE to disable.
pointMethod	character; method to be used for ploting dots. Can be set to "jitter", "linear", "beeswarm" or "distribution".
axisText	character; a length two character vector containing text to be prepended or appended to the major tick labels, respectively.
showCalc	logical; if a p-value can be easily calculated for your data, it will be displayed using the sub annotation setting.
calcType	character; should match one of 'none', 'wilcox', 'Tukey','t.test','anova' which will determine which, if any statistical test should be performed on the data.
drawBox	logical; should the boxes be drawn. The median bar will be drawn regardless.
yLim	numeric vector; manually set the limits of the plotting area (eg. yLim=c(min,max)). Used to format the y-axis by default but will modify the x-axis if side=TRUE.

rotateLabels	logical; sets las=2 for the x-axis category labels. Will affect y-axis if side=TRUE. Note that this may not work well if long names or with subgrouped data.
rotateY	logical; sets las=2 for the y-axis major tick-mark labels. Will affect x-axis if side=TRUE.
add	logical; causes plotting to be added to the existing plot rather the start a new one.
minorGuides	logical; draws guidelines at minor tick-marks
extendTicks	logical; extends minor tick-marks past the first and last major tick to the edge of the graph provided there is enough room. Works for both log-scale and regular settings.
subGroup	logical; use additional column in by to group the data within each level of the major factor.
subGroupLabels	character vector; sets the labels used for the subGroup factor. Defaults to the levels of the factor.
expLabels	logical; prints the major tick labels is $logScale^x$ instead of the raw value
sidePlot	logical; switches the axis to plot horizontally instead of vertically.
drawPoints	logical; draws a dot plot overlay of the data for each box.
pointHighlights	logical; will use additional factors in by to highlight points in the dot plot
pointLaneWidth	numeric; This controls how far data point dots can move along the categorical axis when plotting. Used for pointMethod options 'jitter', 'beeswarm', and 'distribution'.
flipFacts	logical; When a dataframe of values is given, column names are used as a secondary grouping factor by default. Setting flipFacts=TRUE makes the column names the primary factor and by the secondary factor.
na.rm	logical; Should NAs be removed from the data set? Both data input and the factor input from by with be checked.
verbose	logical; Prints summary and p-value calculations to the screen. All data is silently by the function returned either way.
legend	logical/character; Draw a legend in the plot margins. If a character string is given it will overide the factor name default for the legend title.
logAdjustment	= numeric; This number is added to the input data prior to log transformation. Default value is 1.
...	additional options for S3 method variants

Details

This box plot function offers extensive log scale support, outlier detection, data point overlay options, data subsetting with a secondary factor, and data point highlighting with a tertiary factor. The complicated part of using this function is handling its many options. A wrapper function to set up and run it with preset options may be a good idea if you are using it along. The function niceDots is an example of this. Briefly put, the by argument can be a data frame of factors and the function will work through the columns in order as needed. If x is a numeric vector, then by should be a factor to group it into categories. If by is a data frame of factors and subGroup=TRUE, then the first column for by is used as the grouping factor and the second column is used as the sub-grouping factor. If pointHighlights==TRUE, and subGroup=TRUE, the the third column of by is used to highlight points data point overlay (assuming drawPoints=TRUE). If subGroup=FALSE and subGroup=TRUE, then the second column of by is used to control the point highlighting. If x itself is a data frame of numeric vectors, subGroup is automatically set to false and each column of

x is plotted like a sub-group and grouped by the first column of by. Data point highlighting with pointHighlights=TRUE can still be used when x is a data frame and the highlighting factor will be drawn from the second column of by. Please note that the p-values can not always be calculated and are for general exploratory use only. More careful analysis is necessary to determine statistical significance. This function is as S3 generic and can be extended to provide class specific functionality. To further facilitate data exploration, outputs from statistical testing and data set summaries are printed to the console.

See Also

boxplot, beeswarm, quantileTrim, prepCategoryWindow

Examples

```
data(iris)
mCols<-makeColorMatrix()
myCols<-list(fill=c(mCols[1,3],mCols[2,3],mCols[3,3]),lines="darkblue")
Lab<-"Sepal Length"
niceBox(iris$Sepal.Length,iris$Species,minorTick=4,showCalc=TRUE,
    calcType="anova",ylab=Lab,main="Sepal Length by Species",plotColors=myCols)


plot(density(iris$Petal.Length))
lengthFact<-factor(iris$Petal.Length>2.82,labels=c("short","long"))


Title<-"Sepal Length by Species and Petal Length"
factorFrame<-data.frame(Species=iris$Species,PetalLength=lengthFact)
niceBox(iris$Sepal.Length, by=factorFrame, minorTick=4,subGroup=TRUE,
    ylab=Lab,main=Title,plotColors=myCols)
```

---

niceDots   *draw a dot plot*

Description

draws a categorical dot plot with optional data highlighting and log scale support.

Usage

```
niceDots(x, by = NULL, groupNames = NULL, main = NULL, sub = NULL,
  ylab = NULL, minorTick = FALSE, theme = basicTheme,
  guides = TRUE, outliers = 1.5, pointSize = 1, width = 1,
  pointShape = 1, plotColors = list(bg = "open"), logScale = FALSE,
  trim = FALSE, pointMethod = NULL, axisText = c(NULL, NULL),
  showCalc = FALSE, calcType = "none", yLim = NULL,
  rotateLabels = FALSE, rotateY = FALSE, add = FALSE,
  minorGuides = NULL, extendTicks = TRUE, subGroup = FALSE,
  subGroupLabels = NULL, expLabels = TRUE, sidePlot = FALSE,
  pointHighlights = FALSE, pointLaneWidth = 1, na.rm = FALSE,
  flipFacts = FALSE, verbose = FALSE, legend = FALSE,
  logAdjustment = 1, ...)
```

Arguments

x	numeric vector or data frame; The input to `prepCategoryWindow` can be a numeric vector a data frame of numeric vectors.
by	factor or data frame of factors; used as the primary grouping factor and the factor levels will be used as group names if `groupNames` is not specified. If `by` is a data frame and `subGroup=`TRUE, the second column is assumed to be a secondary grouping factor, breaking out the data into sub-categories within each major group determined by the levels of the first column.
groupNames	character vector; overrides the factor levels of `by` to label the groups
main	character; title for the graph which is supplied to the `main` argument.
sub	character; subtitle for the graph which is supplied to the `sub` argument. If NULL and `showCalc=`TRUE it will be used to display the output form `calcStats`.
ylab	character; y-axis label.
minorTick	positive integer; number of minor tick-marks to draw between each pair of major ticks-marks.
theme	list object; Themes are are an optional way of storing graphical preset options that are compatible with all nicePlot graphing functions.
guides	logical; will draw guidelines at the major tick-marks if set to TRUE. Color of the guidelines is determined by `plotColors$guides`.
outliers	positive numeric; number of interquartile ranges (IQR) past the Q1 (25%) and Q3 (75%) cumulative distribution values. Outliers are often defined as $1.5 \times IQR$ and extreme outliers are more than $3 \times IQR$ away from the inner 50% data range.
pointSize	positive integer; sets the cex multiplier for point size.
width	numeric; scaling factor controlling the width of the boxes.
pointShape	positive integer; sets pty for plotting data points. Can be a vector to support additional graphical customization.
plotColors	list; a named list of vectors of colors that set the color options for all NicePlot functions. Names left unspecified will be added and set to default values automatically.
logScale	positive numeric; the base for the for log scale data transformation calculated as `log(x+1,logScale)`.
trim	positive numeric; passed to `threshold` argument of `quantileTrim` if any data points are so extreme that they should be removed before plotting and downstream analysis. Set to FALSE to disable.
pointMethod	character; method to be used for ploting dots. Can be set to "jitter", "linear", "beeswarm" or "distribution".
axisText	character; a length two character vector containing text to be prepended or appended to the major tick labels, respectively.
showCalc	logical; if a p-value can be easily calculated for your data, it will be displayed using the `sub` annotation setting.
calcType	character; should match one of 'none', 'wilcox', 'Tukey','t.test','anova' which will determine which, if any statistical test should be performed on the data.
yLim	numeric vector; manually set the limits of the plotting area (eg. `yLim=c(min,max)`). Used to format the y-axis by default but will modify the x-axis if `side=`TRUE.

rotateLabels	logical; sets las=2 for the x-axis category labels. Will affect y-axis if side=TRUE. Note that this may not work well if long names or with subgrouped data.
rotateY	logical; sets las=2 for the y-axis major tick-mark labels. Will affect x-axis if side=TRUE.
add	logical; causes plotting to be added to the existing plot rather the start a new one.
minorGuides	logical; draws guidelines at minor tick-marks
extendTicks	logical; extends minor tick-marks past the first and last major tick to the edge of the graph provided there is enough room. Works for both log-scale and regular settings.
subGroup	logical; use additional column in by to group the data within each level of the major factor.
subGroupLabels	character vector; sets the labels used for the subGroup factor. Defaults to the levels of the factor.
expLabels	logical; prints the major tick labels is $logScale^x$ instead of the raw value
sidePlot	logical; switches the axis to plot horizontally instead of vertically.
pointHighlights	logical; will use additional factors in by to highlight points in the dot plot
pointLaneWidth	numeric; This controls how far data point dots can move along the categorical axis when plotting. Used for pointMethod options 'jitter', 'beeswarm', and 'distribution'.
na.rm	logical; Should NAs be removed from the data set? Both data input and the factor input from by with be checked.
flipFacts	logical; When a dataframe of values is given, column names are used as a secondary grouping factor by default. Setting flipFacts=TRUE makes the column names the primary factor and by the secondary factor.
verbose	logical; Prints summary and p-value calculations to the screen. All data is silently by the function returned either way.
legend	logical/character; Draw a legend in the plot margins. If a character string is given it will overide the factor name default for the legend title.
logAdjustment	= numeric; This number is added to the input data prior to log transformation. Default value is 1.
...	additional options for S3 method variants

Details

This is a wrapper function for [niceBox](#) that just plots the points with no box distribution data. data point overlay options, data subsetting with a secondary factor, and data point highlighting with a tertiary factor. The complicated part of using this function is handling its many options. A wrapper function to set up and run it with preset options may be a good idea if you are using it along. The function [niceDots](#) is an example of this. Briefly put, the by argument can be a data frame of factors and the function will work through the columns in order as needed. If x is a numeric vector, then by should be a factor to group it into categories. If by is a data frame of factors and subGroup=TRUE, then the first column for by is used as the grouping factor and the second column is used as the sub-grouping factor. If pointHighlights==TRUE, and subGroup=TRUE, the the third column of by is used to highlight points data point overlay (assuming drawPoints=TRUE). If subGroup=FALSE and subGroup=TRUE, then the second column of by is used to control the point highlighting. If x itself is a data frame of numeric vectors, subGroup is automatically set to false and each column of x is plotted like a sub-group and grouped by the first column of by. Data point highlighting

with `pointHighlights=TRUE` can still be used when x is a data frame and the highlighting factor
will be drawn from the second column of by. Please note that the p-values can not always be
calculated and are for general exploratory use only. More careful analysis is necessary to determine
statistical significance. This function is as S3 generic and can be extended to provide class specific
functionality. To further facilitate data exploration, outputs from statistical testing and data set
summaries are printed to the console.

See Also

stripchart, beeswarm, quantileTrim, prepCategoryWindow, niceBox

Examples

```
data(iris)
mCols<-makeColorMatrix()
myCols<-list(fill=mCols[1:3,3],lines="darkblue")
niceDots(iris$Sepal.Length,iris$Species,minorTick=4,showCalc=TRUE,calcType="anova",
    ylab="Sepal Length",main="Sepal Length by Species",plotColors=myCols)
```

---

niceVio   *draw a violin plot*

Description

draws a violin plot with optional scatter plot overlays, subgrouping options and log scale support.

Usage

```
niceVio(x, by = NULL, h = NULL, groupNames = NULL, main = NULL,
  sub = NULL, ylab = NULL, minorTick = FALSE, guides = TRUE,
  theme = basicTheme, outliers = 1.5, pointSize = NULL,
  width = NULL, pointShape = NULL, plotColors = NULL,
  logScale = FALSE, trim = FALSE, pointMethod = NULL,
  axisText = c(NULL, NULL), showCalc = FALSE, calcType = "none",
  drawBox = TRUE, yLim = NULL, rotateLabels = FALSE,
  rotateY = FALSE, add = FALSE, minorGuides = NULL,
  extendTicks = TRUE, subGroup = FALSE, subGroupLabels = NULL,
  expLabels = TRUE, sidePlot = FALSE, drawPoints = TRUE,
  pointHighlights = FALSE, pointLaneWidth = 0.7, flipFacts = FALSE,
  na.rm = FALSE, verbose = FALSE, legend = FALSE,
  trimViolins = TRUE, logAdjustment = 1, ...)
```

Arguments

x	numeric vector or data frame; The input to `prepCategoryWindow` can be a numeric vector a data frame of numeric vectors.
by	factor or data frame of factors; used as the primary grouping factor and the factor levels will be used as group names if `groupNames` is not specified. If by is a data frame and `subGroup=TRUE`, the second column is assumed to be a secondary grouping factor, breaking out the data into sub-categories within each major group determined by the levels of the first column.

h	numeric; Used to override the h hight of density estimator setting in vioplot. Default value is NULL.
groupNames	character vector; overrides the factor levels of by to label the groups
main	character; title for the graph which is supplied to the main argument.
sub	character; subtitle for the graph which is supplied to the sub argument. If NULL and showCalc=TRUE it will be used to display the output form calcStats.
ylab	character; y-axis label.
minorTick	positive integer; number of minor tick-marks to draw between each pair of major ticks-marks.
guides	logical; will draw guidelines at the major tick-marks if set to TRUE. Color of the guidelines is determined by plotColors$guides.
theme	list object; Themes are are an optional way of storing graphical preset options that are compatible with all nicePlot graphing functions.
outliers	positive numeric; number of interquartile ranges (IQR) past the Q1 (25%) and Q3 (75%) cumulative distribution values. Outliers are often defined as $1.5 \times IQR$ and extreme outliers are more than $3 \times IQR$ away from the inner 50% data range.
pointSize	positive integer; sets the cex multiplier for point size.
width	numeric; scaling factor controlling the width of the violins.
pointShape	positive integer; sets pty for plotting data points. Can be a vector to support additional graphical customization.
plotColors	list; a named list of vectors of colors that set the color options for all NicePlot functions. Names left unspecified will be added and set to default values automatically.
logScale	positive numeric; the base for the for log scale data transformation calculated as log(x+1,logScale).
trim	positive numeric; passed to threshold argument of quantileTrim if any data points are so extreme that they should be removed before plotting and downstream analysis. Set to FALSE to disable.
pointMethod	character; method to be used for ploting dots. Can be set to "jitter", "linear", "beeswarm" or "distribution".
axisText	character; a length two character vector containing text to be prepended or appended to the major tick labels, respectively.
showCalc	logical; if a p-value can be easily calculated for your data, it will be displayed using the sub annotation setting.
calcType	character; should match one of 'none', 'wilcox', 'Tukey','t.test','anova' which will determine which, if any statistical test should be performed on the data.
drawBox	logical; should the interquartile boxes be drawn.
yLim	numeric vector; manually set the limits of the plotting area (eg. yLim=c(min,max)). Used to format the y-axis by default but will modify the x-axis if side=TRUE.
rotateLabels	logical; sets las=2 for the x-axis category labels. Will affect y-axis if side=TRUE. Note that this may not work well if long names or with subgrouped data.
rotateY	logical; sets las=2 for the y-axis major tick-mark labels. Will affect x-axis if side=TRUE.
add	logical; causes plotting to be added to the existing plot rather the start a new one.

minorGuides	logical; draws guidelines at minor tick-marks
extendTicks	logical; extends minor tick-marks past the first and last major tick to the edge of the graph provided there is enough room. Works for both log-scale and regular settings.
subGroup	logical; use additional column in by to group the data within each level of the major factor.
subGroupLabels	character vector; sets the labels used for the subGroup factor. Defaults to the levels of the factor.
expLabels	logical; prints the major tick labels is $logScale^x$ instead of the raw value
sidePlot	logical; switches the axis to plot horizontally instead of vertically.
drawPoints	logical; draws a dot plot overlay of the data for each box
pointHighlights	
	logical; will use additional factors in by to highlight points in the dot plot
pointLaneWidth	numeric; This controls how far data point dots can move along the categorical axis when plotting. Used for pointMethod options 'jitter', 'beeswarm', and 'distribution'.
flipFacts	logical; When a dataframe of values is given, column names are used as a secondary grouping factor by default. Setting flipFacts=TRUE makes the column names the primary factor and by the secondary factor.
na.rm	logical; Should NAs be removed from the data set? Both data input and the factor input from by with be checked.
verbose	logical; Prints summary and p-value calculations to the screen. All data is silently by the function returned either way.
legend	logical/character; if not equal to FALSE with cause a legend to be drawn in the margins. If set to a character string instead of a logical value, the string will be used as the legend title insteas of the factor column name from by.
trimViolins	logical; Should the violins be truncated at the edges of the data range.
logAdjustment	= numeric; This number is added to the input data prior to log transformation. Default value is 1.
...	additional options for S3 method variants

Details

This violin plot function offers extensive log scale support, outlier detection, data point overlay options, data subsetting with a secondary factor, and data point highlighting with a tertiary factor. The complicated part of using this function is handling its many options. A wrapper function to set up and run it with preset options may be a good idea if you are using it along. The function [niceDots](#) is an example of this. Briefly put, the by argument can be a data frame of factors and the function will work through the columns in order as needed. If x is a numeric vector, then by should be a factor to group it into categories. If by is a data frame of factors and subGroup=TRUE, then the first column for by is used as the grouping factor and the second column is used as the sub-grouping factor. If pointHighlights==TRUE, and subGroup=TRUE, the the third column of by is used to highlight points data point overlay (assuming drawPoints=TRUE). If subGroup=FALSE and subGroup=TRUE, then the second column of by is used to control the point highlighting. If x itself is a data frame of numeric vectors, subGroup is automatically set to false and each column of x is plotted like a sub-group and grouped by the first column of by. Data point highlighting with pointHighlights=TRUE can still be used when x is a data frame and the highlighting factor will be drawn from the second column of by. Please note that the p-values can not always be

calculated and are for general exploratory use only. More careful analysis is necessary to determine statistical significance. This function is as S3 generic and can be extended to provide class specific functionality. To further facilitate data exploration, outputs from statistical testing and data set summaries are printed to the console.

See Also

[vioplot](), [boxplot](), [niceBox](), [beeswarm](), [prepCategoryWindow]()

Examples

```
data(iris)
mCols<-makeColorMatrix()
myCols<-list(fill=c(mCols[1,3],mCols[2,3],mCols[3,3]),lines="darkblue")
Lab<-"Sepal Length"
niceVio(iris$Sepal.Length,by=iris$Species,minorTick=4,showCalc=TRUE,
    calcType="anova",ylab=Lab,main="Sepal Length by Species",plotColors=myCols)


plot(density(iris$Petal.Length))
lengthFact<-factor(iris$Petal.Length>2.82,labels=c("short","long"))


Title<-"Sepal Length by Species and Petal Length"
factorFrame<-data.frame(Species=iris$Species,PetalLength=lengthFact)
niceVio(iris$Sepal.Length, by=factorFrame, minorTick=4,subGroup=TRUE,
    ylab=Lab,main=Title,plotColors=myCols)
```

---

npColorTheme	*Nice Plots Theme: Colors*

Description

A more colorful and sparse option compared to the default

Usage

```
npColorTheme
```

Format

An object of class npTheme (inherits from list) of length 46.

Details

This theme uses fill, line, and point colors. Not a good option for pointHighlights

---

npGGTheme   *Nice Plots Theme: ggPlot Like*

Description

Looks a bit like stata plots

Usage

```
npGGTheme
```

Format

An object of class npTheme (inherits from list) of length 46.

Details

Todo

---

npStataTheme   *Nice Plots Theme: Stata Like*

Description

Looks a bit like stata plots

Usage

```
npStataTheme
```

Format

An object of class npTheme (inherits from list) of length 46.

Details

Todo

---

prepBarData   *Prepare and print basic statistics for niceBar*

Description

Processess the input data, factors, and options to produce summary data for drawing bar plots

Usage

```
prepBarData(x, by, errorMultiple = 1, upperErrorFun = "sd",
  lowerErrorFun = upperErrorFun, aggFunction = "mean", stack = FALSE,
  subGroup = FALSE)
```

Arguments

x	list; a list object returned by [prepCategoryWindow](#)
by	factor or dataframe of factors; One or more factors that control how the data is grouped. The first column is the primary grouping factor and the second and thrid columns are used for sub-grouping and/or stacking as needed.
errorMultiple	numeric; How many standard errors/deviations should be represented by the error bars.
upperErrorFun	character string; Determines how the error barse are calculated. Options are [sd](#) (standard deviation), [se](#) (standard error), [min](#), [max](#) and [boot95ci](#) (bootstrap 95% confidence interval).
lowerErrorFun	character; A character vector for the primary group names
aggFunction	character string; An string naming the function to be used to aggregate the grouped data. Typically should be either [median](#) or [mean](#).
stack	logical; Should one of the factors in by be used make a stacked bar plot. Note that this sort of analysis is nonsensical for many data sets.
subGroup	logical; If [TRUE](#) the data will be faceted into subgroups within the primary factor levels. Ignored if by is a [factor](#) or x is a [data.frame](#).

Details

todo<-1

See Also

[niceBar](#), [boot95ci](#), [drawBar](#)

Examples

todo<-1

---

prepCategoryWindow | *prepare a plotting environment for categorical data such as bar plots or box plots*

Description

takes untransformed data and draws the x and y axis with support of subgrouping data within factors, log transformation and outlier trimming.

Usage

```
prepCategoryWindow(x, by = NULL, groupNames = levels(by),
  minorTick = FALSE, guides = TRUE, yLim = NULL,
  rotateLabels = FALSE, rotateY = TRUE, theme = NA, plotColors = if
  (is.na(theme)) {      list(bg = "open", guides = "black", lines =
  "gray22", points = "darkgrey", fill = "white") } else {
  theme$plotColors }, trim = FALSE, logScale = FALSE,
  axisText = c(NULL, NULL), minorGuides = FALSE, extendTicks = F,
  subGroup = FALSE, expLabels = TRUE, sidePlot = FALSE,
  subGroupLabels = NULL, strictLimits = F, legend = FALSE,
  pointHighlights = FALSE, logAdjustment = 1)
```

Arguments

x	numeric vector or data frame; The input to `prepCategoryWindow` can be a numeric vector a data frame of numeric vectors.
by	factor or data frame of factors; used as the primary grouping factor and the factor levels will be used as group names if `groupNames` is not specified. If by is a data frame and `subGroup=`TRUE, the second column is assumed to be a secondary grouping factor, breaking out the data into sub-categories within each major group determined by the levels of the first column.
groupNames	character vector; overrides the factor levels of by to label the groups
minorTick	positive integer; number of minor tick-marks to draw between each pair of major ticks-marks.
guides	logical; will draw guidelines at the major tick-marks if set to TRUE. Color of the guidelines is determined by `plotColors$guides`.
yLim	numeric vector; manually set the limits of the plotting area (eg. `yLim=c(min,max)`). Used to format the y-axis by default but will modify the x-axis if `side=`TRUE.
rotateLabels	logical; sets `las=2` for the x-axis category labels. Will affect y-axis if `side=`TRUE. Note that this may not work well if long names or with subgrouped data.
rotateY	logical; sets `las=2` for the y-axis major tick-mark labels. Will affect x-axis if `side=`TRUE.
theme	list; A `NicePlots` plotColor list from a theme.
plotColors	list; a named list of vectors of colors that set the color options for all NicePlot functions. Names left unspecified will be added and set to default values automatically.
trim	positive numeric; passed to `threshold` argument of `quantileTrim` if any data points are so extreme that they should be removed before plotting and downstream analysis. Set to FALSE to disable.

logScale	positive numeric; the base for the for log scale data transformation calculated as `log(x+1,logScale)`.
axisText	character; a length two character vector containing text to be prepended or appended to the major tick labels, respectively.
minorGuides	logical; draws guidelines at minor tick-marks
extendTicks	logical; extends minor tick-marks past the first and last major tick to the edge of the graph provided there is enough room. Works for both log-scale and regular settings.
subGroup	logical; use additional column in by to group the data within each level of the major factor.
expLabels	logical; prints the major tick labels is $logScale^x$ instead of the raw value
sidePlot	logical; switches the axis to plot horizontally instead of vertically.
subGroupLabels	character vector; sets the labels used for the subGroup factor. Defaults to the levels of the factor.
strictLimits	logical; eliminates padding on the value axis so 0 can be flush with the x-axis. Defaults to FALSE.
legend	logical/character; Draw a legend in the plot margins. If a character string is given it will overide the factor name default for the legend title.
pointHighlights	
	logical; Is pointHightlights turned on? This is used to determin with column of by should be used for legend factor levels.
logAdjustment	= numeric; This number is added to the input data prior to log transformation. Default value is 1.

Details

This function does all the hard work of setting up the x and y axis for plotting as well as optionally log transforming and/or trimming the data of outliers. In particular, it adds much more robust support for plotting of log transformed data and subgrouping of primary vectors. Other features include the addition of both major and minor guidelines, support for horizontal plotting and improved label formatting options.

Value

formats the plotting area and returns a named list with 'data' and 'labels' corresponding to the trimmed and/or transformed data and the labels for the primary factors, respectively.

See Also

axisTicks, axis, makeLogTicks, facetSpacing

Examples

```
todo<-1
```

---

prepNiceData   *Prepare and print basic statistics for niceBox and niceVio*

Description

Uses filtred data with subgroup and factor information to calculate quartile data for display and plotting.

Usage

```
prepNiceData(prepedData, by, subGroup = FALSE, outliers = TRUE, filter,
   groupNames, plotLoc, width = 1, flipFacts = FALSE, verbose = FALSE)
```

Arguments

prepedData	list; a list object returned by [prepCategoryWindow](#)
by	factor or dataframe of factors; One or more factors that control how the data is grouped. The first column is the primary grouping factor and the second and thrid columns are used for sub-grouping and highlighting as needed.
subGroup	logical; Should the data be faceted into subgroups within the primary factor levels. Ignored if by is a [factor](#).
outliers	positive numeric; number of interquartile ranges (IQR) past the Q1 (25%) and Q3 (75%) cumulative distribution values. Outliers are often defined as $1.5 \times IQR$ and extreme outliers are more than $3 \times IQR$ away from the inner 50% data range.
filter	logical vector; Used to further filter the data if necissary.
groupNames	character; A character vector for the primary group names
plotLoc	numeric vector; A vector indicating where each element should be plotted
width	numeric; A multiplier that controls how wide the ploting elements will be. Setting width=1.1 would result in plot elements being 10% wider.
flipFacts	logical; When a dataframe of values is given, column names are used as a secondary grouping factor by default. Setting flipFacts=[TRUE](#) makes the column names the primary factor and by the secondary factor.
verbose	logical; Will print summary statistics to the screen if set to [TRUE](#). The function will return the calculations either way.

Details

To aid in data interpretation and exploration, quartile distribution statistics are calculated for each group and subgroup if specified. For [niceBox](#) this data is also used to plot the data. The data is parsed by checking outlier and subGroup status as weel as checking if either prepedData or by are a [data.frame](#) or a [vector](#).

See Also

[niceBox](#), [niceVio](#), [niceDots](#)

Examples

```
data(iris)
filter<-rep(TRUE,length(iris$Species))
loc<-seq(1,length(levels(iris$Species)))
data<-list(data=iris[,1:4])
myData<-prepNiceData(data,by=iris$Species,filter=filter,plotLoc=loc,
    groupNames=levels(iris$Species),outliers=FALSE)
print(myData)
```

---

procNiceOptions   *Process ploting options*

Description

Integrates theme and user arguments to finalize all options prior to plotting

Usage

```
procNiceOptions(x, by, minorTick, pointShape, whiskerLineType, lWidth,
  capWidth, pointLaneWidth, width, guides, pointSize, subGroup = FALSE,
  stack = F, pointHighlights = F, type = c("BP", "VP", "DP", "Bar"),
  theme, plotColors, pointMethod, logScale, drawPoints, groupNames,
  swarmOverflow, errorCap = NULL)
```

Arguments

x	Data to be plotted which has been preprocessed by [dataFlightCheck](#)
by	factor or dataframe of factors; One or more factors that control how the data is grouped. The first column is the primary grouping factor and the second and thrid columns are used for sub-grouping and highlighting as needed.
minorTick	numeric; Number of minor tickmarks to be drawn between the major marks
pointShape	numeric; vector of numbers corresponding to pty options for ploting data overlays.
whiskerLineType	
	numeric; number corresponding to lty option for drawing the whiskers and error bars for box plots and bar plots, respectively.
lWidth	numeric; number creesponding to the lwd option for ploting lines on the graph
capWidth	numeric; Width of the cap relative to the bar/box width for box plots and bar plots.
pointLaneWidth	numeric; This controls how far data point dots can move along the categorical axis when plotting. Used for pointMethod options 'jitter', 'beeswarm', and 'distribution'.
width	numeric; A multiplier that controls how wide the ploting elements will be. Setting width=1.1 would result in plot elements being 10% wider.
guides	logical; Should guidelines be drawn at the major tick marks.
pointSize	numeric; vector of numerics controling the size of points on the data overlay

subGroup	logical; Should the data be faceted into subgroups within the primary factor levels. Ignored if by is a [factor](#).
stack	logical; Triggers stacked bar analysis for bar plots
pointHighlights	
	logical; will use additional factors in by to highlight points in the dot plot
type	character; What kind of plot is this for? Case sensitive options are "BP", "DP", "VP", and "Bar" corresponding to box plots, dot plots, violin plots, and bar plots, respectively.
theme	list object; Themes are are an optional way of storing graphical preset options that are compatible with all nicePlot graphing functions.
plotColors	list; a named list of vectors of colors that set the color options for all NicePlot functions. Names left unspecified will be added and set to default values automatically.
pointMethod	character; method to be used for ploting dots. Can be set to "jitter", "linear", "beeswarm" or "distribution".
logScale	numeric; Should a log scale use used (TRUE/FALSE)? Otherwise indicates the base for the log transformation.
drawPoints	logical; draws a dot plot overlay of the data.
groupNames	character; A character vector for the primary group names
swarmOverflow	character; Valid options are: "none", "wrap", "gutter", "random", and "omit". Controls how to wantly point stacks that would overflow the pointLaneWidth option.
errorCap	character; Determines the style for the ends of the error bars. Valid options are ball, bar or none.

Details

This is a private utility function used by NicePlots to integrate user options with theme defaults. Anything specified by the user is treated literally with no additional optimization. Colors, point shapes, fills, etc. are optimized to best enhance the relevant factor visualisations selected. The finalized parameters are returned as a named list.

Value

Named listed of graphical options

See Also

[formatPlotColors](#), [niceBox](#), [niceDots](#), [niceVio](#), [niceBar](#)

---

quantileTrim *Filter data by interquartile range*

Description

quantileTrim takes a numeric vector and removes data points that fall more than threshold * the interquartile range outside of the interquartile range. If returnFilter is set to TRUE then the function returns a named list with the trimmed data and a logical vector

Usage

```
quantileTrim(x, threshold = 3, na.rm = FALSE, returnFilter = FALSE)
```

Arguments

x	a numeric vector or a object compatible with the quantile function
threshold	numeric; the number of interquartile ranges out side of the inner 50% range of the data to use as a cutoff from trimming. Typical values include 1.5 for outliers and 3 for extreme outliers.
na.rm	logical; if true will remove all NA values from x before analyzing the data.
returnFilter	logical; will cause the function to return a list including with both the trimmed data and a logical vector that can be used to filter objects of the same length as x.

Details

The interquartile range (IQR) also known as the H-spread, represents the range encompassing the middle 50 This is is used to as a measure of dispersion around the median and more frequently to detect outlier data points. Here data points are filtered if $x < Q_1 - threshold \times IQR$ and $x > Q_3 + threshold \times IQR$ where Q_1 and Q_3 represent the cumulative 25

Value

The trimmed numeric vector or a returnFilter is TRUE then a named list labeled data and filter is returned with the trimmed data and the logical filtering vector, respectively.

See Also

quantile.

Examples

```
x<-rnorm(1000)
paste0(mean(x)," (",range(x),")")
x<-quantileTrim(x,threshold=1.5)
paste0(mean(x)," (",range(x),")")

#Example using the filter function:
myData<-c(NA,rnorm(100),NA,NA,rnorm(100),NA,NA,NA,rnorm(300),NA,10000)
myIndex<-1:508
newData<-quantileTrim(myData,na.rm=TRUE,returnFilter=TRUE)
identical(newData$data,myData[newData$filter])
```

---

se | *Calculate the standard error of the mean*

Description

se takes a numeric vector and returns the corresponding standard error of the mean

Usage

```
se(x)
```

Arguments

x   numeric vector

Details

This is a convenience function internal to NicePlots and is not exported. If x is a numeric vector, SD is the standard deviation of x and N is the length of x then the standard error (se) of the mean may be caclulated as:

$$se = \frac{SD}{\sqrt{N}}$$

Value

a double corresponding to the standard error of the mean

See Also

[sd](#)

Examples

```
data(iris)
#se(iris$Sepal.Length)
```

---

setAlpha   *add alpha transparency to a named color*

Description

Takes a named color such as "red" or "darkgreen" and adds a level of transparancy based on the alpha setting.

Usage

```
setAlpha(x, alpha = 0.2)
```

Arguments

x   character string; a text string corresponding to an R color

alpha   numeric [0-1]; sets the level of transparency.

Details

setAlpha is a convenience function that uses the [col2rgb](#) and [rgb](#) to add transparancy to named colors.

Value

An rbg color with transparancy alpha.

See Also

[makeColorMatrix](#), [rainbow](#), [col2rgb](#), [rgb](#).

Examples

```
plot(1,1,col="white",xlim=c(0,10),ylim=c(0,10))
rect(1,1,7,7,col=setAlpha("darkblue"))
rect(3,3,9,9, col=setAlpha("red"))
```

---

t95ci	*Calculate a 95% confidence interval from the t-distribution*

Description

se Calculates the 95% confidence interval of the mean for a vector of data based on the t-distribution

Usage

```
t95ci(x)
```

Arguments

x   numeric vector

Details

This is a convenience function internal to NicePlots and is not exported. If x is a numeric vector, SD is the standard deviation of x, N is the length of x and TQ is the t-distrubition quantile for 0.975 with N-1 degrees for freedom then the 95% confidence interval of the mean may be caclulated as:

$$95\%ci = \frac{QT * SD}{\sqrt{N}}$$

Value

a double corresponding to the length of one arm ofthe 95% confidence interval

See Also

[sd](#)

Examples

```
data(iris)
#t95ci(iris$Sepal.Length)
```

Index