Package: henna (via r-universe)

November 10, 2026

Type Package

Title A Versatile Visualization Suite

Version 0.8.5

Description A visualization suite primarily designed for single-cell RNA-sequencing data analysis applications but well-suited for other purposes as well. It introduces novel plots to represent two-variable and frequency data and optimizes some commonly used plotting options (e.g., correlation, network, density, alluvial and volcano plots) for ease of usage and flexibility.

License MIT + file LICENSE

Imports abdiv, dplyr, ggalluvial, ggeasy, ggforce, ggraph, ggnewscale, ggplot2, ggrepel, grDevices, liver, methods, paletteer, reshape2, rlang, stats, tidygraph, viridis, withr

Encoding UTF-8

RoxygenNote 7.3.3

Suggests EnhancedVolcano, qs2, testthat (>= 3.0.0)

URL https://github.com/andrei-stoica26/henna

BugReports https://github.com/andrei-stoica26/henna/issues

Config/testthat/edition 3

Config/pak/sysreqs libfontconfig1-dev libfreetype6-dev libglpk-dev libicu-dev libxml2-dev

Repository https://andrei-stoica26.r-universe.dev

Date/Publication 2026-07-13 02:40:06 UTC

RemoteUrl https://github.com/andrei-stoica26/henna

RemoteRef HEAD

RemoteSha d87e7134a39f8da11e2e283d3da6be08b55f11a0

Contents

centerTitle . . . 2
classPlot . . . 3
connectedComponents . . . 4
convexHull . . . 5
correlationPlot . . . 6
densityPlot . . . 6
dpColors . . . 9
hpColors . . . 10
hullPlot . . . 10
isPointOnBoundary . . . 12
isPointOnSeg . . . 13
labelPoints . . . 14
networkPlot . . . 15
pointsToSegments . . . 16
radialPlot . . . 17
rankPlot . . . 19
rankSummary . . . 20
reorderDF . . . 21
riverPlot . . . 22
rpColors . . . 23
tilePlot . . . 23
vertexComponents . . . 25
volcanoPlot . . . 26

Index **29**

`centerTitle` *Add a centered title to a plot*

Description

This function adds a centered title to a ggplot object.

Usage

```
centerTitle(p, title, ...)
```

Arguments

`p`	A ggplot object.
`title`	Plot title.
`...`	Other arguments passed to `ggplot2::element_text`.

Value

A ggplot object.

`classPlot` *Plot bars for item counts grouped and colored by class*

Description

This function plots bars for item counts grouped and colored by class.

Usage

```
classPlot(
  df,
  title = NULL,
  xLab = "Value",
  yLab = "Item",
  legendTitle = "Class",
  palette = "Spectral",
  labelSize = 2.5,
  labelColor = "black",
  legendTitleSize = 10,
  legendTextSize = 10,
  axisTextSize = 12,
  axisTitleSize = 12,
  decreasing = TRUE,
  valueCutoff = 0,
  ...
)
```

Arguments

`df`	A data frame with at least three columns. Its first column (categorical) colors the plot bars. The second column (categorical) labels the plots bars. The third column (numeric) sets the bar lengths.
`title`	Plot title.
`xLab`	x axis label.
`yLab`	y axis label.
`legendTitle`	Legend title.
`palette`	Color palette.
`labelSize`	Label size.
`labelColor`	Label color.
`legendTitleSize`	Legend title size.
`legendTextSize`	Legend text size.
`axisTextSize`	Axis text size.
`axisTitleSize`	Axis title size.

decreasing	Whether to display the bars in decreasing order of length.
valueCutoff	Cutoff used for filtering the input data frame based on the third (value) column. Only values above this cutoff will be displayed on the plot.
...	Additional arguments passed to centerTitle.

Value

An object of class gg.

Examples

```
df <- data.frame(Class = sample(paste0('C', seq(13)), 25, replace=TRUE),
Item = paste0('I', seq(25)),
Value = runif(25, 0.5, 1))
classPlot(df)

df <- data.frame(Class = sample(paste0('C', seq(13)), 25, replace=TRUE),
Item = sample(paste0('I', seq(21)), 25, replace=TRUE),
Value = runif(25, 0.5, 1))
classPlot(df)
```

---

connectedComponents *Find the connected components of a graph represented as a data frame*

Description

This function finds the connected components of a graph represented as a data frame.

Usage

```
connectedComponents(df, colName = "component")
```

Arguments

df	A data frame with two categorical columns representing the edges of a graph.
colName	Name of the connected components column to be added.

Value

A data frame with a column indicating the connected component of each edge.

Examples

```
df <- data.frame(
gene1 = paste0('G', c(1, 2, 6, 7, 8, 9,
11, 25, 32, 17, 18)),
gene2 = paste0('G', c(2, 8, 8, 8, 1, 25,
32, 24, 24, 26, 26))
)
connectedComponents(df)
```

---

convexHull *Construct the convex hull of a set of points*

Description

This function constructs the convex hull of a set of points.

Usage

```
convexHull(pointsDF, hullIndices = NULL)
```

Arguments

pointsDF	A data frame with the x and y coordinates of the points.
hullIndices	Precalculated hull indices. Default is NULL: hull indices are not provided, but they are calculated by convexHull.

Details

The points must be provided as a data frame with two columns.

Value

A data frame with two columns representing the points on the convex hull.

Examples

```
pointsDF <- data.frame(a = c(1, 2, 2, 3, 3, 4, 5, 6, 8, 6,
7, 8, 6, 8, 10, 3, 1),
b = c(2, 3, 4, 8, 5, 6, 5, 4, 8, 11, 13, 14, 2, 1, 2, 14, 9))
hull <- convexHull(pointsDF)
```

correlationPlot *Plot a correlation matrix*

Description

This function plots a correlation matrix.

Usage

```
correlationPlot(mat, title = NULL, legendTitle = "Correlation", ...)
```

Arguments

mat	A numeric matrix or data frame.
title	Plot title.
legendTitle	Legend title.
...	Additional parameters passed to tilePlot.

Details

A thin wrapper around `tilePlot`.

Value

An object of class gg.

Examples

```
mat <- matrix(runif(100, -1, 1), nrow=10)
colnames(mat) <- paste0('I', seq(10))
mat <- round(cor(mat), 2)
correlationPlot(mat)
```

densityPlot *Create density plot*

Description

This function creates a density plot.

Usage

```
densityPlot(
  df,
  title = NULL,
 colorScheme = c("cloudy", "cake", "grapes", "lava", "oasis", "orichalc", "sea", "sky",
    "custom"),
  useSchemeDefaults = TRUE,
  drawNN = TRUE,
  drawScores = FALSE,
  xLab = NULL,
  yLab = NULL,
  legendTitle = "Density",
  palette = NULL,
  segColor = "black",
  pointSize = 1,
  pointColor = "red",
  segType = c("dashed", "solid", "dotted", "dotdash", "longdash", "twodash"),
  segWidth = 0.4,
  nGridPoints = 300,
  expandPerc = 20,
  labelType = c("free", "boxed"),
  labelSize = 3,
  labelColor = "black",
  labelRepulsion = 1,
  labelPull = 1,
  maxOverlaps = 10,
  labelPadding = 0,
  boxPadding = 0,
  labelSegWidth = 0.4,
  legendPos = c("right", "none"),
  legendTextSize = 10,
  legendTitleSize = 10,
  axisTextSize = 12,
  axisTitleSize = 12,
  verbose = FALSE,
  ...
)
```

Arguments

df	A data frame with at least two columns, representing the x and y coordinates of the points. A score column can also be provided as the third column. Nearest neighbor information can be provided in the last column as a character vector with elements selected from the rownames.
title	Plot title.
colorScheme	Color scheme. Choose between 'cake', 'cloudy', 'grapes', 'lava', oasis', 'orichalc', 'sea', 'sky' and 'custom'. Default is 'cloudy'.

useSchemeDefaults
: Whether to use the default segColor, pointColor and labelColor values for scheme. Ignored if colorScheme is set to 'custom'.

drawNN
: Whether to draw segments linking each point to its nearest neighbor.

drawScores
: Whether to render scores on the plot. If set to TRUE, the third column of the input data frame will be numeric and scores will be taken from there.

xLab
: x axis label.

yLab
: y axis label.

legendTitle
: Legend title.

palette
: A character vector of colors. Used only if color scheme is set to 'custom'.

segColor
: Nearest neighbor segment color. Ignored if drawNN is set to FALSE, or if useSchemeDefaults is TRUE and colorScheme is different from 'custom'.

pointSize
: Point size.

pointColor
: Point color. Ignored if useSchemeDefaults is TRUE and colorScheme is different from 'custom'.

segType
: Nearest neighbor segment type. Choose between 'solid', 'dashed', 'dotted','dotdash', 'longdash' and 'twodash'. Ignored if drawNN is set to FALSE.

segWidth
: Nearest neighbor segment width. Ignored if drawNN is set to FALSE.

nGridPoints
: Number of grid points in each direction.

expandPerc
: Percentage by which the grid will be expanded.

labelType
: Whether to draw a box around labels (option 'boxed') or not (option 'free'). Default is 'free'.

labelSize
: Label size.

labelColor
: Label color. Ignored if useSchemeDefaults is TRUE and colorScheme is different from 'custom'.

labelRepulsion
: Repulsion strength between labels.

labelPull
: Attraction strength between a text label and its data point.

maxOverlaps
: Maximum number of allowed overlaps.

labelPadding
: Amount of padding around label.

boxPadding
: Amount of padding around box.

labelSegWidth
: Thickness of segment connecting label to point.

legendPos
: Legend position. Choose between 'right' and 'none'.

legendTextSize
: Legend text size.

legendTitleSize
: Legend title size.

axisTextSize
: Axis text size.

axisTitleSize
: Axis title size.

verbose
: Whether output should be verbose.

...
: Additional arguments passed to centerTitle.

Value

An object of class gg.

Examples

```
x <- c(1, 2, 3, 4, 6, 7, 8, 10, 12, 11, 3, 6, 4, 1, 13, 13, 14, 18, 16)
y <- c(1, 3, 1, 4, 3, 2, 8, 2, 1, 11, 8, 8, 10, 14, 13, 11, 11, 12,15)
z <- round(runif(19, 75, 100), 2)
df <- data.frame(x, y, z)
rownames(df) <- paste0('p', rownames(df))
densityPlot(df)
```

---

dpColors *Create a palette designed for densityPlot*

Description

This function returns a palette designed for densityPlot.

Usage

```
dpColors(
 palette = c("cake", "cloudy", "grapes", "lava", "oasis", "orichalc", "sea", "sky")
)
```

Arguments

palette	One of 'lava', 'oasis', 'orichalc', 'sea' and 'sky'.

Value

A character vector of colors.

Examples

```
dpColors('sea')
```

hpColors *Create the default hullPlot palette*

Description

This function returns the default palette used by `hullPlot`.

Usage

```
hpColors()
```

Value

A character vector of colors.

hullPlot *Plot the convex hull of a set of points*

Description

This function plots the convex hull of a set of points. It can also draw a vertical or a horizontal line (or both), dividing the hull into areas of different colors.

Usage

```
hullPlot(
  pointsDF,
  title = NULL,
  xInt = NULL,
  yInt = NULL,
  palette = hpColors(),
  lineColor = "navy",
  lineWidth = 0.3,
  lineType = c("dashed", "solid", "dotted", "dotdash", "longdash", "twodash"),
  hullWidth = 0,
  xLab = NULL,
  yLab = NULL,
  legendLabs = paste0("Group ", seq(4)),
  pointSize = 1,
  pointShape = 4,
  alpha = 0.2,
  labeledPoints = NULL,
  labelOutside = FALSE,
  labXThr = NULL,
  labYThr = NULL,
```

```
  labelType = c("free", "boxed"),
  labelSize = 2.5,
  labelColor = "black",
  labelRepulsion = 1,
  labelPull = 0,
  maxOverlaps = 10,
  legendPos = "bottom",
  legendTextSize = 10,
  axisTextSize = 12,
  axisTitleSize = 12,
  ...
)
```

Arguments

pointsDF	A data frame with the x and y coordinates of the points.
title	Plot title.
xInt	The coordinate where the vertical line intersects the x axis.
yInt	The coordinate where the horizontal line intersects the y axis.
palette	Color palette.
lineColor	The color of the horizontal and vertical dividing lines, if provided. If NULL, no dividing lines will be drawn, though the hull will still be split along these lines if xInt and/or yIntare not NULL.
lineWidth	The width of the horizontal and vertical dividing lines. Ignored if lineColor is NULL.
lineType	The type of the horizontal and vertical dividing lines. Choose between 'dashed','solid', 'dotted', 'dotdash', 'longdash' and 'twodash'. Default is 'dashed'. Ignored if lineColor is NULL.
hullWidth	Width of the convex hull. If 0 (as default), the convex hull will not be displayed.
xLab	x axis label.
yLab	y axis label.
legendLabs	Legend labels.
pointSize	Point size.
pointShape	Point shape.
alpha	Opaqueness level.
labeledPoints	Point labels to be displayed on the plot.
labelOutside	Display labels for points specified by labeledPoints even if they fall outside the boundaries imposed by labXThr and labYThr. Ignored if labeledPoints is NULL or both labXThr and labYThr are NULL.
labXThr	Threshold used to plot labels based on xCol values.
labYThr	Threshold used to plot labels based on yCol values.
labelType	Whether to draw a box around labels (option 'boxed') or not (option 'free'). Default is 'free'.

labelSize	Label size. Ignored if labelDF is NULL.
labelColor	Label color. Ignored if labelDF is NULL.
labelRepulsion	Repulsion strength between labels.
labelPull	Attraction strength between a text label and its data point.
maxOverlaps	Maximum overlaps. Ignored if labelDF is NULL.
legendPos	Legend position.
legendTextSize	Legend text size.
axisTextSize	Axis text size.
axisTitleSize	Axis title size.
...	Additional arguments passed to centerTitle.

Value

An object of class gg.

Examples

```
pointsDF <- data.frame(x = c(1, 2, 4, 7, 10,
12, 13, 15, 16),
y = c(1, 1, 2, 3, 3, 2,
1, 2, 1))
hullPlot(pointsDF, 'Hull plot', 7, 1.5)
```

---

isPointOnBoundary *Check if a point is on a polygon boundary*

Description

This function checks if a point P is on a polygon boundary.

Usage

```
isPointOnBoundary(xPoint, yPoint, boundary)
```

Arguments

xPoint	x coordinate of point P.
yPoint	y coordinate of point P.
boundary	A data frame with four columns representing segments comprising the boundary.

Value

Logical; whether the point is on the boundary.

Examples

```
pointsDF <- data.frame(x = c(1, 2, 4, 7, 10,
12, 13, 15, 16),
y = c(1, 1, 2, 3, 3, 2,
1, 2, 1))

hullIndices <- grDevices::chull(pointsDF[, 1], pointsDF[, 2])
hull <- convexHull(pointsDF, hullIndices)
hullSegments <- pointsToSegments(hull)

isPointOnBoundary(2, 3, hullSegments)
```

---

isPointOnSeg *Check if a point is on a segment*

Description

This function checks if a point P is on a segment AB.

Usage

```
isPointOnSeg(xPoint, yPoint, xStart, yStart, xEnd, yEnd)
```

Arguments

xPoint	x coordinate of point P.
yPoint	y coordinate of point P.
xStart	x coordinate of point A.
yStart	y coordinate of point A.
xEnd	x coordinate of point B.
yEnd	y coordinate of point B.

Value

Logical; whether the point is on the segment.

Examples

```
isPointOnSeg(2, 3, 1, 2, 3, 4)
isPointOnSeg(2, 3, 1, 2, 3, 8)
isPointOnSeg(4, 5, 1, 2, 3, 4)
```

`labelPoints` *Label points in a ggplot object*

Description

This function labels points in a ggplot object.

Usage

```
labelPoints(
  p,
  labelDF,
  pointLabs = rownames(labelDF),
  labelType = c("free", "boxed"),
  labelSize = 2.5,
  labelColor = "black",
  labelRepulsion = 1,
  labelPull = 1,
  maxOverlaps = 50,
  boxPadding = 0.2,
  labelPadding = 0.1,
  labelSegWidth = 0.4,
  ...
)
```

Arguments

`p`	A ggplot object.
`labelDF`	Label data frame.
`pointLabs`	Labels of points.
`labelType`	Whether to draw a box around labels (option 'boxed') or not (option 'free'). Default is 'free'.
`labelSize`	Label size.
`labelColor`	Label color.
`labelRepulsion`	Repulsion strength between labels.
`labelPull`	Attraction strength between a text label and its data point.
`maxOverlaps`	Maximum number of allowed overlaps.
`boxPadding`	Amount of padding around box.
`labelPadding`	Amount of padding around label.
`labelSegWidth`	Thickness of segment connecting label to point.
`...`	Additional arguments passed to `geom_text_repel` (if `labelType` is 'free') or `geom_text_label` (if `labelType` is 'boxed').

Value

A ggplot object.

Examples

```
filePath <- system.file('extdata', 'hullPlot.qs2', package='henna')
sharedDF <- qs2::qs_read(filePath)
name1 <- 'alpha'
name2 <- 'delta'
legendLabs <- as.factor(c('Non-top',
'Shared',
paste0('Top only for ', name2),
paste0('Top only for ', name1)))
p <- hullPlot(sharedDF, 'Shared markers plot', xInt=1.5, yInt=1.3,
xLab=paste0('avg_log2FC (', name1, ')'),
yLab=paste0('avg_log2FC (', name2, ')'),
legendLabs=legendLabs)
labelDF <- sharedDF[sharedDF[, 'avg_log2FC_1'] > 1.5 &
sharedDF[, 'avg_log2FC_2'] > 1.3, ]
p <- labelPoints(p, labelDF, labelType='boxed', nudge_x=0.1, nudge_y=0.1)
```

---

networkPlot *Plot graph with the option of using different colors for connected components*

Description

This function plots the graph of the data frame and optionally uses different colors for nodes belonging to different connected components.

Usage

```
networkPlot(
  df,
  title = NULL,
  numCol = NULL,
  numColType = c("weights", "ranks"),
  nodeSize = 10,
  nodeTextSize = 2.3,
  palette = "grDevices::Spectral",
  nodeColor = NULL,
  edgeWidth = 1,
  edgeColor = "black",
  edgeScales = c(0.2, 0.6),
  ...
)
```

Arguments

df	Data frame.
title	Plot title.
numCol	Name of the numeric column used to vary edge widths. If no such column is provided, set to NULL.
numColType	The type of the numeric column used to vary edge widths. Choose between 'weights' and 'ranks' Ignored if numCol is NULL.
nodeSize	Size of graph nodes.
nodeTextSize	Size of text on graph nodes.
palette	grDevices palette used for coloring nodes. Ignored if nodeColor is not NULL.
nodeColor	Color used for nodes. Default is NULL (palette will instead be used).
edgeWidth	Width to be used for all edges before scaling if numCol is NULL; otherwise ignored.
edgeColor	Color used for edges.
edgeScales	Edge width scales. Must be a numeric vector of size 2 (minimum and maximum).
...	Additional arguments passed to centerTitle.

Value

An object of class ggraph.

Examples

```
df <- data.frame(gene1 = paste0('G', c(1, 2, 5, 6, 7, 17)),
gene2 = paste0('G', c(2, 5, 8, 11, 11, 11)),
rank = c(1, 1, 3, 3, 3, 3))
networkPlot(df, numCol='rank', numColType='ranks')
```

---

pointsToSegments *Construct a data frame of segments from a data frame of points*

Description

This function constructs a data frame of segments from a data frame of points.

Usage

```
pointsToSegments(pointsDF, joinEnds = TRUE)
```

Arguments

pointsDF	A data frame with the x and y coordinates of the points. Each point must appear only once.
joinEnds	Whether to join the last point with the first one.

Value

A data frame of segments represented using four columns (x, y, xEnd, yEnd).

Examples

```
pointsDF <- data.frame(x = c(1, 2, 4, 7, 10,
12, 13, 15, 16),
y = c(1, 1, 2, 3, 3, 2, 1, 2, 1))

hullIndices <- grDevices::chull(pointsDF[, 1], pointsDF[, 2])
hull <- convexHull(pointsDF, hullIndices)
pointsToSegments(hull)
```

---

radialPlot *Draw radial plot for a data frame with positive integer-valued points*

Description

This function draws a radial plot for a data frame, plotting positive integer-valued points over concentric circles, with points located more centrally representing higher values.

Usage

```
radialPlot(
  valuesDF,
  title = NULL,
  valueLegendTitle = "Value",
  groupLegendTitle = NULL,
  extraCircles = 0,
  palette = rpColors(length(unique(valuesDF[, 3]))),
  labelSize = 3,
  pointSize = 0.8,
  legendTitleSize = 10,
  legendTextSize = 10,
  labelRepulsion = 1,
  labelPull = 0,
  maxOverlaps = 15,
  breakDensity = 6,
  seed = 50,
  ...
)
```

Arguments

valuesDF	A data frame with names on the first column and positive integers on the second column.
title	Plot title.
valueLegendTitle	Legend title corresponding to the positive integer column.
groupLegendTitle	Legend title corresponding to the categorical column.
extraCircles	Number of circles drawn beyond those required to include the points.
palette	Color palette.
labelSize	Label size.
pointSize	Point size.
legendTitleSize	Legend title size.
legendTextSize	Legend text size.
labelRepulsion	Repulsion strength between labels.
labelPull	Attraction strength between a text label and its data point.
maxOverlaps	Maximum number of allowed overlaps.
breakDensity	Factor used in calculating the number of breaks for the values legend. Higher values of this argument add more breaks to the legend, but no breaks at a distance below 1 will be allowed.
seed	Random seed.
...	Additional arguments passed to centerTitle.

Value

An object of class gg.

Examples

```
valuesDF <- data.frame(Protein = paste0('P', seq(20)),
Value = sample(10, 20, replace=TRUE),
Group = sample(3, 20, replace=TRUE))
radialPlot(valuesDF, groupLegendTitle='Group')
```

rankPlot *Create a rank plot*

Description

This function creates a rank plot.

Usage

```
rankPlot(
  df,
  title = NULL,
  summarize = TRUE,
  viridisPal = "turbo",
  xLab = "Item",
  yLab = "Rank count",
  legendTitle = "Rank",
  sigDigits = NULL,
  labelSize = 2.5,
  labelColor = "black",
  labelFace = c("plain", "bold", "italic", "bold-italic"),
  legendTextSize = 10,
  legendTitleSize = 10,
  axisTextSize = 12,
  axisTitleSize = 12,
  xAngle = 45,
  vJust = 1,
  hJust = 1,
  labelScalingFactor = 0.9,
  labelOffset = 0.05,
  ...
)
```

Arguments

df	A data frame with ranks as columns and items as rows, or a summary data frame generated with rankSummary. If the latter, summarize must be set to FALSE.
title	Plot title.
summarize	Whether to summarize the ranks with rankSummary. Must be set to FALSE if the input data frame has been generated with rankSummary.
viridisPal	Viridis palette.
xLab	Label of x axis.
yLab	y axis label.
legendTitle	Legend title.

sigDigits	Number of significant digits used when displaying mean ranks. If NULL, the mean ranks will not be displayed.
labelSize	Size of label marking average rank for each item. Ignored if sigDigits is NULL.
labelColor	Color of label marking average rank for each item. Ignored if sigDigits is NULL.
labelFace	Font face of label marking average rank for each item. Must be one among 'plain', 'bold', 'italic' and 'bold-italic'. Ignored if sigDigits is NULL.
legendTextSize	Legend text size.
legendTitleSize	Legend title size.
axisTextSize	Axis text size.
axisTitleSize	Axis title size.
xAngle	Angle of x axis text.
vJust	Vertical justification in [0, 1].
hJust	Horizontal justification in [0, 1].
labelScalingFactor	Scaling factor used when displaying mean ranks. Ignored if sigDigits is NULL.
labelOffset	Vertical offset used when displaying mean ranks. Ignored if sigDigits is NULL.
...	Additional arguments passed to centerTitle.

Value

An object of class gg.

Examples

```
df <- do.call(cbind, lapply(seq(30), function(i) sample(10, 10)))
rownames(df) <- paste0('M', seq(10))
colnames(df) <- paste0('R', seq(30))
rankPlot(df)
```

---

rankSummary *Create a rank summary*

Description

This function creates a summary of multiple ranks provided for input items.

Usage

```
rankSummary(df)
```

Arguments

df	A data frame with ranks as columns and items as rows.

Value

A rank summary data frame with three columns: 'Rank', 'Item' and 'Count'.

Examples

```
df <- do.call(cbind, lapply(seq(30), function(i) sample(10, 10)))
rownames(df) <- paste0('M', seq(10))
colnames(df) <- paste0('R', seq(30))
rankSummary(df)
```

---

reorderDF *Sort a data frame by the first column and convert the second to a factor*

Description

This function sort a data frame by the first column and convert the second to a factor.

Usage

```
reorderDF(df)
```

Arguments

df	A data frame.

Value

An object of class gg.

Examples

```
df <- data.frame(a = c(2, 4, 1, 3, 6),
b = c(2, 8, 3, 19, 3))
reorderDF(df)
```

`riverPlot` *Create an alluvial plot*

Description

This function creates an alluvial plot.

Usage

```
riverPlot(
  df,
  title = NULL,
  fillColIndex = 2,
  curveType = "sigmoid",
  alpha = 0.8,
  strataFill = "lightgoldenrod1",
  labelSize = 3,
  axisTextSize = 12,
  axisTitleSize = 12,
  viridisPal = "turbo",
  ...
)
```

Arguments

`df`	A data frame with two categorical columns and a numeric column.
`title`	Plot title.
`fillColIndex`	Index of column used for coloring the alluvia.
`curveType`	Curve type.
`alpha`	Opaqueness level for the colors of the alluvia.
`strataFill`	Color used for the strata.
`labelSize`	Size of labels of strata elements.
`axisTextSize`	Axis text size.
`axisTitleSize`	Axis title size.
`viridisPal`	Viridis palette.
`...`	Other arguments passed to `centerTitle`.

Value

An object of class gg.

Examples

```
df <- data.frame(x = sample(c('a','b', 'c', 'd', 'e', 'f'), 20,
replace=TRUE),
y = sample(c('p','q', 'r', 's', 't', 'u', 'v', 'w'), 20,
replace=TRUE),
z = runif(20, 1, 3))
riverPlot(df)
```

---

rpColors *Create a palette designed to represent dots over a viridis background*

Description

This function returns a 10-color palette used as the default of `radialPlot`.

Usage

```
rpColors(nColors = 10)
```

Arguments

`nColors`	Number of colors.

Value

A character vector of colors.

---

tilePlot *Plot a numeric matrix or data frame*

Description

This function plots a numeric matrix or data frame.

Usage

```
tilePlot(
  mat,
  title = NULL,
  xLab = NULL,
  yLab = NULL,
  doMelt = TRUE,
  legendTitle = "Value",
  palette = "Spectral",
  reverseColors = TRUE,
```

```
  showNumbers = TRUE,
  sigDigits = 2,
  limits = NULL,
  labelSize = 3,
  labelColor = "black",
  legendTextSize = 10,
  legendTitleSize = 10,
  axisTextSize = 12,
  axisTitleSize = 12,
  tileBoundaryColor = "white",
  tileBoundaryWidth = 0.2,
  xAngle = 45,
  vJust = 1,
  hJust = 1,
  ...
)
```

Arguments

`mat`	A numeric matrix or data frame.
`title`	Plot title.
`xLab`	x axis label.
`yLab`	y axis label.
`doMelt`	Whether the input needs to be melted using `reshape2::melt`.
`legendTitle`	Legend title.
`palette`	Color palette.
`reverseColors`	Whether to reverse the order of colors in the palette.
`showNumbers`	Whether to show a numeric value for each matrix element.
`sigDigits`	Number of significant digits to be displayed for each matrix element.
`limits`	Limits of the color scale. If `NULL`, they will be taken as the minimum and maximum of the values in the input object.
`labelSize`	Label size.
`labelColor`	Label color.
`legendTextSize`	Legend text size.
`legendTitleSize`	Legend title size.
`axisTextSize`	Axis text size.
`axisTitleSize`	Axis title size.
`tileBoundaryColor`	Tile boundary color.
`tileBoundaryWidth`	Tile boundary width.
`xAngle`	Angle of x axis text.

vJust	Vertical justification in [0, 1].
hJust	Horizontal justification in [0, 1].
...	Additional arguments passed to centerTitle.

Value

An object of class gg.

Examples

```
mat <- matrix(round(runif(100, 0, 1), 2), nrow=10)
rownames(mat) <- paste0('R', seq(10))
colnames(mat) <- paste0('C', seq(10))
tilePlot(mat)
```

---

vertexComponents *Return the connected components of vertices*

Description

This function returns the connected components of vertices from a graph data frame in which edges have been assigned connected components.

Usage

```
vertexComponents(df, colName = "component")
```

Arguments

df	A data frame with two categorical columns representing graph edges and a connected components column.
colName	Name of the connected components column.

Value

A factor vector representing the connected component of each vertex.

Examples

```
df <- data.frame(gene1 = c('A', 'B', 'C', 'A'),
gene2 = c('B', 'D', 'F', 'G'),
component = c(1, 1, 2, 1))
vertexComponents(df)
```

volcanoPlot *Create a volcano plot*

Description

This function creates a volcano plot for a data frame with a log column and a p-value column. The gene names must be provided as row names.

Usage

```
volcanoPlot(
  df,
  title = NULL,
  logCol = "avg_log2FC",
  pvalCol = "p_val_adj",
  xLab = expression(log[2] ~ fold ~ change),
  yLab = expression(-log[10] ~ `p-value`),
  legendTitle = "Significance",
  legendLabs = c("Not significant", expression(log[2] ~ FC), "p-value",
    expression(`p-value` ~ and ~ log[2] ~ FC)),
  legendPos = c("right", "top", "left", "bottom"),
  logFCThr = 1,
  pvalThr = 1e-05,
  labeledGenes = NULL,
  labelOutside = FALSE,
  labLogFCThr = 1.8,
  labPvalThr = 1e-12,
  labelType = c("boxed", "free"),
  labelSize = 2.2,
  labelColor = "black",
  labelRepulsion = 1,
  labelPull = 0,
  maxOverlaps = 100,
  boxPadding = 0.2,
  labelPadding = 0.1,
  labelSegWidth = 0.4,
  pointSize = 0.8,
  alpha = 0.6,
  palette = c("gray31", "goldenrod2", "orchid3", "red2"),
  legendTextSize = 10,
  legendTitleSize = 10,
  axisTextSize = 12,
  axisTitleSize = 12,
  theme = c("minimal", "bw", "classic", "linedraw"),
  ...
)
```

Arguments

df	A data frame with rownames as genes, a log column and a p-value column.
title	Plot title.
logCol	Log column.
pvalCol	P-value column.
xLab	x axis label.
yLab	y axis label.
legendTitle	Legend title.
legendLabs	Legend labels.
legendPos	Legend position.
logFCThr	Threshold used to separate significant log values.
pvalThr	Threshold used to separate significant p-values.
labeledGenes	Gene labels to be displayed on the plot. Default is NULL, entailing that gene labels will be displayed on the basis of labLogFCThr and labPvalThr, if at least one of them is not NULL.
labelOutside	Display labels for points specified by labeledGenes even if they fall outside the boundaries imposed by labLogFCThr and labPvalThr. Ignored if labeledGenes is NULL or both labLogFCThr and labPvalThr are NULL.
labLogFCThr	Threshold used to plot gene labels based on log values. Ignored if labeledGenes is not NULL.
labPvalThr	Threshold used to plot gene labels based on p-values. Ignored if labeledGenes is not NULL.
labelType	Whether to draw a box around labels (option 'boxed') or not (option 'free'). Default is 'free'.
labelSize	Label size.
labelColor	Label color.
labelRepulsion	Repulsion strength between labels.
labelPull	Attraction strength between a text label and its data point.
maxOverlaps	Maximum number of allowed overlaps.
boxPadding	Amount of padding around box.
labelPadding	Amount of padding around label.
labelSegWidth	Thickness of segment connecting label to point.
pointSize	Point size.
alpha	Opaqueness level of point color.
palette	Color palette.
legendTextSize	Legend text size.
legendTitleSize	Legend title size.
axisTextSize	Axis text size.
axisTitleSize	Axis title size.
theme	Plot theme. Choose between 'bw', 'classic', 'linedraw' and 'minimal'. Default is 'minimal'.
...	Additional arguments passed to EnhancedVolcano::EnhancedVolcano.

Details

Users can input labeled genes in two ways:

1) By using p-value and log fold-change thresholds (the default option).

2) By inputting a list of labels.

Value

An object of class gg.

Examples

```
if (requireNamespace("EnhancedVolcano", quietly=TRUE)){
filePath <- system.file('extdata', 'volcanoPlot.qs2', package='henna')
df <- qs2::qs_read(filePath)
p <- volcanoPlot(df, title='Volcano plot - beta cells', pvalThr=1e-10,
logFCThr=1,
labPvalThr=1e-150,
labLogFCThr=5.3)
}
```

Index

centerTitle, 2
classPlot, 3
connectedComponents, 4
convexHull, 5
correlationPlot, 6

densityPlot, 6
dpColors, 9

hpColors, 10
hullPlot, 10

isPointOnBoundary, 12
isPointOnSeg, 13

labelPoints, 14

networkPlot, 15

pointsToSegments, 16

radialPlot, 17
rankPlot, 19
rankSummary, 20
reorderDF, 21
riverPlot, 22
rpColors, 23

tilePlot, 23

vertexComponents, 25
volcanoPlot, 26