

Package ‘henna’

October 19, 2025

Type Package

Title A Versatile Visualization Suite

Version 0.3.4

Description A visualization suite primarily designed for single-cell RNA-sequencing data analysis applications, but adaptable to 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 and alluvial plots) for ease of usage and flexibility.

License MIT + file LICENSE

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

Encoding UTF-8

RoxygenNote 7.3.3

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation no

Author Andrei-Florian Stoica [aut, cre] (ORCID:
<<https://orcid.org/0000-0002-5253-0826>>)

Maintainer Andrei-Florian Stoica <andreistoica@foxmail.com>

Repository CRAN

Date/Publication 2025-10-19 16:00:02 UTC

Contents

centerTitle	2
classPlot	3
connectedComponents	4
convexHull	5
correlationPlot	5

densityPlot	6
dpColors	8
hpColors	8
hullPlot	9
isPointOnBoundary	10
isPointOnSeg	11
networkPlot	12
pointsToSegments	13
radialPlot	14
rankPlot	15
rankSummary	17
riverPlot	17
rpColors	18
tilePlot	19
vertexComponents	20
Index	22

centerTitle	<i>Add a centered title to a plot</i>
-------------	---------------------------------------

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	<i>Plot item bars grouped by class</i>
-----------	--

Description

This function plots bars for each item while grouping them by class and ordering them.

Usage

```
classPlot(  
  df,  
  title = "Class plot",  
  xLab = "Value",  
  yLab = "Item",  
  legendLab = "Class",  
  palette = "Spectral",  
  labelSize = 2.5,  
  labelColor = "black",  
  decreasing = TRUE,  
  valueCutoff = 0,  
  ...  
)
```

Arguments

df	A data frame with at least three columns, with class, item and value as the first three columns. The latter must be numeric.
title	Plot title.
xLab	Label of x axis.
yLab	Label of y axis.
legendLab	Legend label.
palette	grDevices palette used for coloring nodes. Ignored if nodeColor is not NULL.
labelSize	Label size. Ignored if labelDF is NULL.
labelColor	Label color.
decreasing	Whether to display the bars in decreasing order of length.
valueCutoff	Cutoff used for filtering the input data frame based on the value column. Only values greater than this cutoff will be displayed on the plot.
...	Other 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	<i>Find the connected components of a graph data frame</i>
---------------------	--

Description

This function finds the connected components of a graph 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	<i>Construct the convex hull of a set of points</i>
------------	---

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	<i>Plot a correlation matrix</i>
-----------------	----------------------------------

Description

This function plots a correlation matrix.

Usage

```
correlationPlot(  
  mat,  
  title = "Correlation plot",  
  legendLab = "Correlation",  
  ...  
)
```

Arguments

<code>mat</code>	A numeric matrix or data frame.
<code>title</code>	Plot title.
<code>legendLab</code>	Legend label.
<code>...</code>	Additional parameters passed to <code>tilePlot</code> .

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 = "Density plot",
  colorScheme = c("oasis", "sea", "lava", "custom"),
  useSchemeDefaults = FALSE,
  drawNN = TRUE,
  drawScores = FALSE,
  palette = NULL,
  segColor = "plum1",
  pointSize = 0.8,
  pointColor = "red",
  segType = c("dashed", "solid", "dotted", "dotdash", "longdash", "twodash"),
  segWidth = 0.4,
  legendPos = c("right", "none"),
  nGridPoints = 300,
```

```

    expandPerc = 20,
    labelSize = 2.5,
    labelColor = "black",
    labelRepulsion = 1,
    labelPull = 1,
    maxOverlaps = Inf,
    verbose = FALSE,
    ...
)

```

Arguments

<code>df</code>	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.
<code>title</code>	Plot title.
<code>colorScheme</code>	Color scheme.
<code>useSchemeDefaults</code>	Whether to use the default <code>segColor</code> , <code>pointColor</code> and <code>labelColor</code> for scheme. Ignored if <code>colorScheme</code> is set to 'custom'.
<code>drawNN</code>	Whether to draw segments linking each point to its nearest neighbor.
<code>drawScores</code>	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.
<code>palette</code>	Color palette. Used only if color scheme is set to 'custom'.
<code>segColor</code>	Nearest neighbor segment color. Ignored if <code>drawNN</code> is set to FALSE, or if <code>useSchemeDefaults</code> is TRUE and <code>colorScheme</code> is different from 'custom'.
<code>pointSize</code>	Point size.
<code>pointColor</code>	Point color. Ignored if <code>useSchemeDefaults</code> is TRUE and <code>colorScheme</code> is different from 'custom'.
<code>segType</code>	Nearest neighbor segment type. Must choose between 'solid', 'dashed', 'dotted', 'dotdash', 'longdash' and 'twodash'. Ignored if <code>drawNN</code> is set to FALSE.
<code>segWidth</code>	Nearest neighbor segment width. Ignored if <code>drawNN</code> is set to FALSE.
<code>legendPos</code>	Legend position. Choose between 'right' and 'none'.
<code>nGridPoints</code>	Number of grid points in each direction.
<code>expandPerc</code>	Percentage by which the grid will be expanded.
<code>labelSize</code>	Size of labels of strata elements.
<code>labelColor</code>	Label color. Ignored if <code>useSchemeDefaults</code> is TRUE and <code>colorScheme</code> is different from 'custom'.
<code>labelRepulsion</code>	Repulsion strength between labels.
<code>labelPull</code>	Attraction strength between a text label and its data point.
<code>maxOverlaps</code>	Maximum overlaps.
<code>verbose</code>	Whether output should be verbose.
<code>...</code>	Other arguments passed to <code>centerTitle</code> .

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("oasis", "sea", "lava"))
```

Arguments

palette One of 'sea' and 'lava'.

Value

A character vector of colors.

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	<i>Plot the convex hull of a set of points</i>
----------	--

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 = "Hull plot",
  xInt = NULL,
  yInt = NULL,
  palette = hpColors(),
  lineColor = "navy",
  lineWidth = 0.3,
  lineType = c("dashed", "solid", "dotted", "dotdash", "longdash", "twodash"),
  hullWidth = 0,
  xLab = "x",
  yLab = "y",
  legendLabs = paste0("Group ", seq(4)),
  legendPos = "bottom",
  pointSize = 1,
  pointShape = 4,
  alpha = 0.2,
  labelDF = NULL,
  labelSize = 2.5,
  labelColor = "black",
  labelRepulsion = 1,
  labelPull = 0,
  maxOverlaps = 10,
  ...
)
```

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 yInt are 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	Label of x axis.
yLab	Label of y axis.
legendLabs	Legend labels.
legendPos	Legend position.
pointSize	Point size.
pointShape	Point shape.
alpha	Opaqueness level for the colors of the alluvia.
labelDF	Label data frame.
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.
...	Other 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	<i>Check if a point is on a polygon boundary</i>
-------------------	--

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	<i>Check if a point is on a segment</i>
--------------	---

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)
```

networkPlot	<i>Plot graph with the option of using different colors for connected components</i>
-------------	--

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 = "Network plot",
  rankCol = "rank",
  edgeScale = 2,
  nodeSize = 10,
  nodeTextSize = 2.3,
  palette = "Spectral",
  nodeColor = NULL,
  edgeColor = "black",
  ...
)
```

Arguments

df	Data frame.
title	Plot title.
rankCol	Name of the rank column.
edgeScale	Scaling factor used in generating edge weights.
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).
edgeColor	Color used for edges.
...	Other 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)
```

pointsToSegments	<i>Construct a data frame of segments from a data frame of points</i>
------------------	---

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	<i>Draw radial plot for a data frame with positive integer-valued points</i>
------------	--

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 = "Radial plot",
  valueLegendTitle = "Value",
  groupLegendTitle = "Group",
  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 overlaps. Ignored if labelDF is NULL.
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.
...	Other 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)
```

rankPlot

*Create a rank plot***Description**

This function creates a rank plot.

Usage

```
rankPlot(
  df,
  title = "Rank plot",
  summarize = TRUE,
  viridisPal = "turbo",
  xLab = "Item",
  yLab = "Rank count",
  legendLab = "Rank",
  sigDigits = NULL,
  labelSize = 2.5,
  labelColor = "black",
  labelFace = c("plain", "bold", "italic", "bold-italic"),
  xAngle = 45,
  vJust = 0.6,
  labelScalingFactor = 0.9,
  labelOffset = 0.05,
  ...
)
```

Arguments

<code>df</code>	A data frame with ranks as columns and items as rows, or a summary data frame generated with <code>rankSummary</code> . If the latter, <code>summarize</code> must be set to <code>FALSE</code> .
<code>title</code>	Plot title.
<code>summarize</code>	Whether to summarize the ranks with <code>rankSummary</code> . Must be set to <code>FALSE</code> if the input data frame has been generated with <code>rankSummary</code> .
<code>viridisPal</code>	Viridis palette.
<code>xLab</code>	Label of x axis.
<code>yLab</code>	Label of y axis.
<code>legendLab</code>	Legend label.
<code>sigDigits</code>	Number of significant digits used when displaying mean ranks. If <code>NULL</code> , the mean ranks will not be displayed.
<code>labelSize</code>	Size of label marking average rank for each item. Ignored if <code>sigDigits</code> is <code>NULL</code> .
<code>labelColor</code>	Color of label marking average rank for each item. Ignored if <code>sigDigits</code> is <code>NULL</code> .
<code>labelFace</code>	Font face of label marking average rank for each item. Must be one among 'plain', 'bold', 'italic' and 'bold-italic'. Ignored if <code>sigDigits</code> is <code>NULL</code> .
<code>xAngle</code>	Angle of x axis text.
<code>vJust</code>	Vertical justification in <code>[0, 1]</code> .
<code>labelScalingFactor</code>	Scaling factor used when displaying mean ranks. Ignored if <code>sigDigits</code> is <code>NULL</code> .
<code>labelOffset</code>	Vertical offset used when displaying mean ranks. Ignored if <code>sigDigits</code> is <code>NULL</code> .
<code>...</code>	Other arguments passed to <code>centerTitle</code> .

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)
```

riverPlot	Create an alluvial plot
-----------	-------------------------

Description

This function creates an alluvial plot.

Usage

```
riverPlot(
  df,
  title = "River plot",
  fillColIndex = 2,
  curveType = "sigmoid",
  alpha = 0.8,
  strataFill = "lightgoldenrod1",
  labelSize = 3,
  viridisPal = "turbo",
  legendPos = "none",
```

```

    margins = margin(0, -10, -10, -10),
    ...
)

```

Arguments

<code>df</code>	A data frame with two categorical columns and a numeric column.
<code>title</code>	Plot title.
<code>fillColIndex</code>	Index of column used for coloring the alluvia.
<code>curveType</code>	Curve type.
<code>alpha</code>	Opaqueness level for the colors of the alluvia.
<code>strataFill</code>	Color used for the strata.
<code>labelSize</code>	Size of labels of strata elements.
<code>viridisPal</code>	Viridis palette.
<code>legendPos</code>	Legend position.
<code>margins</code>	Plot margins. Must be a vector of size 4 listing the desired top, right, bottom and left margin, in that order.
<code>...</code>	Other arguments passed to <code>centerTitle</code> .

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	<i>Plot a numeric matrix or data frame</i>
----------	--

Description

This function plots a numeric matrix or data frame.

Usage

```
tilePlot(
  mat,
  title = "Tile plot",
  xLab = "x",
  yLab = "y",
  legendLab = "Value",
  sigDigits = 2,
  isCor = FALSE,
  labelSize = 3,
  labelColor = "black",
  tileBoundaryColor = "white",
  tileBoundaryWidth = 0.2,
  palette = "Spectral",
  reverseColors = TRUE,
  xAngle = 45,
  vJust = 0.6,
  ...
)
```

Arguments

mat	A numeric matrix or data frame.
title	Plot title.
xLab	Label of x axis.
yLab	Label of y axis.
legendLab	Legend label.
sigDigits	Number of significant digits to be displayed for each matrix element.
isCor	Whether the matrix is a correlation matrix, in which case the limits of the color scale will be set to [-1, 1].

labelSize	Label size. Ignored if labelDF is NULL.
labelColor	Label color.
tileBoundaryColor	Tile boundary color.
tileBoundaryWidth	Tile boundary width.
palette	grDevices palette used for coloring nodes. Ignored if nodeColor is not NULL.
reverseColors	Whether to reverse the order of colors in the palette.
xAngle	Angle of x axis text.
vJust	Vertical justification in [0, 1].
...	Other 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	<i>Return the connected components of vertices</i>
------------------	--

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)
```

Index

centerTitle, [2](#)
classPlot, [3](#)
connectedComponents, [4](#)
convexHull, [5](#)
correlationPlot, [5](#)

densityPlot, [6](#)
dpColors, [8](#)

hpColors, [8](#)
hullPlot, [9](#)

isPointOnBoundary, [10](#)
isPointOnSeg, [11](#)

networkPlot, [12](#)

pointsToSegments, [13](#)

radialPlot, [14](#)
rankPlot, [15](#)
rankSummary, [17](#)
riverPlot, [17](#)
rpColors, [18](#)

tilePlot, [19](#)

vertexComponents, [20](#)