Package: ribiosPlot (via r-universe)

May 18, 2026

Type Package

Title Plotting Module of the 'ribios' Software Suite

Version 1.3.0

Date 2026-02-20

Description Provides data structures and functions for data transformation and visualization in computational biology in drug discovery as part of the 'ribios' software suite. Zhang (2025) <https://github.com/bedapub/ribiosPlot>.

Depends R (>= 3.4.0)

Imports lattice, grDevices, RColorBrewer, grid, gridExtra, graphics, ribiosUtils, stats, ggplot2

Suggests latticeExtra, ribiosArg, plotrix, MASS, testthat, knitr, rmarkdown

Enhances Vennerable

Collate AllClasses.R AllMethods.R utils.R ribiosPlot-package.R robustdist.R p2asterisk.R ellipse.R colors.R fcol.R hist.R jitter.xyplot.R compactTrellis.R cascadePlot.R biosHeatmap.R PCAScoreMatrix.R expVar.R pcaPlots.R plotVenn.R pairs.R figurePanel.R ggSmoothScatter.R openFileDevice.R vennMembers.R

License GPL-3

URL https://github.com/bedapub/ribiosPlot

BugReports https://github.com/bedapub/ribiosPlot/issues

LazyLoad yes

Encoding UTF-8

RoxygenNote 7.3.3

Additional_repositories https://bedapub.r-universe.dev

Repository https://bedapub.r-universe.dev

Date/Publication 2026-02-20 21:11:39 UTC

RemoteUrl https://github.com/bedapub/ribiosPlot

RemoteRef HEAD

RemoteSha 7e00b2c76f1933f414c2319170195588f437bd01

Contents

`[.PCAScoreMatrix` *Subsetting PCAScoreMatrix while keeping the expVar attribute*

Description

Subsetting PCAScoreMatrix while keeping the expVar attribute

Usage

```
## S3 method for class 'PCAScoreMatrix'
x[i, j, ..., drop = TRUE]
```

Arguments

`x`	A `PCAScoreMatrix` object
`i`	Integer or logical vector, subsetting rows
`j`	Integer or logical vector, subsetting columns
`...`	Not used
`drop`	Logical, whether to drop dimensions if only one column is left

Value

A `PCAScoreMatrix` object

`as.data.frame.PCAScoreMatrix` *Coerece a PCAScoreMatrix into data.frame*

Description

Coerece a PCAScoreMatrix into data.frame

Usage

```
## S3 method for class 'PCAScoreMatrix'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)
```

Arguments

`x`	A `PCAScoreMatrix` S3 object.
`row.names`	See `as.data.frame`.
`optional`	See `as.data.frame`.
`...`	See `as.data.frame`

Value

A data.frame consisting of the score matrix

See Also

`as.data.frame`

Examples

```
myPCmat <- PCAScoreMatrix(matrix(rnorm(15),ncol=3), c(0.25, 0.15, 0.1))
as.matrix(myPCmat)
```

`as.matrix.PCAScoreMatrix` *Coerece a PCAScoreMatrix into score matrix*

Description

Coerece a PCAScoreMatrix into score matrix

Usage

```
## S3 method for class 'PCAScoreMatrix'
as.matrix(x, ...)
```

Arguments

x	A PCAScoreMatrix S3 object
...	Currently ignored

Value

A numeric matrix, the score matrix

Examples

```
myPCmat <- PCAScoreMatrix(matrix(rnorm(15),ncol=3), c(0.25, 0.15, 0.1))
as.matrix(myPCmat)
```

---

biosHeatmap *CUSTOMED HEATMAP.2 FOR BIOS USERS*

Description

A tailored version of the heatmap.2 function in the gplots package, by giving the default values in the paramter list.

Usage

```
biosHeatmap(
  x,
  Rowv = TRUE,
  Colv = if (symm) "Rowv" else TRUE,
  distfun = dist,
  hclustfun = function(x) hclust(x, method = "ward.D2"),
  dendrogram = c("both", "row", "column", "none"),
  symm = FALSE,
  scale = c("none", "row", "column"),
  na.rm = TRUE,
  revC = identical(Colv, "Rowv"),
  add.expr,
  breaks,
  symbreaks = min(x < 0, na.rm = TRUE) || scale != "none",
  col = "greenred",
  na.color = "darkgray",
  colsep,
  rowsep,
  sepcolor = "white",
  sepwidth = c(0.05, 0.05),
  cellnote,
  notecex = 1,
  notecol = "cyan",
```

```
  trace = c("none", "column", "row", "both"),
  tracecol = "cyan",
  hline = median(breaks),
  vline = median(breaks),
  linecol = tracecol,
  margins = NULL,
  main = NULL,
  xlab = NULL,
  ylab = NULL,
  labRow = NULL,
  labCol = NULL,
  cexMain = NULL,
  cexRow = pmin(1, 0.2 + 1/log10(nr)),
  cexCol = pmin(1, 0.2 + 1/log10(nc)),
  ColSideColors,
  RowSideColors,
  color.key.title = "Color Key",
  key = TRUE,
  keysize = 1.5,
  density.info = c("none", "histogram", "density"),
  denscol = tracecol,
  symkey = min(x < 0, na.rm = TRUE) || symbreaks,
  densadj = 0.25,
  zlim,
  lhei = c(1, 7),
  lwid = c(1, 7),
  lmat = NULL,
  ...
)
```

Arguments

x	A matrix
Rowv	Logical, whether row-wise dendrogram should be calculated
Colv	Logical, whether column-wise dendrogram should be calculated
distfun	Function, for calculating distance
hclustfun	Function, for hierarchical clustering. By default, the ward.D2 method is used.
dendrogram	Character, specify which dendrogram to be drawn. Note that Rowv and Colv determines whether dendrograms are calculated and rows or columns are re-ordered.
symm	Logical. Should the matrix be treated as symmetric
scale	Logical, should the matrix be row-scaled
na.rm	Logical, should NA values should be omitted
revC	Logical, should columns be reversed
add.expr	Expression
breaks	Numeric vector, where to set breaks. Can be missing.

symbreaks	Logical, should be breaks symmetric
col	Colors for the heatmap, by default green indicates low and red indicates high values
na.color	Color for NA cells, darkgray by default
colsep, rowsep	Integer vector, positions at which columns or rows are separated
sepcolor, sepwidth	Color and width of separating lines
cellnote	Cell labelling
notecex	Cell labelling font size
notecol	Cell labelling font color
trace	Logical , whether drawing tracing lines, by default not
tracecol	Level trace
hline	Level trace hline
vline	Level trace vline
linecol	Level trace color
margins	Margins of labs, automatically guessed if no value was provided
main, xlab, ylab	Heatmap title, X and Y axis labels
labRow, labCol	Row and column labels
cexMain, cexRow, cexCol	Title, row and column label font sizes
ColSideColors, RowSideColors	Column and row side colors
color.key.title	Color key title
key	Logical, whether key should be drawn
keysize	Key size
density.info	Logical, drawing density info in the key histogram, by default not
denscol	Logical, should density information be displayed
symkey	Logical, should the key be symmetric
densadj	densadj
zlim	zlim
lhei	Heights of rows
lwid	Widths of columns
lmat	lmat
...	Other paramters passed to heatmap.2 function

Details

Customed version of the heatmap.2, with the common settings used by JItao David Zhang

Value

See heatmap.2 in the gplots package.

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

Examples

```
set.seed(123)
test <- matrix(rnorm(100), nrow=10)
biosHeatmap(test)

## do not draw row-wise dendrogram
biosHeatmap(test, Rowv=FALSE, dendrogram="column")
## do not draw column-wise dendrogram
biosHeatmap(test, Colv=FALSE, dendrogram="row")
## do not re-sort columns/rows (e.g. for visualization purposes)
biosHeatmap(test, Rowv=FALSE, Colv=FALSE, dendrogram="none")

## define the color range by zlim
biosHeatmap(test, zlim=c(-5, 5))
```

---

brewer.pal.factor *Build brewer.pal colors from factor (Deprecated)*

Description

The functionality has been replaced by fcbrewer. The functions will be removed in the future release.

Usage

```
brewer.pal.factor(factor, name = "Greys")
```

Arguments

factor	A factor vector
name	Color panel name to be passed to brewer.pal

Details

The function is useful to build named RGB color values from factors. brewer.pal.factor return a color-HTML-string vector as the same length of the input factor vector, which is named by the input factor as well. brewer.pal.factorLevels returns a color vector of the length of the factor level, and the colors are named by the levels. See examples below.

From version 1.1-16, the color palette is automatically reduced/expanded when the levels of input factors underlies or exceeds the minimum and maximum colors. See example below.

Value

Named HTML RGB colors.

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

Examples

```
myFac <- factor(c("HSV", "BVB", "FCB", "HSV", "BVB", "HSV"))
brewer.pal.factor(myFac, name="Set1")

myLongFac <- factor(paste("Sample", 1:20))
brewer.pal.factor(myLongFac, name="Set1")

myShortFac <- factor(paste("Sample", 1:2))
brewer.pal.factor(myShortFac, name="Set1")
```

---

cascadeOrder *Order rows of a matrix in the cascade order*

Description

The 'cascade order' is defined by three criteria (1) Rows are divided into two groups by the condition given by 'dichotomy'. (2) The positive and negative rows are ordered respectively so that rows reaching its absolute maximal values in column n are ordered prior to rows reaching reaching its absolute maximal values in columns n+1, where n can be from 1 to column number minus one. (3) If two rows reach the maximum value at the same column, they are ordered by the (decreasing) order of the absolute value in that column.

Usage

```
cascadeOrder(matrix, dichotomy = c("maxabs", "mean", "median"))
```

Arguments

matrix	A numeric matrix
dichotomy	How are the rows divided into two? By maximal abs value (default), mean value, or the median value of each row.

Details

See example for illustration of the idea.

Value

An integer vector of row indices in cascade order.

Examples

```
checkBoard <- function(seed=1887) {
  set.seed(seed)
  mat <- matrix(rnorm(76, sd=1), ncol=4)
  delta <- 3
  for(i in seq(1, 16, 2)) {
    rowInd <- i:(i+1)
    colInd <- (i %/% 2) %% 4 +1
    delta <- ifelse(i>8, -6, 6) * c(0.6, 1)
    mat[rowInd, colInd] <- mat[rowInd, colInd] + delta
  }
  mat[17,1:4] <- rep(-1, 4)
  mat[18,1] <- NA
  mat[18,2] <- mat[18, 2] -6
  mat[19,1:4] <- rep(NA,4)
  rord <- sample(1:nrow(mat), replace=FALSE)
  mat <- mat[rord,]
  rownames(mat) <- sprintf("Row%d", 1:nrow(mat))
  return(mat)
}
myMat <- checkBoard(1887)
biosHeatmap(myMat, Rowv=FALSE, Colv=FALSE, dendrogram="none",
            zlim=c(-4,4), col="royalbluered",
            main="Original matrix")
## since dist by default does not accept rows full of NAs, we remove them in the example below
biosHeatmap(myMat[apply(myMat, 1, function(x) !all(is.na(x))),],
            Rowv=TRUE, Colv=TRUE, dendrogram="both",
            zlim=c(-4,4), col="royalbluered",
            main="hclust/dist clustering")
## note that cascadeOrder handles invariant rows and rows full of NA values
biosHeatmap(myMat[cascadeOrder(myMat),], Rowv=FALSE, Colv=FALSE, dendrogram="none",
            zlim=c(-4,4), col="royalbluered",
            main="Cascade order")
```

`colorpanel` *Generates a set of colors that varies smoothly.*

Description

(copied from the `colorpanel` man page from the `gplots` package. See NOTES below)

Usage

```
colorpanel(n, low, mid, high)
```

Arguments

n	Desired number of color elements in the panel
low	Color to use for the lowest value
mid	Color to use for the middle value. It may be ommited
high	Color to use for the highest value

Details

The values for 'low, mid, high' can be given as color names ('red'), plot color index (2=red), and HTML-style RGB, ("#FF0000"=red).

If 'mid' is supplied, then the returned color panel will consist of 'n - floor(n/2)' HTML-style RGB elements which vary smoothly between 'low' and 'mid', then between 'mid' and 'high'. Note that if 'n' is even, the color 'mid' will occur twice at the center of the sequence.

If 'mid' is omitted, the color panel will vary smoothly beween 'low' and 'high'.

Value

Vector of HTML-style RGB colors.

Note

The colorpanel function is copied from the gplots package (written by Warnes et al.) under the GPL-2 license. The gplots require heavy dependencies that prevent this function being used in speed-sensitive scenarios, e.g. in command-line tools.

Author(s)

Originally by Gregory R. Warnes <greg@warnes.net>. Adapted by Jitao David Zhang <jitao_david.zhang@roche.com>.

References

See gplots package.

See Also

blackyellow and royalbluered for two- and three-color panels.

Examples

```
showpanel <- function(col) {
  image(z=matrix(1:100, ncol=1), col=col, xaxt="n", yaxt="n")
}

oldpar <- par(mfrow=c(3,3))

# two colors only:
showpanel(colorpanel(8,low="red",high="green"))

# three colors
```

```
showpanel(colorpanel(8,"red","black","green"))
# note the duplicatation of black at the center, using an odd
# number of elements resolves this:
showpanel(colorpanel(9,"red","black","green"))

showpanel(greenred(64))
showpanel(redgreen(64))
showpanel(bluered(64))
showpanel(redblue(64))

showpanel(royalbluered(64))
showpanel(royalredblue(64))
par(oldpar)
```

---

compactPar *Compact par setting*

Description

For compact figures

Usage

```
compactPar(mar = c(3, 3, 1.5, 1.5), mgp = c(2, 1, 0), ...)
```

Arguments

mar	marginal option passed to par
mgp	margin line option passed to par
...	other parameters passed to par

Value

A named list of the previous par settings (invisibly), as returned by par.

Author(s)

Jitao David Zhang

See Also

par

Examples

```
compactPar()
plot(1:4)
```

compactTrellis *Return a compact setting for lattice plots, useful for preparing publications*

Description

The function returns a set of lattice options that are useful for compact figures, with less room for padding and therefore more room for the figure. It is often used to prepare for publications.

Usage

```
compactTrellis()
```

Value

A list that can be used in `lattice.options`

Examples

```
opts <- compactTrellis()
```

confEllipse *Plot confidence ellipse based on two-dimenstional data*

Description

Plot confidence ellipse based on two-dimenstional data

Usage

```
confEllipse(x, y = NULL, conf = 0.95, ...)
```

Arguments

`x`	either a matrix of two columns, or a numeric vector
`y`	either a numeric vector of the same length as `x`, or NULL
`conf`	Confidence interval of the ellipse
`...`	Parameters passed to `ellipse`

Value

Invisible coordinates of points on the ellipse

Examples

```
if(interactive()) {
  testX <- rnorm(100, mean=1, sd=2)
  testY <- rnorm(100, mean=2, sd=3)
  plot(testX, testY, pch=16, xlim=c(-5,7), ylim=c(-7,11))
  confEllipse(testX, testY, col="red", lwd=1)
  confEllipse(testX, testY, conf=0.99, col="red", lwd=2)
  confEllipse(testX, testY, conf=0.9, col="red", lwd=0.5)
}

if(interactive() & require("MASS")) {
  testMVR <- mvrnorm(n=100, mu=c(2,3), Sigma=matrix(c(1, 0.65, 0.65, 1), nrow=2))
  plot(testMVR, pch=16, xlim=c(-2,6), ylim=c(0,6))
  confEllipse(testMVR, col="orange")
  confEllipse(testMVR, conf=0.99, col="red")
  confEllipse(testMVR, conf=0.9, col="lightblue")
}
```

`degree2radian` *Convert degree to radian values*

Description

Convert degree to radian values

Usage

```
degree2radian(x)
```

Arguments

x	Degree value

Value

Radian value

Examples

```
degree2radian(90)
degree2radian(-225)
```

display.colorpanels *Display color panels*

Description

Display color panels

Usage

```
display.colorpanels(panel.names, nc)
```

Arguments

panel.names	A vector of character strings, panels to be visualized
nc	Number of color columns

Value

Side effect (visuzalization is used)

Author(s)

Jitao David Zhang

Examples

```
display.colorpanels(threecolor.panels(), 6)
```

display.threecolor.panels *Display three-color panels*

Description

Display three-color panels

Usage

```
display.threecolor.panels(nc = 20)
```

Arguments

nc	Number of columns

Value

Side effect is used

Examples

```
display.threecolor.panels()
```

`display.twocolor.panels` *Display two-color panels*

Description

Display two-color panels

Usage

```
display.twocolor.panels(nc = 20)
```

Arguments

nc	Number of columns

Value

Side effect is used

Examples

```
display.twocolor.panels()
```

`ellipse` *Add an ellipse in an existing plot*

Description

Add an ellipse in an existing plot

Usage

```
ellipse(
  x0 = 0,
  y0 = 0,
  a = 1,
  b = 2,
  alpha = 0,
  length = 1000,
  col = NULL,
  fill = NA,
  border,
  ...
)
```

Arguments

x0	x-coordinate of the ellipse center
y0	y-coordinate of the ellipse center
a	Length of semi-major axis
b	Length of semi-minor axis
alpha	Rotation of the ellipse with regard to the X-axis in radian
length	How many points are generated to simulate the ellipse
col	Ellipse border color
fill	Ellipse fill color
border	Equivalent to col
...	further parameters passed to polygon

Value

Invisible coordinates of points on the ellipse

Examples

```
if(interactive()) {
  plot.new()
  plot.window(xlim=c(-1, 1), ylim=c(-1,1))
  ellipseCols <- heat.colors(11)
  ellipse(0, 0, a=1, b=0.5, alpha=0)
  for(i in 1:11) {
    ellipse(0, 0, a=1, b=0.5, alpha=degree2radian(i*15), col=ellipseCols[i])
  }
  ellipse(0, 0, a=1, b=1, col="black", lwd=2)
  ellipse(0, 0, a=0.5, b=0.5, fill="steelblue")
}
```

---

expVar *S3 function expVar to extract explained variance from prcomp and PCAScoreMatrix objects*

Description

S3 function expVar to extract explained variance from prcomp and PCAScoreMatrix objects

Usage

```
expVar(x, choices)

## S3 method for class 'prcomp'
expVar(x, choices)

## S3 method for class 'PCAScoreMatrix'
expVar(x, choices)
```

Arguments

x	A prcomp or PCAScoreMatrix object.
choices	Either missing, or an integer vector of indices, indicating which PCs should be returned.

Value

A numeric vector of variance explained

Methods (by class)

- expVar(prcomp): Extract explained variance from a prcomp object
- expVar(PCAScoreMatrix): Extract explained variance from a PCAScoreMatrix object

Methods (by class)

- prcomp: Extract explained variance from a prcomp object
- PCAScoreMatrix: Extract explained variance from a PCAScoreMatrix object

Examples

```
myMat <- matrix(rnorm(100), ncol=10)
myPrcomp <- prcomp(myMat)
myPcaScoreMatrix <- pcaScores(myPrcomp, choices=NULL)
expVar(myPrcomp)
expVar(myPcaScoreMatrix)

expVar(myPrcomp, 1:5)
expVar(myPcaScoreMatrix, 1:5)
```

`expVarLabel` *Generic function expVarLabel to generate a label of explained variance from prcomp and PCAScoreMatrix objects*

Description

Generic function expVarLabel to generate a label of explained variance from prcomp and PCAScoreMatrix objects

Usage

```
expVarLabel(x, choices, compact)
```

Arguments

`x`	`prcomp` or `PCAScoreMatrix` Object
`choices`	Integer indices of which PCs to be returned
`compact`	Logical, whether a compact format is returned, see example

Value

A character vector of labels describing the explained variance of each principal component.

`expVarLabel.PCAScoreMatrix` *Labels of principal components from PCAScoreMatrix*

Description

Labels of principal components from PCAScoreMatrix

Usage

```
## S3 method for class 'PCAScoreMatrix'
expVarLabel(x, choices, compact = FALSE)
```

Arguments

`x`	A `PCAScoreMatrix` object
`choices`	Either a logical/integer vector to indicate which PCs to be returned, or `NULL` or missing, in which case all PCs are returned
`compact`	Logical, either a `compact` label is returned, see examples.

Value

A character string vector of the same length as `choices` (or the same length as the column count of the PCAScoreMatrix), which are the labels of the PCs

Examples

```
pcaMat <- PCAScoreMatrix(matrix(rnorm(15),ncol=3), c(0.25, 0.15, 0.1))
expVarLabel(pcaMat)
expVarLabel(pcaMat, choices=1:2)
expVarLabel(pcaMat, choices=1:2, compact=TRUE)
expVarLabel(pcaMat, choices=c(1,3), compact=TRUE)
```

---

expVarLabel.prcomp *Labels of principal components from prcomp*

Description

Labels of principal components from prcomp

Usage

```
## S3 method for class 'prcomp'
expVarLabel(x, choices, compact = FALSE)
```

Arguments

`x`	A `PCAScoreMatrix` object
`choices`	Either a logical/integer vector to indicate which PCs to be returned, or `NULL` or missing, in which case all PCs are returned
`compact`	Logical, either a `compact` label is returned, see examples.

Value

A character string vector of the same length as `choices` (or the same length as the column count of the scores), which are the labels of the PCs

Examples

```
myPr <- prcomp(matrix(rnorm(100), ncol=5))
expVarLabel(myPr)
expVarLabel(myPr, choices=1:2)
expVarLabel(myPr, choices=1:2, compact=TRUE)
```

fcbase *Return base colors of a fcol object*

Description

Return base colors of a fcol object

Usage

```
fcbase(fcol)
```

Arguments

fcol	A fcol object, likely constructed by fcol

Value

A character vector, representing base colors

Examples

```
fc <- fcol(c("lightblue", "orange", "lightblue"), base=c("orange", "lightblue"))
fcbase(fc)
```

fcbase<- *Replace base colors of a fcol object with a different value*

Description

Replace base colors of a fcol object with a different value

Usage

```
fcbase(fcol) <- value
```

Arguments

fcol	A fcol object, likely constructed by fcol
value	A character vector, indicating the new values

Value

A new fcol object

Examples

```
fc <- fcol(c("lightblue", "orange", "lightblue"), base=c("orange", "lightblue"))
fcbase(fc)
fcbase(fc)[1] <- "red"
print(fc)
fcbase(fc) <- c("gray", "darkblue")
fc
```

`fcbrewer` *Factor color brewer*

Description

The function generates a vector of color names for a factor(-like) object.

Usage

```
fcbrewer(factor, panel = "Set1")
```

Arguments

`factor`	A vector of factors. Non-factors will be cast to factors by calling the `factor` function.
`panel`	This parameter can take three types of values: (1) a color set name defined in `brewer.pal.info` in the RColorBrewer package, (2) a function (or the name of a function) that takes an integer as input and returns a vector of colors that will be used as the base colors of levels of the factor, or (3) a character vector which represents the base colors. In the last case, the length of the vector must match the number of levels of the factor. See examples below.

Details

When using `brewer.pal` to generate palettes, the panel is automatically expanded using `colorRampPalette` when the number of levels of the input factor exceeds the limit of respective panel. This is done automatically.

Value

An `fcol` object encoding colors matching the factors as well as the base colors. The latter is often needed in figure legends.

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

See Also

`brewer.pal.info` for color panels.

Examples

```
testFactor <- gl(4,25)
testCol1 <- fcbrewer(testFactor, panel="Set2")
testCol2 <- fcbrewer(testFactor, panel=heat.colors)
testCol3 <- fcbrewer(testFactor, panel="heat.colors")
testCol4 <- fcbrewer(testFactor, panel=c("black", "green", "orange", "lightblue"))

testRan <- runif(100)
## use colors of each item and colors of each level
plot(testRan, pch=21, bg=testCol1)
legend("topright", legend=paste("Class", 1:4),
       pch=21, pt.bg=fcbase(testCol1))

## boxplot uses colors matching to each level only
boxplot(testRan ~ testFactor, col=fcbase(testCol1))
```

`fcol` *Construct a fcol object*

Description

Construct a fcol object

Usage

```
fcol(colors, base)
```

Arguments

`colors`	A character vector, containing colors
`base`	A character vector, containing base colors

Value

A S3 class known as `fcol`, containing colors as vector and base colors in the attribute `fcbase`

Examples

```
fcol(c("lightblue", "orange", "lightblue"), base=c("orange", "lightblue"))
```

figurePanel *Make a figure panel with title*

Description

Make a figure panel with title

Usage

```
figurePanel(gg, title)
```

Arguments

gg	A grob object
title	Character, title of the plot panel, e.g. A, (B), etc.

Value

A grob object

Examples

```
require("ggplot2")
df <- data.frame(
  gp = factor(rep(letters[1:3], each = 10)),
  y = rnorm(30)
)

ds <- do.call(rbind, lapply(split(df, df$gp), function(d) {
  data.frame(mean = mean(d$y), sd = sd(d$y), gp = d$gp)
  }))

g1 <- ggplot(df, aes(gp, y)) +
  geom_point() +
  geom_point(data = ds, aes(y = mean), colour = 'red', size = 3)

g2 <- ggplot() +
  geom_point(data = df, aes(gp, y)) +
  geom_point(data = ds, aes(gp, mean), colour = 'red', size = 3) +
  geom_errorbar(
    data = ds,
    aes(gp, mean, ymin = mean - sd, ymax = mean + sd),
    colour = 'red',
    width = 0.4
  )

panelA <- figurePanel(g1, "(A)")
panelB <- figurePanel(g2, "(B)")
layoutMat <- matrix(c(1,2), nrow=1)
```

```
gridExtra::grid.arrange(grobs=list(panelA, panelB),
 layout_matrix=layoutMat)
```

`getDefaultFontFamily` *Get default font family*

Description

Get default font family

Usage

```
getDefaultFontFamily()
```

Value

Character string, the default font family

`getExpVarLabel` *Helper function to print PC and explained variances*

Description

Helper function to print PC and explained variances

Usage

```
getExpVarLabel(ev, choices, compact = FALSE)
```

Arguments

`ev`	A numeric vector of explained variances
`choices`	An integer vector to indicate which PCs to be returned. If `NULL` or `NA` or missing, all elements are returned.
`compact`	Logical, either a `compact` label is returned, see examples.

Value

A character vector of labels in the form `"Principal component N (X% variance explained)"`.

getLims *Get xlim/ylim ranges for plots from real values*

Description

Get xlim/ylim ranges for plots from real values

Usage

```
getLims(..., perc = 0.99, symm = TRUE)
```

Arguments

...	one or more vectors of real values
perc	percentage of dynamic range that should be covered by the limits; if set to 1 the whole range is used.
symm	logical value; if set to TRUE, the range will be symmetric around zero

Value

A numeric vector of length 2 giving the lower and upper axis limits.

Examples

```
myX <- rnorm(100, mean=1)
myY <- rnorm(100)
myLim <- getLims(myX, myY, perc=0.99)
plot(myX, myY, xlim=myLim, ylim=myLim)
mySymmLim <- getLims(myX, myY, perc=0.99, symm=TRUE)
plot(myX, myY, xlim=myLim, ylim=mySymmLim)
```

ggSmoothScatter *Mimicking the graphics::smoothScatter behaviour for GGally::ggpairs*

Description

Mimicking the graphics::smoothScatter behaviour for GGally::ggpairs

Usage

```
ggSmoothScatter(
  data,
  mapping,
  colours = colorRampPalette(c("white", blues9[5:9], "black"))(256),
  xlim = NULL,
  ylim = NULL,
  ...
)
```

Arguments

data	Data to be visualized, normally not directly set by the user
mapping	Data mapping, normally not directly set by the user
colours	Colours to be used
xlim	NULL or a vector of two numbers
ylim	NULL or a vector of two numbers
...	Other parameters passed to stat_density_2d

Note

So far the outliers are not plotted, to be done later

---

ggSmoothScatterWithAux

Mimicking the graphics::smoothScatter behaviour for GGally::ggpairs, with aux lines

---

Description

Mimicking the graphics::smoothScatter behaviour for GGally::ggpairs, with aux lines

Usage

```
ggSmoothScatterWithAux(
  data,
  mapping,
  colours = colorRampPalette(c("white", blues9[5:9], "black"))(256),
  xlim = NULL,
  ylim = NULL,
  ...
)
```

Arguments

data	Data to be visualized, normally not directly set by the user
mapping	Data mapping, normally not directly set by the user
colours	Colours to be used
xlim	NULL or a vector of two numbers
ylim	NULL or a vector of two numbers
...	Other parameters passed to stat_density_2d Compared with ggSmoothScatter,

---

guessWH *Guess width and height parameters for a heatmap*

Description

guessWH helps biosHeatmap determines a proper canvas dimension as well as the proportion between legends and the main figure, especially in the command line mode.

Usage

```
guessWH(
  nrow,
  ncol,
  rownames,
  colnames,
  cexRow,
  cexCol,
  xlab,
  ylab,
  width,
  height
)
```

Arguments

nrow	Row count of the matrix to be visualized
ncol	Column count of the matrix to be visualized
rownames	Row names of the matrix, helping to determine horizontal margins. Can be missing or set as NULL
colnames	Column names of the matrix, helping to determine vertical margins. Can me missing or set as NULL
cexRow	Row name font size. Can be missing or NA.
cexCol	Column name font size. Can be missing or NA.
xlab	X-axis (column side) name. Character string. Can be missing or NA

ylab	Y-axis (row side) name. Character string. Can be missing or NA
width	Width suggested by the user. Can be NA
height	Height suggested by the user. Can be NA

Details

guessWH determines for visual purposes the best height/width and legend/figure proportion for heatmaps. Interested users are invited to read the codes to get insights how the task is done.

Value

A list of width proportions, height proportions, the total width and the total length.

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

See Also

biosHeatmap

Examples

```
myMat <- matrix(rnorm(256), nrow=16)
rownames(myMat) <- sample(paste(letters, LETTERS, sep="_"),16)
colnames(myMat) <- sample(paste(LETTERS, letters, sep="_"), 16)
guessWH(nrow=nrow(myMat), ncol=ncol(myMat), width=4, height=4)
guessWH(nrow=nrow(myMat), ncol=ncol(myMat), width=NA, height=NA)
myWH <- guessWH(nrow=nrow(myMat), ncol=ncol(myMat),
xlab="321", ylab="ABC",
rownames=rownames(myMat), colnames=colnames(myMat))

if(interactive()) {
X11(width=myWH$width, height=myWH$height)
biosHeatmap(myMat, lwid=myWH$lwid, lhei=myWH$lhei, xlab="321",
ylab="ABC", cexRow=2L, cexCol=2L)
dev.off()
}
```

---

histMat *Make histograms for matrix*

Description

Make histograms for matrix

Usage

```
histMat(
  mat,
  linesOpt = list(lwd = NULL, col = NULL, lty = NULL, type = NULL, pch = NULL),
  main = NULL,
  xlab = NULL,
  xlim = NULL,
  ...
)
```

Arguments

mat	A numerical matrix
linesOpt	Line options
main	Title text
xlab	Xlab
xlim	Xlim
...	Other parameters passed to hist

Value

Invisibly, a list as returned by hist, with additional elements xlim and linesOpt.

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

See Also

hist

Examples

```
testMat <- matrix(rnorm(1000), nrow=100)
histMat(testMat)
```

---

idev *Interative dev.print and pdf print*

Description

Execute dev.print only if R session is interactive.

Usage

```
idev(...)

ipdf(file, ...)
```

Arguments

...	Parameters passed to dev.print
file	PDF file name

Details

ipdf is a shortcut in case PDF is used as the device, with the twist that useDingbats is set to FALSE by default. See NOTE.

dev.print will make a R-script fail if the session is not interactive (e.g. when the script is executed with the -f option from R command line). Function idev checks first whether the session is interative, and executes dev.print only if the session is interactive.

A commonly used shortcut is ipdf, which prints the current device to a PDF file.

Value

Side effect used.

Note

useDingbats is set to FALSE in ipdf. Setting the option to TRUE causes problem in importing the PDF to Inkscape, a vector-based figure modifying software. Though the option may reduces smaller and (according to the R manual) better output, we have noticed no difference.

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

See Also

dev.print, pdf

Examples

```
tmfile <- tempfile()
plot(1:15, type="h")
idev(png, tmfile,width=600, height=800)
ipdf(tmfile)
```

intRange *Return a range defined by integers*

Description

The function is similar to range but returns integer ranges that are just outside the real range: i.e. the floor of the left range and the ceiling of the right range.

Usage

```
intRange(x, na.rm = TRUE)
```

Arguments

x	A numeric vector
na.rm	Logical, whether NA should be removed

Value

A vector of integers of length 2.

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

Examples

```
intRange(rnorm(100))
```

jitter.xyplot *Make boxplots or dotplots with sample-size proportional jitters*

Description

Make boxplots or dotplots with jitters of the size proportional to the sample size. See examples.

Usage

```
jitter.xyplot(x, y, N = 20, factor = 1, ...)
```

Arguments

x	X-axis variable, numeric or factor
y	Y-axis variable, numeric
N	Number of groups that y-axis should be cut
factor	Jitter factor, passed to jitter
...	Other parameters passed to panel.xyplot

Value

Side effects are used.

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

Examples

```
library(lattice)

testX <- gl(8,5)
testY <- rnorm(40)
xyplot(testY ~ testX)
xyplot(testY ~ testX, panel=jitter.xyplot)

(xyBw <- bwplot(testY ~ testX))
(xyBwJitter <- update(xyBw, panel=jitter.xyplot))

testXnum <- rep(1:8, 5)
xyplot(testY ~ testXnum, panel=jitter.xyplot)
```

---

midCol *Blender two colors to get the midpoint color of two colors*

Description

Blender two colors to get the midpoint color of two colors

Usage

```
midCol(col1, col2)
```

Arguments

col1	Character string, represting the first color. It can be of length 2 or 1; in the former case, col2 should be missing
col2	Character string, represting the second color

Value

The midpoint color of the two in the Lab space.

Examples

```
midCol("black", "red")
midCol(c("black", "red"))

set.seed(1778)
nCol <- 20
candCol <- grep("gr[a|e]y", colors(), value=TRUE, invert=TRUE)
firstCols <- sample(candCol, nCol)
secondCols <- rev(sample(candCol, nCol))
midCols <- sapply(seq(along=firstCols), function(i)
  midCol(firstCols[i], secondCols[i]))
plot.new()
plot.window(xaxt="n", yaxt="n", xlim=c(0, nCol),
     ylim=c(0.5, 4), bty="n")
title("Example of midCol")
segments(x0=1:nCol, y0=0, x1=1:nCol, y1=4, col="lightgray")
points(x=rep(1:nCol, each=3),
     y=rep(1:3, nCol),
     pch=21, cex=1.75,
     bg=as.vector(rbind(firstCols, midCols, secondCols)))
text(0, c(1.5, 2.5, 3.5), c("Second", "Midpoint", "First"),
     pos=4)
```

---

nonNull *Make sure that x is assigned a reasonable value*

Description

Make sure that x is assigned a reasonable value

Usage

```
nonNull(x, default, length = NULL, defaultNULL.ok = FALSE)
```

Arguments

x	Any vector
default	A default value
length	Desired length
defaultNULL.ok	Logical, whether the default can be NULL or not

Value

non-null values

`openFileDevice` *Open a device as a file preparing for plotting in the file*

Description

The function `openFileDevice` opens a device of the type specified by the file extension name. It such prepares the file for visualizing data. User must call `dev.off` once the writing (plotting) to the device is finished.

Usage

```
openFileDevice(filename, width = 7, height = 7, dpi = 300L, family)
```

Arguments

`filename`	Character, file name to be written to. The type of file is determined by the extension. See details below.
`width`	Number, figure width of the file in *inch*.
`height`	Number, figure height of the file in *inch*.
`dpi`	Number, resolution as "dots per inch". For publication 300dpi is usually enough.
`family`	Font family name. Only applicable to PDF files

Details

`closeFileDevice` quietly closes the current device: it does not print the information of the next device.

The function `openFileDevice` calls `extname` to determine the file type to be drawn in. Currently supported types include `PDF`, `tiff` (`tif`), `bmp`, `jpeg` (`jpeg`). When the file type is not recognized, the `PDF` format is used as a fallback.

As an example, `myplot.pdf` will triggers openning a PDF device, `newplot.png` a PNG device, and `oldplot.tiff` a TIFF device, whereas `myfile.abc` will be openned as a PDF device.

For bitmap files like `BMP`, `JPEG`,`PNG` and `TIFF`, we use `inch` as the size unit in order to be compatible with PDF. And the resolution is always set to 300dpi.Furthermore, JPEG quality is set to 90 instead of the default value 75, and TIFF do not use any compression. These settings follow our practices for scientific publication while allowing generic post-precessing of figures.

Value

Both functions are used for its side effect.

Note

After plotting, user should call `dev.off` to close the device in the file, otherwise the file can probably not be read.

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

See Also

`extname` for getting extension name of file. See `pdf`, `png`, `jpeg`, `tiff` and `bmp` for file formats.

Examples

```
if(interactive()) {
  tempfile1 <- paste(tempfile(), ".pdf", sep="")
  openFileDevice(tempfile1)
  plot(rnorm(100), rnorm(100))
  closeFileDevice()

  tempfile2 <- paste(tempfile(), ".png", sep="")
  openFileDevice(tempfile2, width=5, height=5)
  plot(rnorm(100), rnorm(100))
  closeFileDevice()
}
```

---

p2asterisk *Convert p values into asterisks*

Description

The function map p values into asterisks by common definitions

Usage

```
p2asterisk(p, use0.1 = TRUE)
```

Arguments

p	Numerical, p values (between 0 and 1), can be a matrix or more generally an array
use0.1	Logical, whether a dot should be displayed if 0.05<p<0.1

Value

A character vector of the same length as p of asterisk symbols. In case p is an array, both ‘dim‘ and ‘dimnames‘ properties are copied to the returning value.

Note

NA will be mapped to empty strings.

Examples

```
myPvals <- c(0.0005, 0.02, 0.4, 0.075, NA, 0.0044)
myPasterisks <- p2asterisk(myPvals, use0.1=FALSE)
stopifnot(identical(myPasterisks, c("***", "*", "", "", "", "**")))

myPasterisks2 <- p2asterisk(myPvals, use0.1=TRUE)
stopifnot(identical(myPasterisks2, c("***", "*", "", ".", "",  "**")))
```

---

panel.cor *Correlation panel for pairs*

Description

Correlation panel for pairs

Usage

```
panel.cor(x, y, digits = 2, prefix = "", cex.cor, ...)
```

Arguments

x	A numeric vector
y	A numeric vector, must be of the same length as x.
digits	Integer, number of digits to show
prefix	Prefix of the label
cex.cor	Numeric, if missing, automatically guessed
...	Passed to cor.
	This function can be used with pairs to display correlations.

Value

No return value, called for side effects as a panel function in pairs.

See Also

pairs.

`panel.lmSmooth` *Correlation panel*

Description

Correlation panel

Usage

```
panel.lmSmooth(
  x,
  y,
  col = par("col"),
  bg = NA,
  pch = par("pch"),
  cex = 0.8,
  method = "spearman",
  use = "complete",
  ...
)
```

Arguments

`x`	A numeric vector
`y`	A numeric vector, must be of the same length as `x`.
`col`	Color
`bg`	Background
`pch`	Ponit symbol
`cex`	Font size
`method`	Correlation method, passed to `cor`
`use`	passed to `cor`
`...`	Passed to `panel.smooth` This function can be used with `pairs` to display correlations.

Value

No return value, called for side effects as a panel function in `pairs`.

See Also

`pairs`.

pcaRotation *Retrieve PCA rotations from prcomp objects*

Description

Retrieve PCA rotations from prcomp objects

Usage

```
pcaRotation(x, choices, offset, reverse = c(FALSE, FALSE))
```

Arguments

x	An object of prcomp
choices	Integer vector, indices of principal components, default the first two PCs. If missing, NULL or NA, all PCs are returned.
offset	Oither one or more rows's names in the rotation matrix, or indices, or a logical vector. The average of the rows specified by offset is set to zero.
reverse	Logical of the same length as choices or 1 (which will be repeated), indicating whether the sign of values in the indexed axis should be reversed

Examples

```
testMatrix <- matrix(rnorm(1000), nrow=10)
testPCA <- prcomp(testMatrix)
testPCAscores <- pcaScores(testPCA)

testPCAscores.withOffset <- pcaScores(testPCA, offset=c(1,3,5))
## notice the average of offset-rows are near zero
colMeans(as.matrix(testPCAscores.withOffset)[c(1,3,5),])

testPCAscores.withReverse <- pcaScores(testPCA, reverse=c(TRUE, FALSE))
colMeans(as.matrix(testPCAscores.withReverse)[c(1,3,5),])
```

PCAScoreMatrix *Construct a S3-class PCAScoreMatrix object*

Description

Construct a S3-class PCAScoreMatrix object

Usage

```
PCAScoreMatrix(scoreMatrix, expVar)
```

Arguments

scoreMatrix	Numeric matrix, objects in rows and PCs in columns
expVar	Numeric vector, length must equal the number of columns of scoreMatrix, explained variance by respective PCs

Value

A S3-class PCAScoreMatrix object, which is a score matrix with explained variances (expVar) as attribute.

See Also

as.matrix.PCAScoreMatrix, expVar.PCAScoreMatrix, print.PCAScoreMatrix. This function is usually not called by the end user; instead, it is used by the function pcaScores

Examples

```
myPCmat <- PCAScoreMatrix(matrix(rnorm(15),ncol=3), c(0.25, 0.15, 0.1))
myPCmat
```

---

pcaScores *Retrieve PCA scores from prcomp objects*

Description

Retrieve PCA scores from prcomp objects

Usage

```
pcaScores(x, choices, offset, reverse = c(FALSE, FALSE))
```

Arguments

x	An object of prcomp
choices	Integer vector, indices of principal components, default the first two PCs. If missing, NULL or NA, all PCs are returned.
offset	Oither one or more rows's names in the loading matrix, or indices, or a logical vector. The average loading of the rows specified by offset is set to zero.
reverse	Logical of the same length as choices or 1 (which will be repeated), indicating whether the sign of values in the indexed axis should be reversed

Value

A PCAScoreMatrix object containing the PCA scores.

Examples

```
testMatrix <- matrix(rnorm(1000), nrow=10)
testPCA <- prcomp(testMatrix)
testPCAscores <- pcaScores(testPCA)

testPCAscores.withOffset <- pcaScores(testPCA, offset=c(1,3,5))
## notice the average of offset-rows are near zero
colMeans(as.matrix(testPCAscores.withOffset)[c(1,3,5),])

testPCAscores.withReverse <- pcaScores(testPCA, reverse=c(TRUE, FALSE))
colMeans(as.matrix(testPCAscores.withReverse)[c(1,3,5),])
```

---

pcaScoresFromLogFC *Perform principal component analysis and derive PCA scores from a logFC matrix*

Description

Perform principal component analysis and derive PCA scores from a logFC matrix

Usage

```
pcaScoresFromLogFC(lfcMat, reference = 0, choices, reverse = c(FALSE, FALSE))
```

Arguments

lfcMat	Log fold-chnage matrix, genes (features) in rows and samples in columns
reference	The reference value that should be set as 0 in the scores, default: 0
choices	How many PCs should be retured. Passed to pcaScores
reverse	Whether the axes should be reversed. Passed to pcaScores
	Perform PCA and get scores from a logFC matrix by setting a pseudo profile of no change (0 for all features) at the origin point

Value

A PCAScoreMatrix object containing the PCA scores derived from the log fold-change matrix.

Examples

```
lfcMat <- matrix(rnorm(9), nrow=3)
pcaScoresFromLogFC(lfcMat)
```

pdf2png *Use 'convert' (ImageMagick) to convert PDF to high-quality PNG*

Description

The function makes a system call to convert PDF files to high-quality (300 dpi) PNG files.

Usage

```
pdf2png(
  ...,
  convert = "convert",
  density = 300,
  outdir = NULL,
  outfile = NULL,
  wait = FALSE
)
```

Arguments

...	PDF files
convert	Name of the convert program. It is overwritten if the program is running on the udis machine (rbaus024). See the code for more details.
density	DPI. Default 300 is good enough for publications in most biology/medicine journals
outdir	Output directory. If the value is NULL, the output files will be written in the same directory as the input file
outfile	Output file names. If the value is NULL, the output file names will be basename of the input PDF files appended with the .png suffix. If given, its length must equal the length of PDF files.
wait	Logical, should the function wait until the conversion is finished?

Value

Output file names are returned invisibly.

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

Examples

```
tmpdir <- tempdir()

pdffile <- file.path(tmpdir, "test-plot.pdf")
pdf(pdffile)
```

```
plot(1:5)
dev.off()

pngfile <- file.path(tmpdir, "test.png")
pdf2png(pdffile)
pdf2png(pdffile, outfile=pngfile)

testfile <- system.file("/doc/intro.pdf", package="limma")
if(file.exists(testfile)) {
  pdf2png(testfile, outdir=tmpdir)
  Sys.sleep(1)
  dir(tmpdir)

 ## or waiting
  pdf2png(testfile, outdir=tmpdir, wait=TRUE)
}
```

`plotPCA` *S3 method plotPCA*

Description

S3 method plotPCA

Usage

```
plotPCA(x, choices, ...)
```

Arguments

`x`	A prcomp object
`choices`	Integer index, choices to plot
`...`	Other parameters

Value

Depends on the method; see individual method documentation.

plotPCA.prcomp *Visualise PCA results of expression data with the sample plot*

Description

plotPCA is designed to visualize sample relationships revealed by PCA analysis of high-dimensional expression data. It is adapted from the biplot function in the stats package, with functionalities useful for sample visualization and labelling, and removing the visualization of features (usually genes) in the input matrix. The rationale is that in most cases there are too many features to provide an informative visualization.

Usage

```
## S3 method for class 'prcomp'
plotPCA(
  x,
  choices = c(1, 2),
  text = FALSE,
 points = list(pch = NULL, col = NULL, cex = NULL, bg = NULL, lwd = NULL, lty = NULL,
    order = NULL),
  arrows = FALSE,
  grid = FALSE,
  abline = FALSE,
  xlim = NULL,
  ylim = NULL,
  xlab = NULL,
  ylab = NULL,
  offset,
  main = NULL,
  reverse = c(FALSE, FALSE),
  ...
)
```

Arguments

x	prcomp object produced by the prcomp function
choices	An integer vector of length 2, indicating which PCs to visualize. By default the first (X-axis) and second (Y-axis) are visualized
text	A logical value or a list of options to label samples. See Details.
points	A logical value or a list of options to pinpoint samples. See details.
arrows	A logical or a list of options to draw arrows.
grid	Logical value, indicating whether grid lines should be added to the plot.
abline	A logical or a list of options to draw abline
xlim	xlim of the plot. Automatically determined if missing.
ylim	ylim of the plot. Automatically determined if missing.

xlab	xlab of the plot. If missing, the PC and the explained variability are shown.
ylab	ylab of the plot. If missing, the PC and the explained variability are shown.
offset	Offset should be either one or more rows's names in the loading matrix, or indices, or a logical vector. The average loading of the rows specified by offset is set to zero.
main	Title of the plot
reverse	ogical of length 2 or 1 (which will be repeated to 2), indicating whether the sign of values in the 1st/2nd axis should be reversed.
...	Other parameters passed to plot.window

Details

The values for text, points and arrows can be

1. Logical. If FALSE, no text or point is added.
2. List. A list containing options passed to text, points, and arrows respectively, such as col, cex, lwd, lty, code, length,angle, and pos (only for text). order decides in what order are the points drawn, which can be useful when there are points to be drawn 'above' other points.
3. A vector of character strigns (only for text)

See examples below.

Value

The value of the rotated data, namely the centered (and scaled if requested) data multiplied by the rotation matrix.

Note

prcomp should be called with retx=TRUE, which is the default behaviour.

See Also

prcomp and pcaScores

Examples

```
testVal <- matrix(rnorm(10000), nrow=500)
colnames(testVal) <- paste("Sample", 1:ncol(testVal), sep="")
rownames(testVal) <- paste("Gene",1:nrow(testVal), sep="")

testPCA <- prcomp(t(testVal), center=TRUE, scale=TRUE)


plotPCA(testPCA)

plotPCA(testPCA, points=FALSE, text=TRUE, grid=TRUE)

pointsList <- list(col=1:3, bg=21,pch=22, cex=4:1, lwd=1:2, lty=2:4)
```

```
textList <- list(col=c("orange", "royalblue"), cex=1.2, srt=15, pos=1)

plotPCA(testPCA, choices=c(1,2), grid=TRUE, points=pointsList,
        text=TRUE)

## visualize dimension 1:3
rop <- par(mfrow=c(1,2), pty="s")
plotPCA(testPCA, choices=c(1,2), grid=TRUE, points=pointsList, text=textList)
plotPCA(testPCA, choices=c(2,3), grid=TRUE, points=pointsList, text=textList)
par(rop)
```

`plotPCAloading` *Plot PCA loading*

Description

Plot PCA loading

Usage

```
plotPCAloading(
  loadings,
  x = 1L,
  y = 2L,
  circle = FALSE,
  title = "",
  subtitle = "",
  ...
)
```

Arguments

`loadings`	PCA loadings
`x`	Integer, which loading is to be visualized on the X axis?
`y`	Integer, the loading to be visualized on the X axis.
`circle`	Logical, whether draw circle or not
`title`	Character string
`subtitle`	Character string
`...`	Passed to `plot`

Value

No return value, called for side effects (plotting).

plotVenn *Plot Venn object of the Vennerable package*

Description

The function plots Venn objects from the Vennerable package with custom margins that better suit publication figures.

Usage

```
plotVenn(
  venn,
  main = "",
  show = list(FaceText = "weight", Universe = FALSE),
  ...
)
```

Arguments

venn	Venn object from the Vennerable package
main	Figure title
show	Named list controlling which elements to display. Supported elements: `Universe` (logical), `Sets` (logical), `SetLabels` (logical), `DarkMatter` (logical), `Faces` (logical), `FaceText` (character, e.g. `"weight"`).
...	Other parameters passed to `Vennerable::compute.Venn`

Value

Side effect is used - a plot is generated.

Note

The function requires Vennerable package version 3.0 or later. Plotting logic adapted from Vennerable (GPL license).

Examples

```
if (requireNamespace("Vennerable", quietly = TRUE)) {
  myVenn <- list(A = LETTERS[1:24], B = LETTERS[3:8], C = LETTERS[5:9])
  plotVenn(Vennerable::Venn(myVenn), main = "Letters")
}
```

print.fcol *Print a fcol object*

Description

Print a fcol object

Usage

```
## S3 method for class 'fcol'
print(x, ...)
```

Arguments

x	A fcol object, likely constructed by fcol
...	Not used now

Value

The input x, invisibly.

Examples

```
fc <- fcol(c("lightblue", "orange", "lightblue"), base=c("orange", "lightblue"))
fc
```

print.PCAScoreMatrix *Print PCAScoreMatrix*

Description

Print PCAScoreMatrix

Usage

```
## S3 method for class 'PCAScoreMatrix'
print(x, ...)
```

Arguments

x	A PCAScoreMatrix S3-object
...	Ignored

Value

The input x, invisibly.

Examples

```
myPCmat <- PCAScoreMatrix(matrix(rnorm(15),ncol=3), c(0.25, 0.15, 0.1))
myPCmat
```

---

qBreaks *Internal function to re-calculate breaks of histograms when x-axis is clipped*

Description

The function calculates the new break numbers caused by the clipping of x axis. This is usally larger than the original number of breaks .

Usage

```
qBreaks(x, quantiles = c(0, 0.99), breaks = 100)
```

Arguments

x	Value to draw histgrams with
quantiles	Quantiles of x that determine the clip boundary of x-axis
breaks	Integer, number of breaks applied to original data

Value

Integer: number of breaks

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

See Also

This function is directly used by qHist

Examples

```
testVal <- rnorm(1000)
qBreaks(testVal, quantiles=c(0.25, 0.75), breaks=100)
```

qHist *Histogram with quantile line(s) and text(s)*

Description

A handy function to plot histogram with display elements of quantile.

Usage

```
qHist(x, quantiles = 0.25, breaks = 100, qlty = 2, qlwd = 2, qcol = "red", ...)
```

Arguments

x	Value to draw the histogram
quantiles	Numeric values or NULL; in case of numeric values, at the corresponding quantile values vertical lines and text labels are drawn; if set to NULL, no extra items will display. See examples below.
breaks	Integer, number of breaks
qlty	Type of vertical quantile lines
qlwd	Width of vertical quantile lines
qcol	Color of vertical quantile lines
...	Other parameters that are passed to hist

Details

The appends vertical lines and texts to histograms produced by hist. This can be useful in unspecific filtering of expression data.

Value

The object returned by the hist function, with an extra item named quantiles.

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

See Also

hist

Examples

```
testVal <- rnorm(1000)
hist(testVal)
qHist(testVal, quantiles=c(0.25, 0.75), border="lightgray")
```

radian2degree *Convert radian to degree values*

Description

Convert radian to degree values

Usage

```
radian2degree(x)
```

Arguments

x	Radian value

Value

Degree value

Examples

```
radian2degree(2*pi)
radian2degree(-.5*pi)
```

robustDist *Robust distance*

Description

A wrapper function for the `dist` function in the `stats` package. It replaces NA values in the distance matrix by the maximum distance in the same matrix, therefore prevents cases where hclust fails because of `NA` distances.

Usage

```
robustDist(x, ...)
```

Arguments

x	a numeric matrix, data frame or 'dist' object
...	Other parameters passed to the `dist` function.

Details

In the rare case of all-NA distance matrices, all values are assigned arbitrarily to one.

Value

The same as dist, however without NAs.

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

Examples

```
mymat <- matrix(c(3,2,1,NA,NA,NA,
                4,1,2,NA,NA,NA,
                NA,NA,NA,5,1,2), ncol=6, byrow=TRUE)
dist(mymat)
robustDist(mymat)
```

---

royalbluered *Two and three-color panels*

Description

Two and three-color panels

Usage

```
royalbluered(n)

royalredblue(n)

royalbluegrayred(n)

royalredgrayblue(n)

blackyellow(n)

yellowblack(n)

whiteblue(n)

whitered(n)

blackred(n)

blackgreen(n)

whiteblack(n)
```

blackwhite(n)

turyeb(n)

redgreen(n)

greenred(n)

bluered(n)

redblue(n)

blueblackred(n)

cyanblackyellow(n)

yellowblackcyan(n)

redblackblue(n)

blackredyellow(n)

blackgoldred(n)

magentayellow(n)

yellowmagenta(n)

whiteblueblackheat(n)

heat(n)

Arguments

n	Number of colors needed

Value

Character vector of length n coding colors

See Also

blackyellow for two-color systems

Examples

```
display.threecolor.panels()
```

setCompactTrellis *Set compact trellis as default*

Description

The function sets compact trellis options as default. The previous `lattice.options` are saved and restored via `on.exit` when the calling function exits.

Usage

```
setCompactTrellis()
```

Value

Invisibly, the previous value of the `default.theme` lattice option, so it can be restored manually if needed.

Examples

```
old <- setCompactTrellis()
```

squareLayout *Plan a square/matrix layout of plots*

Description

Plan a square/matrix layout of plots

Usage

```
squareLayout(n)
```

Arguments

`n`	Number of plots

Value

A vector of integers of length 2. Can be passed to `layout` or `mfrow` in `par` to make the layout.

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

Examples

```
op <- par(mfrow=squareLayout(7))
plot(1:5)
plot(2:6)
plot(3:7)
plot(-9:-4)
plot(8:5)
plot(5:1)
plot(1:9)
par(op)
```

`symrange` *Return a symmetric range*

Description

Return a symmetric range

Usage

```
symrange(x, mid = 0)
```

Arguments

`x`	A numeric vector
`mid`	Number, the mid point

Value

A vector of two numbers, a symmetric range with `mid` in the middle

---

`threecolor.panels` *Return available three-color panels*

Description

Return available three-color panels

Usage

```
threecolor.panels()
```

Value

A vector of character strings

twocolor.panels *Return available three-color panels*

Description

Return available three-color panels

Usage

```
twocolor.panels()
```

Value

A vector of character strings

vennMembersDataframe *Extract members of each region in Venn diagrams in to a data.frame*

Description

Extract members of each region in Venn diagrams in to a data.frame

Usage

```
vennMembersDataframe(venn)
```

Arguments

venn	A Venn object

Value

A data.frame containing logical values of sets and elements

Examples

```
if(requireNamespace("Vennerable")) {
  myList <- list(A=LETTERS[1:5], B=LETTERS[2:7], C=LETTERS[seq(2,9,2)])
  myVenn <- Vennerable::Venn(myList)
  myVennDf <- vennMembersDataframe(myVenn)
  print(myVennDf)
}
```

vennMembersList *Extract members of each region in Venn diagrams in to a list*

Description

Extract members of each region in Venn diagrams in to a list

Usage

```
vennMembersList(venn, removeNULL = TRUE)
```

Arguments

venn	A Venn object
removeNULL	Logical, whether NULL elements should be removed

Value

A named list of members, each list item corresponding to a region in Venn diagrams

Examples

```
if(requireNamespace("Vennerable")) {
  myList <- list(A=LETTERS[1:5], B=LETTERS[2:7], C=LETTERS[seq(2,9,2)])
  myVenn <- Vennerable::Venn(myList)
  myVennList <- vennMembersList(myVenn)
}
```

xclipHist *Histogram with clipped x axis*

Description

Draw histograms with clipped x axis; the clipping is determined by quantiles of x values.

Usage

```
xclipHist(
  x,
  xclip = c(0.01, 0.99),
  breaks = 100,
  quantiles = 0.25,
  qlty = 2,
  qlwd = 2,
  qcol = "red",
  ...
)
```

Arguments

x	Value to draw the histogram
xclip	Quantiles of x-values that should be displayed; values outside of this range are not shown in the histogram
breaks	A integer number indicating how many breaks should the *original unclipped* data have; the function will automatically re-calculate the breaks of the clipped data so that they look consistent.
quantiles	Numeric values or NULL; in case of numeric values, at the corresponding quantile values vertical lines and text labels are drawn; if set to NULL, no extra items will display. See examples below.
qlty	Type of vertical quantile lines
qlwd	Width of vertical quantile lines
qcol	Color of vertical quantile lines
...	Other parameters that are passed to hist

Details

The function clips (subsets) x-axis and recalcualte the breaks so that the clipped image looks like a real subset of the original data.

Value

The object returned by the hist function, with an extra item named quantiles.

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

See Also

qHist, which draws quantile line and texts onto histograms.

Examples

```
testVal <- c(rnorm(1000),10)
hist(testVal, breaks=100)
xclipRes <- xclipHist(testVal, xclip=c(0.001, 0.999), quantiles=0.50)

xclipRes$quantiles
```

Index