

Package ‘oaPlots’

July 22, 2025

Maintainer Jason Waddell <jason.waddell@openanalytics.eu>

License GPL-3 + file LICENSE

Title OpenAnalytics Plots Package

Type Package

LazyLoad yes

Author Jason Waddell, Willem Ligtenberg

Description Offers a suite of functions for enhancing R plots.

Version 0.0.25

Date 2015-11-29

URL <http://www.openanalytics.eu>

Depends oaColors

Imports ggplot2

Suggests RColorBrewer

NeedsCompilation no

Repository CRAN

Date/Publication 2015-11-30 14:51:34

Contents

addLegend	2
blankPlot	3
colorPoly	4
customRound	4
densityLegend	5
drawSplitDensity	6
findLocations	7
getBreaks	8
oaTemplate	8
oaTheme	10
plotBars	11

plotDensityTrace	11
plotDots	12
plotPolygonRegions	13
pointsOnBoxplot	13
pointsOnBoxplot.default	14
pointsOnBoxplot.formula	16
prepLegend	16
scatterplotDL	18
splitCircle	19
splitColorVar	20

Index 21

addLegend	<i>Function for adding a legend to an existing device</i>
-----------	---

Description

Function for adding a legend to an existing device

Usage

```
addLegend(x = "center", y = NULL, legend, font = NULL, bty = "n",
          xjust = 0.5, yjust = 0.5, ...)
```

Arguments

x	legend x location
y	legend y location
legend	vector of legend labels
font	legend text font
bty	A character string which determined the type of box which is drawn about plots. If bty is one of "o" (the default), "l", "7", "c", "u", or "]" the resulting box resembles the corresponding upper case letter. A value of "n" suppresses the box.
xjust	how the legend is to be justified relative to the legend x location. A value of 0 means left justified, 0.5 means centered and 1 means right justified.
yjust	the same as xjust for the legend y location.
...	additional optional arguments to be passed to legend()

Value

none; legend is added to the current device

Author(s)

Jason Waddell

Examples

```

layout <- c(2,3);
side <- "left"
proportion <- 0.2

preLegend(layout = layout, side = side, proportion = proportion)
for(i in 1:(layout[1]*layout[2]))
plot(1:7, 1:7, col = 1:7, pch = 19, cex = 2.2, xaxt = "n",
yaxt = "n", ann = FALSE)
addLegend(legend = paste("Group", 1:7), font = 2,
pch = 19, pt.cex = 2, text.col = 1:7, col = 1:7,
y.intersp = 1.5, cex = 1.5)

layout = rbind(c(1, 2, 3), c(0, 4, 3), c(0, 4, 5))
side = "right"
proportion = 0.15

preLegend(layout = layout, side = side, proportion = proportion)
for(i in 1:max(layout))
plot(1:7, 1:7, col = 1:7, pch = 19, cex = 2.2, xaxt = "n",
yaxt = "n", xlab = "", ylab = "", main = paste("Plot", i))
addLegend(legend = paste("Group", 1:7), font = 2,
pch = 19, pt.cex = 2, text.col = 1:7, col = 1:7,
y.intersp = 1.5, cex = 1.5)

```

blankPlot

*Create a Blank Plot***Description**

Create a Blank Plot

Usage

```
blankPlot(xlim, ylim)
```

Arguments

xlim	x limits for the plot
ylim	y limits for the plot

Value

none, plot is created on the current device

Author(s)

Jason Waddell

colorPoly	<i>Function for plotting a colored polygon as part of a density legend</i>
-----------	--

Description

Function for plotting a colored polygon as part of a density legend

Usage

```
colorPoly(de1, tempDex, col, side)
```

Arguments

de1	a density() object
tempDex	a set of indices corresponding to the range of the current segment to be plotted (which indices of the density object to use)
col	the color of the polygon to be plotted
side	the side of the plot that the density legend should be plotted on

Value

none, graphics are added to the current device

Author(s)

Jason Waddell

customRound	<i>Custom rounding function to round to the nearest specified interval</i>
-------------	--

Description

Custom rounding function to round to the nearest specified interval

Usage

```
customRound(x, roundTo)
```

Arguments

x	numeric value(s)
roundTo	rounding interval

Value

rounded numeric value(s)

Author(s)

Jason Waddell

densityLegend

Create a colored density legend for visually representing the distribution of a color variable on a plot

Description

Create a colored density legend for visually representing the distribution of a color variable on a plot

Usage

```
densityLegend(x, colorPalette, colorBreaks, side = "right", main = NULL)
```

Arguments

x	a numeric vector used to create the density trace
colorPalette	a vector of color values
colorBreaks	a vector of cutoff values for the color regions
side	the side of the plot to place the density legend
main	the main title for the density legend (optional, recommended to use a title that describes x)

Value

none, graphics are added to the current device

Author(s)

Jason Waddell

Examples

```
library(ggplot2)
library(RColorBrewer)

# subset the data object
dsub <- subset(diamonds, x > 5 & x < 6 & y > 5 & y < 6)
dsub <- dsub[-which(dsub$z > 4), ]
dsub <- dsub[-which(dsub$z < 3), ]

# define color palette, color vector and color region breaks
colorPalette <- brewer.pal(9, "Blues")[4:9]
colorObj <- splitColorVar(colorVar = dsub$z, colorPalette)
colorVec <- colorObj$colorVec
```

```

breaks <- colorObj$breaks

# plot the data
preLegend(side = "right", proportion = 0.3)
oaTemplate(xlim = range(dsub$x), ylim = range(dsub$y),
main = "Diamond Length by Width \n Colored by Depth",
xlab = "Length (mm)", ylab = "Width (mm)")
points(x = dsub$x, y = dsub$y, col = colorVec, pch = 19, cex = 0.6)

# add the legend
densityLegend(x = dsub$z, colorPalette = colorPalette, side = "right",
main = "Diamond Depth", colorBreaks = breaks)

```

drawSplitDensity	<i>Draw a Split Density Plot</i>
------------------	----------------------------------

Description

Draw a Split Density Plot

Usage

```

drawSplitDensity(x = NULL, y = NULL, densityObj = NULL, yshift = 0,
  colVec, outerCol, lwd = 2, split = NULL, yScale = NULL,
  fillBackground = FALSE)

```

Arguments

x	x vector from a density object. e.g. data <- rnorm(100); x <- density(data)\$x
y	y vector from a density object
densityObj	an object created by the function density()
yshift	vertical shift to be applied to the y object
colVec	color vector for the shaded regions that compose the interior of the plot. The length of 'colVec' should be one greater than the length of split
outerCol	the color for the outer density line
lwd	line width for the outer density line
split	vector of x values at which to split the density plot
yScale	vertical scale at which to plot the density. For example, a call with 'yScale = 1' will produce a density curve scaled between 0 and 1
fillBackground	binary specification of whether to fill in the background the outerCol color

Value

none. Graph is plotted to the current device

Author(s)

Jason Waddell

Examples

```
library(RColorBrewer)
data <- rnorm(1000)
x <- density(data)$x
y <- density(data)$y
colVec <- brewer.pal(9, "Blues")[3:8]
outerCol <- brewer.pal(9, "Blues")[9]

oaTemplate(xlim = range(x), ylim = c(0, 1), ygrid = 0, cex.axis = 1.2)
drawSplitDensity(x, y, colVec = colVec, split = c(-8),
  outerCol = outerCol,
  yScale = 0.95, yshift = 0)
```

findLocations	<i>Returns a Vector of x Locations</i>
---------------	--

Description

Returns a Vector of x Locations

Usage

```
findLocations(n, space, center)
```

Arguments

n	number of observations for a given value
space	space between points
center	center plotting value

Value

numeric vector of location values

Author(s)

Jason Waddell

getBreaks	<i>Divide the range of x into intervals, returning the breakpoints of these intervals</i>
-----------	---

Description

Divide the range of x into intervals, returning the breakpoints of these intervals

Usage

```
getBreaks(x, breaks, dig.lab = 3L)
```

Arguments

x	a numeric vector which is to be converted to a factor by cutting
breaks	a single number (greater than or equal to 2) giving the number of intervals into which x is to be cut
dig.lab	integer which is used when labels are not given. It determines the number of digits used in formatting the break numbers

Value

a vector of numeric breakpoints

Author(s)

Jason Waddell

oaTemplate	<i>Create a OA Plot Template</i>
------------	----------------------------------

Description

Create a OA Plot Template

Usage

```
oaTemplate(xlim, ylim, xgrid = NULL, ygrid = NULL, xlab = NULL,
  ylab = NULL, main = NULL, bgCol = gray(0.9), col.axis = gray(0.6),
  col.lab = gray(0.4), col.main = gray(0.3), cex.axis = 0.7,
  cex.lab = 1, cex.main = 1.5, xaxs = "r", yaxs = "r", add = FALSE,
  box = FALSE, box.col = "black", box.lwd = 1, ylabels = NULL,
  xlabels = NULL, buffer = 0, gridLabelBuffer = 0.01, ylabBuffer = 0.1,
  xlabBuffer = 0.08, mainBuffer = 0.07)
```


Arguments

xlim	x limits for the plot
ylim	y limits for the plot
xgrid	values at which to draw the x axis gridlines
ygrid	values at which to draw the y axis gridlines
xlab	a title for the x axis
ylab	a title for the y axis
main	an overall title for the plot
bgCol	background color for the plot
col.axis	color for the axis labels
col.lab	color for the xlab and ylab titles
col.main	color for the main title
cex.axis	size of the axis labels
cex.lab	size of the xlab and ylab titles
cex.main	size of the main title
xaxs	The style of axis interval calculation to be used for the x-axis. Possible values are "r", "i". Style "r" (regular) first extends the data range by 4 percent at each end and then finds an axis with pretty labels that fits within the extended range. Style "i" (internal) just finds an axis with pretty labels that fits within the original data range.
yaxs	The style of axis interval calculation to be used for the y-axis. See xaxs above.
add	A logical value specifying whether to add the template to an existing plot. If FALSE, a new plot will be created
box	binary specifying whether to draw a bounding box around the plot
box.col	color of the bounding box
box.lwd	width of the bounding box lines
ylabls	labels to print at the y tickmarks
xlabls	labels to print at the x tickmarks
buffer	optional buffer around all edges of the plot (as a percentage of the plot)
gridLabelBuffer	buffer between plot and grid labels (as a proportion of plotting range)
ylabBuffer	distance between plot and y-axis title, as proportion of total plot width
xlabBuffer	distance between plot and x-axis title, as proportion of total plot height
mainBuffer	distance between plot and main title, as proportion of total plot height

Value

none, objects are plotted to the current device

Author(s)

Jason Waddell

Examples

```
par(plt = c(0, 1, 0, 1))
oaTemplate(xlim = c(0, 10), ylim = c(20, 50), add = FALSE, xlab = "X Label", ylab = "Y Label",
main = "Main Title")
```

oaTheme

Apply OA ggplot2 theme

Description

Apply OA ggplot2 theme

Usage

```
oaTheme(p, useOAColors = TRUE, expand = "both", bgColor = gray(0.9))
```

Arguments

p	ggplot2 plot object
useOAColors	boolean which indicates wether or not to use the oaColors package to provide a color scheme. Default: TRUE
expand	specify wether or not to expand the axis valid options are: (both, x, y, none) Default: both
bgColor	specify a different background color (useful for plotting colors with alpha values) Default: gray(0.9)

Value

ggplot2 plot object

Author(s)

Willem Ligtenberg

plotBars	<i>A function for creating the segmented color bars in a density legend</i>
----------	---

Description

A function for creating the segmented color bars in a density legend

Usage

```
plotBars(de1, side, colorPalette, colorBreaks)
```

Arguments

de1	a density() object
side	the side of the plot that the density legend should be plotted on
colorPalette	A vector of color values
colorBreaks	A vector of cutoff values for the color regions

Value

none, graphics are added to the current device

Author(s)

Jason Waddell

plotDensityTrace	<i>Function for plotting the density trace outline in a density legend</i>
------------------	--

Description

Function for plotting the density trace outline in a density legend

Usage

```
plotDensityTrace(de1, side)
```

Arguments

de1	a density() object
side	the side of the plot that the density legend should be plotted on

Value

none, graphics are added to the current device

Author(s)

Jason Waddell

plotDots

Adds Points on a Pre-existing Plot using Shifted Locations

Description

Adds Points on a Pre-existing Plot using Shifted Locations

Usage

```
plotDots(vec = NULL, xLeft = 0.8, xRight = 1.2, ...)
```

Arguments

<code>vec</code>	numeric vector
<code>xLeft</code>	left x boundary of the point plotting region
<code>xRight</code>	right x boundary of the point plotting region
<code>...</code>	further arguments to be handed to the points function

Value

points are added to the current graphics device

Author(s)

Jason Waddell

Examples

```
x <- sample(1:5, size = 25, replace = TRUE)
plot(x = -1, y = -1, xlim = c(0.5,1.5), ylim = range(x),
     ylab = "", xlab = "", xaxt = "n")
colVec <- c(rep("olivedrab", 15), rep("goldenrod", 5), rep("red", 5))
plotDots(vec = x, xLeft = 0.8, xRight = 1.2, pch = 19,
        col = colVec, cex = 2)
```

plotPolygonRegions	<i>Function to plot all colored density regions of a density legend</i>
--------------------	---

Description

Function to plot all colored density regions of a density legend

Usage

```
plotPolygonRegions(de1, side, colorPalette, colorBreaks)
```

Arguments

de1	a density() object
side	the side of the plot that the density legend should be plotted on
colorPalette	a vector of color values
colorBreaks	a vector of cutoff values for the color regions

Value

none, graphics are added to the current device

Author(s)

Jason Waddell

pointsOnBoxplot	<i>Generic pointsOnBoxplot function. Calls pointsOnBoxplot.default or pointsOnBoxplot.formula</i>
-----------------	---

Description

Generic pointsOnBoxplot function. Calls pointsOnBoxplot.default or pointsOnBoxplot.formula

Usage

```
pointsOnBoxplot(x, ...)
```

Arguments

x	a vector of numeric values to be passed on
...	further arguments for the methods, such as a vector of categories 'y' for the default method

Author(s)

Jason Waddell

See Also

[pointsOnBoxplot.default](#) [pointsOnBoxplot.formula](#)

Examples

```
# Examples run in the formula and default methods
x2 <- runif(50, 0, 10);
table(customRound(x2, roundTo = 0.5))
boxplot(x2)
pointsOnBoxplot(x2, pch = 19, roundTo = 0.5)

# Set up input data
x <- c(sample(1:5, size = 25, replace = TRUE), rpois(25, lambda = 4))
colVec <- c(rep("olivedrab", 10), rep("red", 5), rep("goldenrod", 15),
            rep("red", 15), rep("olivedrab", 5))
y <- rep(c("Awesome Rats", "Stupid Rats"), each = 25)
y2 <- rep(c("Open", "Analytics"), 25)

x2 <- c(1, 2, 2, 3, 3, 1, 1, 1, 4, 5)
y3 <- c(rep("A", 5), rep("B", 5))
levels(y3) <- c("A", "B", "C")

boxplot(x ~ y, horizontal = TRUE)
pointsOnBoxplot(x ~ y, horizontal = TRUE)

boxplot(x ~ y)
pointsOnBoxplot(x = x, y = y, col = colVec, pch = 19, cex = 2)

boxplot(x ~ y + y2)
pointsOnBoxplot(x ~ y + y2, col = colVec, pch = 19, cex = 2)
```

pointsOnBoxplot.default

Draw Points on Top of a Boxplot using Appropriate Shifting

Description

Draw Points on Top of a Boxplot using Appropriate Shifting

Usage

```
## Default S3 method:
pointsOnBoxplot(x = NULL, y = NULL, totalSpread = 0.3,
               roundTo = NULL, horizontal = FALSE, ...)
```

Arguments

<code>x</code>	vector of numeric values that were used to create boxplots
<code>y</code>	vector of values representing a categorical variable
<code>totalSpread</code>	total spread of point plotting range within a boxplot. Defaults to 0.3 so that points plot between 0.85 and 1.15
<code>roundTo</code>	optional rounding interval. For example, if given <code>roundTo = 0.25</code> , all numeric <code>x</code> values will be rounded to the nearest quarter
<code>horizontal</code>	logical indicating if the boxplots should be horizontal; default <code>FALSE</code> means vertical boxes.
<code>...</code>	further parameters to be passed to the <code>points</code> function

Value

points are drawn to the current device

Author(s)

Jason Waddell

Examples

```
# Examples run in the formula and default methods
x2 <- runif(50, 0, 10);
table(customRound(x2, roundTo = 0.5))
boxplot(x2)
pointsOnBoxplot(x2, pch = 19, roundTo = 0.5)

# Set up input data
x <- c(sample(1:5, size = 25, replace = TRUE), rpois(25, lambda = 4))
colVec <- c(rep("olivedrab", 10), rep("red", 5), rep("goldenrod", 15),
            rep("red", 15), rep("olivedrab", 5))
y <- rep(c("Awesome Rats", "Stupid Rats"), each = 25)
y2 <- rep(c("Open", "Analytics"), 25)

x2 <- c(1, 2, 2, 3, 3, 1, 1, 1, 4, 5)
y3 <- c(rep("A", 5), rep("B", 5))
levels(y3) <- c("A", "B", "C")

boxplot(x ~ y, horizontal = TRUE)
pointsOnBoxplot(x ~ y, horizontal = TRUE)

boxplot(x ~ y)
pointsOnBoxplot(x = x, y = y, col = colVec, pch = 19, cex = 2)

boxplot(x ~ y + y2)
pointsOnBoxplot(x ~ y + y2, col = colVec, pch = 19, cex = 2)
```

```
pointsOnBoxplot.formula
```

Draw Points on Top of a Boxplot using Appropriate Shifting

Description

Draw Points on Top of a Boxplot using Appropriate Shifting

Usage

```
## S3 method for class 'formula'
pointsOnBoxplot(formula, data = NULL, ...,
  na.action = NULL)
```

Arguments

formula	a formula of the form $a \sim b$ (+ c, etc.), where a is a numeric vector and all other variables are categorical
data	an optional input parameter of a data.frame containing the variables used in the formula
...	further arguments to be passed to pointsOnBoxplot.default
na.action	parameter specifying how to handle missingness

Author(s)

Jason Waddell

```
prepLegend
```

Function for arranging plotting layout to accomodate a legend panel

Description

Function for arranging plotting layout to accomodate a legend panel

Usage

```
prepLegend(layout = c(1, 1), type = if (is.matrix(layout)) "layout" else
  "mfrow", side = "right", proportion = 0.15, heights = NULL,
  widths = NULL)
```


Arguments

layout	layout vector or matrix
type	type of layout; either "mfrow" or "layout"
side	side of the plot to place legend on; one of "top", "bottom", "left" or "right"
proportion	proportion of plotting window to allocate to legend
heights	height vector for original layout (before the legend panel is appended)
widths	width vector for original layout (before the legend panel is appended)

Value

none; layout is passed to current device

Author(s)

Jason Waddell

Examples

```
layout <- c(2,3);
side <- "left"
proportion <- 0.2

prepLegend(layout = layout, side = side, proportion = proportion)
for(i in 1:(layout[1]*layout[2]))
plot(1:7, 1:7, col = 1:7, pch = 19, cex = 2.2, xaxt = "n",
yaxt = "n", ann = FALSE)
addLegend(legend = paste("Group", 1:7), font = 2,
pch = 19, pt.cex = 2, text.col = 1:7, col = 1:7,
y.intersp = 1.5, cex = 1.5)

layout = rbind(c(1, 2, 3), c(0, 4, 3), c(0, 4, 5))
side = "right"
proportion = 0.15

prepLegend(layout = layout, side = side, proportion = proportion)
for(i in 1:max(layout))
plot(1:7, 1:7, col = 1:7, pch = 19, cex = 2.2, xaxt = "n",
yaxt = "n", xlab = "", ylab = "", main = paste("Plot", i))
addLegend(legend = paste("Group", 1:7), font = 2,
pch = 19, pt.cex = 2, text.col = 1:7, col = 1:7,
y.intersp = 1.5, cex = 1.5)
```

scatterplotDL

Plot a base-graphics scatterplot with accompanying density legend

Description

Plot a base-graphics scatterplot with accompanying density legend

Usage

```
scatterplotDL(x, y, colorVar, colorPalette, side = "right",
  proportion = 0.3, legendTitle = NULL, ...)
```

Arguments

x	the x coordinates to be handed to plot()
y	the y coordinates of points in the plot()
colorVar	the numeric vector of values used to color the points
colorPalette	a color palette. If 'colorPalette' contains, for example, 6 colors, then the values of colorVar will be split and assigned to these 6 colors
side	the side of the plot to put the density legend on ("left", "right", "top", or "bottom")
proportion	the proportion of the plot (from 0 to 1) to allocate to the density legend (defaults to 0.3)
legendTitle	string for labelling the density legend
...	additional parameters to be passed to plot()

Value

none, plot is added to device

Author(s)

Jason Waddell

Examples

```
library(ggplot2)
library(RColorBrewer)
colorPalette <- brewer.pal(9, "YlOrRd")[4:9]
scatterplotDL(x = mtcars$mpg, y = mtcars$wt, colorVar = mtcars$hp,
  legendTitle = "Horse Power", colorPalette = colorPalette, pch = 19,
  xlab = "MPG (miles per gallon)", ylab = "Weight (tonnes)",
  main = "MPG by Weight in Cars \n Colored by Horse Power")
```

splitCircle	<i>Function for drawing a split circle (two differently colored semicircles)</i>
-------------	--

Description

Function for drawing a split circle (two differently colored semicircles)

Usage

```
splitCircle(x, y, radius, splitAngle = pi/4, nv = 100, border = NA,  
            col1 = NA, col2 = NA, lty = 1, lwd = 1)
```

Arguments

x	x location of the circle center
y	y location of the circle center
radius	radius of the circle
splitAngle	angle (in radians) that splits the color in two halves
nv	number of vertices used to draw the circle
border	binary whether to include a border on the circle
col1	color of the first semicircle
col2	color of the second semicircle
lty	line type used for drawing the circle polygon
lwd	line width used for drawing the circle polygon

Value

none, split circle is drawn to the current device

Author(s)

Jason Waddell

Examples

```
plot(-1, -1, xlim = c(0, 1), ylim = c(0,1), type = "n")  
splitCircle(x = 0.5, y = 0.5, radius = 0.48,  
            splitAngle = pi/4, nv = 1000, border = NA,  
            col1 = "blue", col2 = "red")
```

splitColorVar	<i>Function to take a numeric vector 'colorVar' and palette 'colorPalette', and return a list containing a vector of color assignments for each element of 'colorVar' (to be used in plot calls), and a vector of breaks defining the color regions (to be used in densityLegend)</i>
---------------	---

Description

Function to take a numeric vector 'colorVar' and palette 'colorPalette', and return a list containing a vector of color assignments for each element of 'colorVar' (to be used in plot calls), and a vector of breaks defining the color regions (to be used in densityLegend)

Usage

```
splitColorVar(colorVar, colorPalette, breaks = NULL)
```

Arguments

colorVar	the numeric vector of values used to color the points
colorPalette	a color palette. If 'colorPalette' contains, for example, 6 colors, then the values of colorVar will be split and assigned to these 6 colors
breaks	(optional) a numeric vector of two or more unique cut points

Value

a list containing a vector of color assignments ('colorVec') for each element of 'colorVar' (to be used in plot calls), and a vector of breaks ('breaks') defining the color regions (to be used in densityLegend)

Author(s)

Jason Waddell

Index

`addLegend`, [2](#)

`blankPlot`, [3](#)

`colorPoly`, [4](#)

`customRound`, [4](#)

`densityLegend`, [5](#)

`drawSplitDensity`, [6](#)

`findLocations`, [7](#)

`getBreaks`, [8](#)

`oaTemplate`, [8](#)

`oaTheme`, [10](#)

`plotBars`, [11](#)

`plotDensityTrace`, [11](#)

`plotDots`, [12](#)

`plotPolygonRegions`, [13](#)

`pointsOnBoxplot`, [13](#)

`pointsOnBoxplot.default`, [14](#), [14](#)

`pointsOnBoxplot.formula`, [14](#), [16](#)

`prepLegend`, [16](#)

`scatterplotDL`, [18](#)

`splitCircle`, [19](#)

`splitColorVar`, [20](#)