Package ‘SCpubr’

January 20, 2025

Type Package

Title Generate Publication Ready Visualizations of Single Cell Transcriptomics Data

Version 2.0.2

Description A system that provides a streamlined way of generating publication ready plots for known Single-Cell transcriptomics data in a “publication ready” format. This is, the goal is to automatically generate plots with the highest quality possible, that can be used right away or with minimal modifications for a research article.

License GPL-3

URL `https://github.com/enblacar/SCpubr/`, `https://enblacar.github.io/SCpubr-book/`

BugReports `https://github.com/enblacar/SCpubr/issues/`

Depends R (>= 4.0.0)

Suggests AnnotationDbi, assertthat, AUCell, circlize, cli, cluster, clusterProfiler, colorspace, ComplexHeatmap, covr, decoupleR, dplyr (>= 1.1.0), enrichplot, forcats, ggalluvial, ggbeeswarm, ggdist, ggExtra, ggh4x, ggnewscale, ggplot2 (>= 3.4.0), ggplotify, ggrastr, ggrepel, ggridges, ggsignif, graphics, infercnv, knitr, labeling, magrittr, MASS, Matrix, methods, Nebulosa, org.Hs.eg.db, patchwork, pheatmap, plyr, purrr, qpdf, RColorBrewer, rjags, rlang, rmarkdown, scales, scattermore, Seurat, SeuratObject, sf, stringr, svglite, testthat (>= 3.0.0), tibble, tidyr, UCell, viridis, withr

VignetteBuilder knitr

biocViews Software, SingleCell, Visualization

Config/testthat/edition 3

Encoding UTF-8

LazyData true

LazyDataCompression xz

RoxygenNote 7.2.1

NeedsCompilation no

Author Enrique Blanco-Carmona [cre, aut]
(<`https://orcid.org/0000-0002-1208-1691`>)

Maintainer Enrique Blanco-Carmona <`scpubr@gmail.com`>

Repository CRAN

Date/Publication 2023-10-11 09:50:02 UTC

Contents

`do_AlluvialPlot` *Generate Alluvial plots.*

Description

This function is based on the **ggalluvial** package. It allows you to generate alluvial plots from a given Seurat object.

Usage

```
do_AlluvialPlot(
  sample,
  first_group,
  last_group,
  middle_groups = NULL,
  colors.use = NULL,
  plot.title = NULL,
  plot.subtitle = NULL,
  plot.caption = NULL,
  font.size = 14,
  font.type = "sans",
  xlab = NULL,
  ylab = "Number of cells",
  repel = FALSE,
  fill.by = last_group,
  use_labels = FALSE,
  stratum.color = "black",
  stratum.fill = "white",
  stratum.width = 1/3,
  stratum.fill.conditional = FALSE,
  use_geom_flow = FALSE,
  alluvium.color = "white",
  flow.color = "white",
  flip = FALSE,
  label.color = "black",
  curve_type = "sigmoid",
  use_viridis = FALSE,
  viridis.palette = "G",
  viridis.direction = -1,
  sequential.palette = "YlGnBu",
  sequential.direction = 1,
  plot.grid = FALSE,
  grid.color = "grey75",
  grid.type = "dashed",
  na.value = "white",
  legend.position = "right",
  legend.title = NULL,
```

```
  plot.title.face = "bold",
  plot.subtitle.face = "plain",
  plot.caption.face = "italic",
  axis.title.face = "bold",
  axis.text.face = "plain",
  legend.title.face = "bold",
  legend.text.face = "plain"
)
```

Arguments

`sample`	`Seurat` | A Seurat object, generated by CreateSeuratObject.
`first_group`	`character` | Categorical metadata variable. First group of nodes of the alluvial plot.
`last_group`	`character` | Categorical metadata variable. Last group of nodes of the alluvial plot.
`middle_groups`	`character` | Categorical metadata variable. Vector of groups of nodes of the alluvial plot.
`colors.use`	`character` | Named list of colors corresponding to the unique values in fill.by (which defaults to last_group).
`plot.title`, `plot.subtitle`, `plot.caption`	`character` | Title, subtitle or caption to use in the plot.
`font.size`	`numeric` | Overall font size of the plot. All plot elements will have a size relationship with this font size.
`font.type`	`character` | Base font family for the plot. One of: • `mono`: Mono spaced font. • `serif`: Serif font family. • `sans`: Default font family.
`xlab`, `ylab`	`character` | Titles for the X and Y axis.
`repel`	`logical` | Whether to repel the text labels.
`fill.by`	`character` | One of first_group, middle_groups (one of the values, if multiple mid_groups) or last_group. These values will be used to color the alluvium/flow.
`use_labels`	`logical` | Whether to use labels instead of text for the stratum.
`stratum.color`, `alluvium.color`, `flow.color`	`character` | Color for the border of the alluvium (and flow) and stratum.
`stratum.fill`	`character` | Color to fill the stratum.
`stratum.width`	`logical` | Width of the stratum.
`stratum.fill.conditional`	`logical` | Whether to fill the stratum with the same colors as the alluvium/flow.
`use_geom_flow`	`logical` | Whether to use `geom_flow` instead of `geom_alluvium`. Visual results might differ.
`flip`	`logical` | Whether to invert the axis of the displayed plot.
`label.color`	`character` | Color for the text labels.

`curve_type`	`character` | Type of curve used in `geom_alluvium`. One of: • `linear`. • `cubic`. • `quintic`. • `sine`. • `arctangent`. • `sigmoid`. • `xspline`.
`use_viridis`	`logical` | Whether to use viridis color scales.
`viridis.palette`	`character` | A capital letter from A to H or the scale name as in scale_fill_viridis.
`viridis.direction`	`numeric` | Either 1 or -1. Controls how the gradient of viridis scale is formed.
`sequential.palette`	`character` | Type of sequential color palette to use. Out of the sequential palettes defined in `brewer.pal`.
`sequential.direction`	`numeric` | Direction of the sequential color scale. Either 1 or -1.
`plot.grid`	`logical` | Whether to plot grid lines.
`grid.color`	`character` | Color of the grid in the plot. In heatmaps, color of the border of the cells.
`grid.type`	`character` | One of the possible linetype options: • `blank`. • `solid`. • `dashed`. • `dotted`. • `dotdash`. • `longdash`. • `twodash`.
`na.value`	`character` | Color value for NA.
`legend.position`	`character` | Position of the legend in the plot. One of: • `top`: Top of the figure. • `bottom`: Bottom of the figure. • `left`: Left of the figure. • `right`: Right of the figure. • `none`: No legend is displayed.
`legend.title`	`character` | Title for the legend.
`plot.title.face`, `plot.subtitle.face`, `plot.caption.face`, `axis.title.face`, `axis.text.face`, `legend.title.face`, `legend.text.face`	`character` | Controls the style of the font for the corresponding theme element. One of:

- `plain`: For normal text.
- `italic`: For text in itallic.
- `bold`: For text in bold.
- `bold.italic`: For text both in itallic and bold.

Value

A ggplot2 object.

Examples

```
# Check Suggests.
value <- SCpubr:::check_suggests(function_name = "do_AlluvialPlot", passive = TRUE)
message(value)
if (isTRUE(value)){
  # Consult the full documentation in https://enblacar.github.io/SCpubr-book/

  # Define your Seurat object.
 sample <- readRDS(system.file("extdata/seurat_dataset_example.rds", package = "SCpubr"))

  # Compute basic sankey plot.
  p <- SCpubr::do_AlluvialPlot(sample = sample,
                               first_group = "orig.ident",
                               last_group = "seurat_clusters")

} else if (base::isFALSE(value)){
  message("This function can not be used without its suggested packages.")
  message("Check out which ones are needed using `SCpubr::state_dependencies()`.")
}
```

`do_BarPlot` *Create Bar Plots.*

Description

Create Bar Plots.

Usage

```
do_BarPlot(
  sample,
  group.by,
  order = FALSE,
  add.n = FALSE,
  add.n.face = "bold",
  add.n.expand = c(0, 1.15),
  add.n.size = 4,
  order.by = NULL,
```

```
  split.by = NULL,
  facet.by = NULL,
  position = "stack",
  font.size = 14,
  font.type = "sans",
  legend.position = "bottom",
  legend.title = NULL,
  legend.ncol = NULL,
  legend.nrow = NULL,
  legend.byrow = FALSE,
  axis.text.x.angle = 45,
  xlab = NULL,
  ylab = NULL,
  colors.use = NULL,
  flip = FALSE,
  plot.title = NULL,
  plot.subtitle = NULL,
  plot.caption = NULL,
  plot.grid = FALSE,
  grid.color = "grey75",
  grid.type = "dashed",
  plot.title.face = "bold",
  plot.subtitle.face = "plain",
  plot.caption.face = "italic",
  axis.title.face = "bold",
  axis.text.face = "plain",
  legend.title.face = "bold",
  legend.text.face = "plain",
  strip.text.face = "bold",
  return_data = FALSE
)
```

Arguments

`sample`	`Seurat` | A Seurat object, generated by CreateSeuratObject.
`group.by`	`character` | Metadata column to compute the counts of. Has to be either a character or factor column.
`order`	`logical` | Whether to order the results in descending order of counts.
`add.n`	`logical` | Whether to add the total counts on top of each bar.
`add.n.face`	`character` | Font face of the labels added by `add.n`.
`add.n.expand`	`numeric` | Vector of two numerics representing the start and end of the scale. Minimum should be 0 and max should be above 1. This basically expands the Y axis so that the labels fit when `flip` = `TRUE`. • `stack`: Set the bars side by side, displaying the total number of counts. Uses position_stack. • `fill`: Set the bars on top of each other, displaying the proportion of counts from the total that each group represents. Uses position_fill.

`add.n.size`	`numeric` | Size of the labels
`order.by`	`character` | When `split.by` is used, value of `group.by` to reorder the columns based on its value.
`split.by`	`character` | Metadata column to split the values of group.by by. If not used, defaults to the active idents.
`facet.by`	`character` | Metadata column to gather the columns by. This is useful if you have other overarching metadata.
`position`	`character` | Position function from **ggplot2**. Either stack or fill.
`font.size`	`numeric` | Overall font size of the plot. All plot elements will have a size relationship with this font size.
`font.type`	`character` | Base font family for the plot. One of: • `mono`: Mono spaced font. • `serif`: Serif font family. • `sans`: Default font family.
`legend.position`	`character` | Position of the legend in the plot. One of: • `top`: Top of the figure. • `bottom`: Bottom of the figure. • `left`: Left of the figure. • `right`: Right of the figure. • `none`: No legend is displayed.
`legend.title`	`character` | Title for the legend.
`legend.ncol`	`numeric` | Number of columns in the legend.
`legend.nrow`	`numeric` | Number of rows in the legend.
`legend.byrow`	`logical` | Whether the legend is filled by row or not.
`axis.text.x.angle`	`numeric` | Degree to rotate the X labels. One of: 0, 45, 90.
`xlab`, `ylab`	`character` | Titles for the X and Y axis.
`colors.use`	`named_vector` | Named vector of valid color representations (either name of HEX codes) with as many named colors as unique values of group.by. If group.by is not provided, defaults to the unique values of Idents. If not provided, a color scale will be set by default.
`flip`	`logical` | Whether to invert the axis of the displayed plot.
`plot.title`, `plot.subtitle`, `plot.caption`	`character` | Title, subtitle or caption to use in the plot.
`plot.grid`	`logical` | Whether to plot grid lines.
`grid.color`	`character` | Color of the grid in the plot. In heatmaps, color of the border of the cells.
`grid.type`	`character` | One of the possible linetype options: • `blank`. • `solid`.

- `dashed`.
- `dotted`.
- `dotdash`.
- `longdash`.
- `twodash`.

`plot.title.face`, `plot.subtitle.face`, `plot.caption.face`, `axis.title.face`, `axis.text.face`, `legend.title.face`, `legend.text.face`

`character` | Controls the style of the font for the corresponding theme element. One of:

- `plain`: For normal text.
- `italic`: For text in itallic.
- `bold`: For text in bold.
- `bold.italic`: For text both in itallic and bold.

`strip.text.face`

`character` | Controls the style of the font for the strip text. One of:

- `plain`: For normal text.
- `italic`: For text in itallic.
- `bold`: For text in bold.
- `bold.italic`: For text both in itallic and bold.

`return_data` `logical` | Returns a data.frame with the count and proportions displayed in the plot.

Value

A ggplot2 object containing a Bar plot.

Examples

```
# Check Suggests.
value <- SCpubr:::check_suggests(function_name = "do_BarPlot", passive = TRUE)

if (isTRUE(value)){
  # Consult the full documentation in https://enblacar.github.io/SCpubr-book/

  # Define your Seurat object.
 sample <- readRDS(system.file("extdata/seurat_dataset_example.rds", package = "SCpubr"))

  # Basic bar plot, horizontal.
  p1 <- SCpubr::do_BarPlot(sample = sample,
                           group.by = "seurat_clusters",
                           legend.position = "none",
                           plot.title = "Number of cells per cluster")

  # Split by a second variable.
  sample$modified_orig.ident <- sample(x = c("Sample_A", "Sample_B", "Sample_C"),
                                       size = ncol(sample),
                                       replace = TRUE,
                                       prob = c(0.2, 0.7, 0.1))
```

```
    p <- SCpubr::do_BarPlot(sample,
                            group.by = "seurat_clusters",
                            split.by = "modified_orig.ident",
                            plot.title = "Number of cells per cluster in each sample",
                            position = "stack")


  } else if (base::isFALSE(value)){
    message("This function can not be used without its suggested packages.")
    message("Check out which ones are needed using `SCpubr::state_dependencies()`.")
  }
```

---

do_BeeSwarmPlot *BeeSwarm plot.*

Description

BeeSwarm plot.

Usage

```
do_BeeSwarmPlot(
  sample,
  feature_to_rank,
  group.by = NULL,
  assay = NULL,
  reduction = NULL,
  slot = NULL,
  continuous_feature = FALSE,
  order = FALSE,
  colors.use = NULL,
  legend.title = NULL,
  legend.type = "colorbar",
  legend.position = "bottom",
  legend.framewidth = 0.5,
  legend.tickwidth = 0.5,
  legend.length = 20,
  legend.width = 1,
  legend.framecolor = "grey50",
  legend.tickcolor = "white",
  legend.ncol = NULL,
  legend.icon.size = 4,
  plot.title = NULL,
  plot.subtitle = NULL,
  plot.caption = NULL,
  xlab = NULL,
```

```
  ylab = NULL,
  font.size = 14,
  font.type = "sans",
  remove_x_axis = FALSE,
  remove_y_axis = FALSE,
  flip = FALSE,
  use_viridis = FALSE,
  viridis.palette = "G",
  viridis.direction = 1,
  sequential.palette = "YlGnBu",
  sequential.direction = 1,
  verbose = TRUE,
  raster = FALSE,
  raster.dpi = 300,
  plot_cell_borders = TRUE,
  border.size = 1.5,
  border.color = "black",
  pt.size = 2,
  min.cutoff = NA,
  max.cutoff = NA,
  na.value = "grey75",
  number.breaks = 5,
  plot.title.face = "bold",
  plot.subtitle.face = "plain",
  plot.caption.face = "italic",
  axis.title.face = "bold",
  axis.text.face = "plain",
  legend.title.face = "bold",
  legend.text.face = "plain"
)
```

Arguments

`sample`	`Seurat` | A Seurat object, generated by CreateSeuratObject.
`feature_to_rank`	`character` | Feature for which the cells are going to be ranked. Ideal case is that this feature is stored as a metadata column.
`group.by`	`character` | Metadata variable to group the output by. Has to be a character of factor column.
`assay`	`character` | Assay to use. Defaults to the current assay.
`reduction`	`character` | Reduction to use. Can be the canonical ones such as "umap", "pca", or any custom ones, such as "diffusion". If you are unsure about which reductions you have, use `Seurat::Reductions(sample)`. Defaults to "umap" if present or to the last computed reduction if the argument is not provided.
`slot`	`character` | Data slot to use. Only one of: counts, data, scale.data. Defaults to "data".

`continuous_feature`	`logical` | Is the feature to rank and color for continuous? I.e: an enrichment score.
`order`	`logical` | Whether to reorder the groups based on the median of the ranking.
`colors.use`	`named_vector` | Named vector of valid color representations (either name of HEX codes) with as many named colors as unique values of group.by. If group.by is not provided, defaults to the unique values of Idents. If not provided, a color scale will be set by default.
`legend.title`	`character` | Title for the legend.
`legend.type`	`character` | Type of legend to display. One of: • `normal`: Default legend displayed by **ggplot2**. • `colorbar`: Redefined colorbar legend, using guide_colorbar.
`legend.position`	`character` | Position of the legend in the plot. One of: • `top`: Top of the figure. • `bottom`: Bottom of the figure. • `left`: Left of the figure. • `right`: Right of the figure. • `none`: No legend is displayed.
`legend.framewidth`, `legend.tickwidth`	`numeric` | Width of the lines of the box in the legend.
`legend.length`, `legend.width`	`numeric` | Length and width of the legend. Will adjust automatically depending on legend side.
`legend.framecolor`	`character` | Color of the lines of the box in the legend.
`legend.tickcolor`	`character` | Color of the ticks of the box in the legend.
`legend.ncol`	`numeric` | Number of columns in the legend.
`legend.icon.size`	`numeric` | Size of the icons in legend.
`plot.title`, `plot.subtitle`, `plot.caption`	`character` | Title, subtitle or caption to use in the plot.
`xlab`, `ylab`	`character` | Titles for the X and Y axis.
`font.size`	`numeric` | Overall font size of the plot. All plot elements will have a size relationship with this font size.
`font.type`	`character` | Base font family for the plot. One of: • `mono`: Mono spaced font. • `serif`: Serif font family. • `sans`: Default font family.
`remove_x_axis`, `remove_y_axis`	`logical` | Remove X axis labels and ticks from the plot.

`flip`	`logical` | Whether to invert the axis of the displayed plot.
`use_viridis`	`logical` | Whether to use viridis color scales.
`viridis.palette`	`character` | A capital letter from A to H or the scale name as in scale_fill_viridis.
`viridis.direction`	`numeric` | Either 1 or -1. Controls how the gradient of viridis scale is formed.
`sequential.palette`	`character` | Type of sequential color palette to use. Out of the sequential palettes defined in `brewer.pal`.
`sequential.direction`	`numeric` | Direction of the sequential color scale. Either 1 or -1.
`verbose`	`logical` | Whether to show extra comments, warnings,etc.
`raster`	`logical` | Whether to raster the resulting plot. This is recommendable if plotting a lot of cells.
`raster.dpi`	`numeric` | Pixel resolution for rasterized plots. Defaults to 1024. Only activates on Seurat versions higher or equal than 4.1.0.
`plot_cell_borders`	`logical` | Whether to plot border around cells.
`border.size`	`numeric` | Width of the border of the cells.
`border.color`	`character` | Color for the border of the heatmap body.
`pt.size`	`numeric` | Size of the dots.
`min.cutoff`, `max.cutoff`	`numeric` | Set the min/max ends of the color scale. Any cell/group with a value lower than min.cutoff will turn into min.cutoff and any cell with a value higher than max.cutoff will turn into max.cutoff. In FeaturePlots, provide as many values as features. Use NAs to skip a feature.
`na.value`	`character` | Color value for NA.
`number.breaks`	`numeric` | Controls the number of breaks in continuous color scales of ggplot2-based plots.
`plot.title.face`, `plot.subtitle.face`, `plot.caption.face`, `axis.title.face`, `axis.text.face`, `legend.title.face`, `legend.text.face`	`character` | Controls the style of the font for the corresponding theme element. One of: • `plain`: For normal text. • `italic`: For text in itallic. • `bold`: For text in bold. • `bold.italic`: For text both in itallic and bold.

Value

A ggplot2 object containing a Bee Swarm plot.

Examples

```
# Check Suggests.
value <- SCpubr:::check_suggests(function_name = "do_BeeSwarmPlot", passive = TRUE)

if (isTRUE(value)){
  # Consult the full documentation in https://enblacar.github.io/SCpubr-book/

  # Define your Seurat object.
 sample <- readRDS(system.file("extdata/seurat_dataset_example.rds", package = "SCpubr"))

  # Basic Bee Swarm plot - categorical coloring.
  # This will color based on the unique values of seurat_clusters.
  p <- SCpubr::do_BeeSwarmPlot(sample = sample,
                               feature_to_rank = "PC_1",
                               group.by = "seurat_clusters",
                               continuous_feature = FALSE)

  # Basic Bee Swarm plot - continuous coloring.
  # This will color based on the PC_1 values.
  p <- SCpubr::do_BeeSwarmPlot(sample = sample,
                               feature_to_rank = "PC_1",
                               group.by = "seurat_clusters",
                               continuous_feature = TRUE)
} else if (base::isFALSE(value)){
  message("This function can not be used without its suggested packages.")
  message("Check out which ones are needed using `SCpubr::state_dependencies()`.")
}
```

`do_BoxPlot` *Generate Box Plots.*

Description

Generate Box Plots.

Usage

```
do_BoxPlot(
  sample,
  feature,
  group.by = NULL,
  split.by = NULL,
  assay = NULL,
  slot = "data",
  font.size = 14,
  font.type = "sans",
  axis.text.x.angle = 45,
```

```
    colors.use = NULL,
    na.value = "grey75",
    plot.title = NULL,
    plot.subtitle = NULL,
    plot.caption = NULL,
    xlab = NULL,
    ylab = NULL,
    legend.title = NULL,
    legend.title.position = "top",
    legend.position = "bottom",
    boxplot.line.color = "black",
    outlier.color = "black",
    outlier.alpha = 0.5,
    boxplot.linewidth = 0.5,
    boxplot.width = NULL,
    plot.grid = TRUE,
    grid.color = "grey75",
    grid.type = "dashed",
    flip = FALSE,
    order = FALSE,
    use_silhouette = FALSE,
    use_test = FALSE,
    comparisons = NULL,
    test = "wilcox.test",
    map_signif_level = TRUE,
    plot.title.face = "bold",
    plot.subtitle.face = "plain",
    plot.caption.face = "italic",
    axis.title.face = "bold",
    axis.text.face = "plain",
    legend.title.face = "bold",
    legend.text.face = "plain"
  )
```

Arguments

`sample`	`Seurat` | A Seurat object, generated by CreateSeuratObject.
`feature`	`character` | Feature to represent.
`group.by`	`character` | Metadata variable to group the output by. Has to be a character of factor column.
`split.by`	`character` | Secondary metadata variable to further group (split) the output by. Has to be a character of factor column.
`assay`	`character` | Assay to use. Defaults to the current assay.
`slot`	`character` | Data slot to use. Only one of: counts, data, scale.data. Defaults to "data".
`font.size`	`numeric` | Overall font size of the plot. All plot elements will have a size relationship with this font size.

`font.type`	`character` | Base font family for the plot. One of: • `mono`: Mono spaced font. • `serif`: Serif font family. • `sans`: Default font family.
`axis.text.x.angle`	`numeric` | Degree to rotate the X labels. One of: 0, 45, 90.
`colors.use`	`named_vector` | Named vector of valid color representations (either name of HEX codes) with as many named colors as unique values of group.by. If group.by is not provided, defaults to the unique values of Idents. If not provided, a color scale will be set by default.
`na.value`	`character` | Color value for NA.
`plot.title`, `plot.subtitle`, `plot.caption`	`character` | Title, subtitle or caption to use in the plot.
`xlab`, `ylab`	`character` | Titles for the X and Y axis.
`legend.title`	`character` | Title for the legend.
`legend.title.position`	`character` | Position for the title of the legend. One of: • `top`: Top of the legend. • `bottom`: Bottom of the legend. • `left`: Left of the legend. • `right`: Right of the legend.
`legend.position`	`character` | Position of the legend in the plot. One of: • `top`: Top of the figure. • `bottom`: Bottom of the figure. • `left`: Left of the figure. • `right`: Right of the figure. • `none`: No legend is displayed.
`boxplot.line.color`	`character` | Color of the borders of the boxplots if use_silhouette is FALSE.
`outlier.color`	`character` | Color of the outlier dots.
`outlier.alpha`	`numeric` | Alpha applied to the outliers.
`boxplot.linewidth`	`numeric` | Width of the lines in the boxplots. Also controls the lines of the tests applied if use_test is set to true.
`boxplot.width`	`numeric` | Width of the boxplots.
`plot.grid`	`logical` | Whether to plot grid lines.
`grid.color`	`character` | Color of the grid in the plot. In heatmaps, color of the border of the cells.
`grid.type`	`character` | One of the possible linetype options: • `blank`. • `solid`.

- `dashed`.
- `dotted`.
- `dotdash`.
- `longdash`.
- `twodash`.

`flip`	`logical` | Whether to invert the axis of the displayed plot.
`order`	`logical` | Whether to order the boxplots by average values. Can not be used alongside split.by.
`use_silhouette`	`logical` | Whether to color the borders of the boxplots instead of the inside area.
`use_test`	`logical` | Whether to apply a statistical test to a given pair of elements. Can not be used alongside split.by.
`comparisons`	A list of length-2 vectors. The entries in the vector are either the names of 2 values on the x-axis or the 2 integers that correspond to the index of the columns of interest.
`test`	the name of the statistical test that is applied to the values of the 2 columns (e.g. `t.test`, `wilcox.test` etc.). If you implement a custom test make sure that it returns a list that has an entry called `p.value`.
`map_signif_level`	Boolean value, if the p-value are directly written as annotation or asterisks are used instead. Alternatively one can provide a named numeric vector to create custom mappings from p-values to annotation: For example: `c("***"=0.001, "**"=0.01, "*"=0.05)`. Alternatively, one can provide a function that takes a numeric argument (the p-value) and returns a string.
`plot.title.face`, `plot.subtitle.face`, `plot.caption.face`, `axis.title.face`, `axis.text.face`, `legend.title.face`, `legend.text.face`	`character` | Controls the style of the font for the corresponding theme element. One of: • `plain`: For normal text. • `italic`: For text in itallic. • `bold`: For text in bold. • `bold.italic`: For text both in itallic and bold.

Value

A ggplot2 object.

Examples

```
# Check Suggests.
value <- SCpubr:::check_suggests(function_name = "do_BoxPlot", passive = TRUE)

if (isTRUE(value)){
  # Consult the full documentation in https://enblacar.github.io/SCpubr-book/

  # Define your Seurat object.
 sample <- readRDS(system.file("extdata/seurat_dataset_example.rds", package = "SCpubr"))
```

```
    # Basic box plot.
    p <- SCpubr::do_BoxPlot(sample = sample,
                            feature = "nCount_RNA")
    p

    # Use silhouette style.
    p <- SCpubr::do_BoxPlot(sample = sample,
                            feature = "nCount_RNA",
                            use_silhouette = TRUE)
    p

    # Order by mean values.
    p <- SCpubr::do_BoxPlot(sample = sample,
                            feature = "nCount_RNA",
                            order = TRUE)
    p

    # Apply second grouping.
   sample$orig.ident <- ifelse(sample$seurat_clusters %in% c("0", "1", "2", "3"), "A", "B")
    p <- SCpubr::do_BoxPlot(sample = sample,
                            feature = "nCount_RNA",
                            split.by = "orig.ident")
    p

    # Apply statistical tests.
    p <- SCpubr::do_BoxPlot(sample = sample,
                            feature = "nCount_RNA",
                            group.by = "orig.ident",
                            use_test = TRUE,
                            comparisons = list(c("A", "B")))
    p

  } else if (base::isFALSE(value)){
    message("This function can not be used without its suggested packages.")
    message("Check out which ones are needed using `SCpubr::state_dependencies()`.")
  }
```

`do_CellularStatesPlot` *Cellular States plot.*

Description

This plot aims to show the relationships between distinct enrichment scores. If 3 variables are provided, the relationship is between the Y axis and the dual X axis. If 4 variables are provided, each corner of the plot represents how enriched the cells are in that given list. How to interpret this? In a 3-variable plot, the Y axis just means one variable. The higher the cells are in the Y axis the more enriched they are in that given variable. The X axis is a dual parameter one. Cells falling into each extreme of the axis are highly enriched for either x1 or x2, while cells falling in between are not enriched for any of the two. In a 4-variable plot, each corner shows the enrichment for one of the 4 given features. Cells will tend to locate in either of the four corners, but there will be cases of cells locating mid-way between two given corners (enriched in both features) or in the middle of the plot (not enriched for any).

Usage

```
do_CellularStatesPlot(
  sample,
  input_gene_list,
  x1,
  y1,
  x2 = NULL,
  y2 = NULL,
  group.by = NULL,
  colors.use = NULL,
  legend.position = "bottom",
  legend.icon.size = 4,
  legend.ncol = NULL,
  legend.nrow = NULL,
  legend.byrow = FALSE,
  plot.title = NULL,
  plot.subtitle = NULL,
  plot.caption = NULL,
  font.size = 14,
  font.type = "sans",
  xlab = NULL,
  ylab = NULL,
  axis.ticks = TRUE,
  axis.text = TRUE,
  verbose = FALSE,
  enforce_symmetry = FALSE,
  plot_marginal_distributions = FALSE,
  marginal.type = "density",
  marginal.size = 5,
  marginal.group = TRUE,
  plot_cell_borders = TRUE,
  plot_enrichment_scores = FALSE,
  border.size = 2,
  border.color = "black",
  pt.size = 2,
  raster = FALSE,
  raster.dpi = 1024,
  plot_features = FALSE,
  features = NULL,
  use_viridis = TRUE,
  viridis.palette = "G",
  viridis.direction = 1,
  sequential.palette = "YlGnBu",
```

```
    sequential.direction = -1,
    nbin = 24,
    ctrl = 100,
    number.breaks = 5,
    plot.title.face = "bold",
    plot.subtitle.face = "plain",
    plot.caption.face = "italic",
    axis.title.face = "bold",
    axis.text.face = "plain",
    legend.title.face = "bold",
    legend.text.face = "plain"
  )
```

Arguments

`sample`	`Seurat` | A Seurat object, generated by CreateSeuratObject.
`input_gene_list`	`named_list` | Named list of lists of genes to be used as input.
`x1`	`character` | A name of a list from input_gene_list. First feature in the X axis. Will go on the right side of the X axis if y2 is not provided and top-right quadrant if provided.
`y1`	`character` | A name of a list from input_gene_list. First feature on the Y axis. Will become the Y axis if y2 is not provided and bottom-right quadrant if provided.
`x2`	`character` | A name of a list from input_gene_list. Second feature on the X axis. Will go on the left side of the X axis if y2 is not provided and top-left quadrant if provided.
`y2`	`character` | A name of a list from input_gene_list. Second feature on the Y axis. Will become the bottom-left quadrant if provided.
`group.by`	`character` | Metadata variable to group the output by. Has to be a character of factor column.
`colors.use`	`named_vector` | Named vector of valid color representations (either name of HEX codes) with as many named colors as unique values of group.by. If group.by is not provided, defaults to the unique values of Idents. If not provided, a color scale will be set by default.
`legend.position`	`character` | Position of the legend in the plot. One of: • `top`: Top of the figure. • `bottom`: Bottom of the figure. • `left`: Left of the figure. • `right`: Right of the figure. • none: No legend is displayed.
`legend.icon.size`	`numeric` | Size of the icons in legend.
`legend.ncol`	`numeric` | Number of columns in the legend.

`legend.nrow`	`numeric` | Number of rows in the legend.
`legend.byrow`	`logical` | Whether the legend is filled by row or not.
`plot.title`, `plot.subtitle`, `plot.caption`	`character` | Title, subtitle or caption to use in the plot.
`font.size`	`numeric` | Overall font size of the plot. All plot elements will have a size relationship with this font size.
`font.type`	`character` | Base font family for the plot. One of: • `mono`: Mono spaced font. • `serif`: Serif font family. • `sans`: Default font family.
`xlab`, `ylab`	`character` | Titles for the X and Y axis.
`axis.ticks`	`logical` | Whether to show axis ticks.
`axis.text`	`logical` | Whether to show axis text.
`verbose`	`logical` | Whether to show extra comments, warnings,etc.
`enforce_symmetry`	`logical` | Whether to enforce the plot to follow a symmetry (3 variables, the X axis has 0 as center, 4 variables, all axis have the same range and the plot is squared).
`plot_marginal_distributions`	`logical` | Whether to plot marginal distributions on the figure or not.
`marginal.type`	`character` | One of: • `density`: Compute density plots on the margins. • `histogram`: Compute histograms on the margins. • `boxplot`: Compute boxplot on the margins. • `violin`: Compute violin plots on the margins. • `densigram`: Compute densigram plots on the margins.
`marginal.size`	`numeric` | Size ratio between the main and marginal plots. A value of 5 means that the main plot is 5 times bigger than the marginal plots.
`marginal.group`	`logical` | Whether to group the marginal distribution by group.by or current identities.
`plot_cell_borders`	`logical` | Whether to plot border around cells.
`plot_enrichment_scores`	`logical` | Whether to report enrichment scores for the input lists as plots.
`border.size`	`numeric` | Width of the border of the cells.
`border.color`	`character` | Color for the border of the heatmap body.
`pt.size`	`numeric` | Size of the dots.
`raster`	`logical` | Whether to raster the resulting plot. This is recommendable if plotting a lot of cells.
`raster.dpi`	`numeric` | Pixel resolution for rasterized plots. Defaults to 1024. Only activates on Seurat versions higher or equal than 4.1.0.

`plot_features`	`logical` | Whether to also report any other feature onto the primary plot.
`features`	`character` | Additional features to plot.
`use_viridis`	`logical` | Whether to use viridis color scales.
`viridis.palette`	`character` | A capital letter from A to H or the scale name as in scale_fill_viridis.
`viridis.direction`	`numeric` | Either 1 or -1. Controls how the gradient of viridis scale is formed.
`sequential.palette`	`character` | Type of sequential color palette to use. Out of the sequential palettes defined in `brewer.pal`.
`sequential.direction`	`numeric` | Direction of the sequential color scale. Either 1 or -1.
`nbin`	`numeric` | Number of bins to use in AddModuleScore.
`ctrl`	`numeric` | Number of genes in the control set to use in AddModuleScore.
`number.breaks`	`numeric` | Controls the number of breaks in continuous color scales of ggplot2-based plots.
`plot.title.face`, `plot.subtitle.face`, `plot.caption.face`, `axis.title.face`, `axis.text.face`, `legend.title.face`, `legend.text.face`	`character` | Controls the style of the font for the corresponding theme element. One of: • `plain`: For normal text. • `italic`: For text in itallic. • `bold`: For text in bold. • `bold.italic`: For text both in itallic and bold.

Details

This plots are based on the following publications:

- Neftel, C. *et al.* An Integrative Model of Cellular States, Plasticity, and Genetics for Glioblastoma. Cell 178, 835-849.e21 (2019). doi:10.1016/j.cell.2019.06.024
- Tirosh, I., Venteicher, A., Hebert, C. *et al.* Single-cell RNA-seq supports a developmental hierarchy in human oligodendroglioma. Nature 539, 309–313 (2016). doi:10.1038/nature20123

Value

A ggplot2 object containing a butterfly plot.

Examples

```
 # Check Suggests.
value <- SCpubr:::check_suggests(function_name = "do_CellularStatesPlot", passive = TRUE)

 if (isTRUE(value)){
   # Consult the full documentation in https://enblacar.github.io/SCpubr-book/
```

```
  # Define your Seurat object.
 sample <- readRDS(system.file("extdata/seurat_dataset_example.rds", package = "SCpubr"))

  # Define some gene sets to query. It has to be a named list.
  gene_set <- list("A" = rownames(sample)[1:10],
                   "B" = rownames(sample)[11:20],
                   "C" = rownames(sample)[21:30],
                   "D" = rownames(sample)[31:40])

  # Using two variables: A scatter plot X vs Y.
  p <- SCpubr::do_CellularStatesPlot(sample = sample,
                                     input_gene_list = gene_set,
                                     x1 = "A",
                                     y1 = "B",
                                     nbin = 1,
                                     ctrl = 10)
  p


  # Using three variables. Figure from: https://www.nature.com/articles/nature20123.
  p <- SCpubr::do_CellularStatesPlot(sample = sample,
                                     input_gene_list = gene_set,
                                     x1 = "A",
                                     y1 = "B",
                                     x2 = "C",
                                     nbin = 1,
                                     ctrl = 10)
  p


  # Using four variables. Figure from: https://pubmed.ncbi.nlm.nih.gov/31327527/
  p <- SCpubr::do_CellularStatesPlot(sample = sample,
                                     input_gene_list = gene_set,
                                     x1 = "A",
                                     y1 = "C",
                                     x2 = "B",
                                     y2 = "D",
                                     nbin = 1,
                                     ctrl = 10)
  p
} else if (base::isFALSE(value)){
  message("This function can not be used without its suggested packages.")
  message("Check out which ones are needed using `SCpubr::state_dependencies()`.")
}
```

`do_ChordDiagramPlot` *Generate a Chord diagram.*

Description

Generate a Chord diagram.

Usage

```
do_ChordDiagramPlot(
  sample = NULL,
  from = NULL,
  to = NULL,
  colors.from = NULL,
  colors.to = NULL,
  big.gap = 10,
  small.gap = 1,
  link.border.color = NA,
  link.border.width = 1,
  highlight_group = NULL,
  alpha.highlight = 25,
  link.sort = NULL,
  link.decreasing = TRUE,
  z_index = FALSE,
  self.link = 1,
  symmetric = FALSE,
  directional = 1,
  direction.type = c("diffHeight", "arrows"),
  link.arr.type = "big.arrow",
  scale = FALSE,
  alignment = "default",
  annotationTrack = c("grid", "axis"),
  padding_labels = 4,
  ...
)
```

Arguments

`sample`	`Seurat` | A Seurat object, generated by CreateSeuratObject.
`from, to`	`character` | Categorical metadata variable to be used as origin and end points of the interactions.
`colors.from, colors.to`	`named_vector` | Named vector of colors corresponding to the unique values of "from" and "to".
`big.gap`	`numeric` | Space between the groups in "from" and "to".
`small.gap`	`numeric` | Space within the groups.
`link.border.color`	`character` | Color for the border of the links. NA = no color.
`link.border.width`	`numeric` | Width of the border line of the links.

`highlight_group`	`character` | A value from from that will be used to highlight only the links coming from it.
`alpha.highlight`	`numeric` | A value between 00 (double digits) and 99 to depict the alpha of the highlighted links. No transparency needs "FF"
`link.sort`	pass to `chordDiagramFromMatrix` or `chordDiagramFromDataFrame`
`link.decreasing`	pass to `chordDiagramFromMatrix` or `chordDiagramFromDataFrame`
`z_index`	`logical` | Whether to bring the bigger links to the top.
`self.link`	`numeric` | Behavior of the links. One of: • `1`: Prevents self linking. • `2`: Allows self linking.
`symmetric`	pass to `chordDiagramFromMatrix`
`directional`	`numeric` | Set the direction of the links. One of: • `0`: Non-directional data. • `1`: Links go from "from" to "to". • `-1`: Links go from "to" to "from". • `2`: Links go in both directions.
`direction.type`	`character` | How to display the directions. One of: • `diffHeight`: Sets a line at the origin of the group showing to how many groups and in which proportion this group is linked to. • `arrows`: Sets the connection as arrows. • `both`: Sets up both behaviors. Use as: `c("diffHeight", "arrows")`.
`link.arr.type`	`character` | Sets the appearance of the arrows. One of: • `triangle`: Arrow with a triangle tip at the end displayed on top of the link. • `big.arrow`: The link itself ends in a triangle shape.
`scale`	`logical` | Whether to put all nodes the same width.
`alignment`	`character` | How to align the diagram. One of: • `default`: Allows **circlize** to set up the plot as it sees fit. • `horizontal`: Sets the break between "from" and "to" groups on the horizontal axis. • `vertical`: Sets the break between "from" and "to" groups on the vertical axis.
`annotationTrack`	pass to `chordDiagramFromMatrix` or `chordDiagramFromDataFrame`
`padding_labels`	`numeric` | Number of extra padding (white spaces) of the labels so that they do not overlap with the scales.
`...`	For internal use only.

Value

A circlize plot.

Examples

```
# Check Suggests.
value <- SCpubr:::check_suggests(function_name = "do_ChordDiagramPlot", passive = TRUE)

if (isTRUE(value)){
  # Consult the full documentation in https://enblacar.github.io/SCpubr-book/

  # Define your Seurat object.
 sample <- readRDS(system.file("extdata/seurat_dataset_example.rds", package = "SCpubr"))

  # Basic chord diagram.
  sample$assignment <- ifelse(sample$seurat_clusters %in% c("0", "4", "7"), "A", "B")
  sample$assignment[sample$seurat_clusters %in% c("1", "2")] <- "C"
  sample$assignment[sample$seurat_clusters %in% c("10", "5")] <- "D"
  sample$assignment[sample$seurat_clusters %in% c("8", "9")] <- "E"

  p <- SCpubr::do_ChordDiagramPlot(sample = sample,
                                   from = "seurat_clusters",
                                   to = "assignment")

  p

} else if (base::isFALSE(value)){
  message("This function can not be used without its suggested packages.")
  message("Check out which ones are needed using `SCpubr::state_dependencies()`.")
}
```

`do_ColorPalette` *Generate color scales based on a value.*

Description

This function is an adaptation of colortools package. As the package was removed from CRAN on 23-06-2022, this utility function came to existence in order to cover the gap. It is, on its basis, an adaptation of the package into a single function. Original code, developed by Gaston Sanchez, can be found in: https://github.com/gastonstat/colortools

Usage

```
do_ColorPalette(
  colors.use,
  n = 12,
  opposite = FALSE,
  adjacent = FALSE,
  triadic = FALSE,
  split_complementary = FALSE,
  tetradic = FALSE,
```

```
    square = FALSE,
    complete_output = FALSE,
    plot = FALSE,
    font.size = 14,
    font.type = "sans"
  )
```

Arguments

`colors.use`	`character` | One color upon which generate the color scale. Can be a name or a HEX code.
`n`	`numeric` | Number of colors to include in the color wheel. Use it when all other options are FALSE, otherwise, it becomes 12.
`opposite`	`logical` | Return the opposing color to the one provided.
`adjacent`	`logical` | Return the adjacent colors to the one provided.
`triadic`	`logical` | Return the triadic combination of colors to the one provided.
`split_complementary`	`logical` | Return the split complementary combination of colors to the one provided.
`tetradic`	`logical` | Return the tetradic combination of colors to the one provided.
`square`	`logical` | Return the square combination of colors to the one provided.
`complete_output`	`logical` | Runs all the previous options and returns all the outputs as a list that contains all color vectors, all plots and a combined plot with everything.
`plot`	`logical` | Whether to also return a plot displaying the values instead of a vector with the color.
`font.size`	`numeric` | Overall font size of the plot. All plot elements will have a size relationship with this font size.
`font.type`	`character` | Base font family for the plot. One of: • `mono`: Mono spaced font. • `serif`: Serif font family. • `sans`: Default font family.

Value

A character vector with the desired color scale.

Examples

```
# Check Suggests.
value <- SCpubr:::check_suggests(function_name = "do_ColorPalette", passive = TRUE)

if (isTRUE(value)){
  # Generate a color wheel based on a single value.
  colors <- SCpubr::do_ColorPalette(colors.use = "steelblue")
  p <- SCpubr::do_ColorPalette(colors.use = "steelblue",
```

```
                             plot = TRUE)

# Generate a pair of opposite colors based on a given one.
colors <- SCpubr::do_ColorPalette(colors.use = "steelblue",
                                  opposite = TRUE)
p <- SCpubr::do_ColorPalette(colors.use = "steelblue",
                             opposite = TRUE,
                             plot = TRUE)

# Generate a trio of adjacent colors based on a given one.
colors <- SCpubr::do_ColorPalette(colors.use = "steelblue",
                                  adjacent = TRUE)
p <- SCpubr::do_ColorPalette(colors.use = "steelblue",
                             adjacent = TRUE,
                             plot = TRUE)

# Generate a trio of triadic colors based on a given one.
colors <- SCpubr::do_ColorPalette(colors.use = "steelblue",
                                  triadic = TRUE)
p <- SCpubr::do_ColorPalette(colors.use = "steelblue",
                             triadic = TRUE,
                             plot = TRUE)

# Generate a trio of split complementary colors based on a given one.
colors <- SCpubr::do_ColorPalette(colors.use = "steelblue",
                                  split_complementary = TRUE)
p <- SCpubr::do_ColorPalette(colors.use = "steelblue",
                             split_complementary = TRUE,
                             plot = TRUE)

# Generate a group of tetradic colors based on a given one.
colors <- SCpubr::do_ColorPalette(colors.use = "steelblue",
                                  tetradic = TRUE)
p <- SCpubr::do_ColorPalette(colors.use = "steelblue",
                             tetradic = TRUE,
                             plot = TRUE)

# Generate a group of square colors based on a given one.
colors <- SCpubr::do_ColorPalette(colors.use = "steelblue",
                                  square = TRUE)
p <- SCpubr::do_ColorPalette(colors.use = "steelblue",
                             square = TRUE,
                             plot = TRUE)

# Retrieve the output of all options.
out <- SCpubr::do_ColorPalette(colors.use = "steelblue",
                               complete_output = TRUE)
## Retrieve the colors.
colors <- out$colors
## Retrieve the plots.
plots <- out$plots
## Retrieve a combined plot with all the options.
p <- out$combined_plot
```

```
} else if (base::isFALSE(value)){
  message("This function can not be used without its suggested packages.")
  message("Check out which ones are needed using `SCpubr::state_dependencies()`.")
}
```

---

do_CopyNumberVariantPlot
Display CNV scores from inferCNV as Feature Plots.

Description

Display CNV scores from inferCNV as Feature Plots.

Usage

```
do_CopyNumberVariantPlot(
  sample,
  infercnv_object,
  chromosome_locations,
  group.by = NULL,
  using_metacells = FALSE,
  metacell_mapping = NULL,
  legend.type = "colorbar",
  legend.position = "bottom",
  legend.length = 20,
  legend.width = 1,
  legend.framewidth = 0.5,
  legend.tickwidth = 0.5,
  legend.framecolor = "grey50",
  legend.tickcolor = "white",
  font.size = 14,
  pt.size = 1,
  font.type = "sans",
  axis.text.x.angle = 45,
  enforce_symmetry = TRUE,
  legend.title = NULL,
  na.value = "grey75",
  viridis.palette = "G",
  viridis.direction = 1,
  verbose = FALSE,
  min.cutoff = NA,
  max.cutoff = NA,
  number.breaks = 5,
  diverging.palette = "RdBu",
  diverging.direction = -1,
```

```
  sequential.palette = "YlGnBu",
  sequential.direction = -1,
  use_viridis = TRUE,
  return_object = FALSE,
  grid.color = "white",
  border.color = "black",
  flip = FALSE,
  plot.title.face = "bold",
  plot.subtitle.face = "plain",
  plot.caption.face = "italic",
  axis.title.face = "bold",
  axis.text.face = "plain",
  legend.title.face = "bold",
  legend.text.face = "plain"
)
```

Arguments

`sample`	`Seurat` | A Seurat object, generated by CreateSeuratObject.
`infercnv_object`	`infercnv` | Output inferCNV object run on the same Seurat object.
`chromosome_locations`	`tibble` | Tibble containing the chromosome regions to use. Can be obtained using `utils::data("human_chr_locations", package = "SCpubr")`.
`group.by`	`character` | Metadata variable to group the output by. Has to be a character of factor column.
`using_metacells`	`logical` | Whether inferCNV was run using metacells or not.
`metacell_mapping`	`named_vector` | Vector or cell - metacell mapping.
`legend.type`	`character` | Type of legend to display. One of: • `normal`: Default legend displayed by **ggplot2**. • `colorbar`: Redefined colorbar legend, using guide_colorbar.
`legend.position`	`character` | Position of the legend in the plot. One of: • `top`: Top of the figure. • `bottom`: Bottom of the figure. • `left`: Left of the figure. • `right`: Right of the figure. • `none`: No legend is displayed.
`legend.length, legend.width`	`numeric` | Length and width of the legend. Will adjust automatically depending on legend side.
`legend.framewidth, legend.tickwidth`	`numeric` | Width of the lines of the box in the legend.

`legend.framecolor`	`character` | Color of the lines of the box in the legend.
`legend.tickcolor`	`character` | Color of the ticks of the box in the legend.
`font.size`	`numeric` | Overall font size of the plot. All plot elements will have a size relationship with this font size.
`pt.size`	`numeric` | Size of the dots.
`font.type`	`character` | Base font family for the plot. One of: • `mono`: Mono spaced font. • `serif`: Serif font family. • `sans`: Default font family.
`axis.text.x.angle`	`numeric` | Degree to rotate the X labels. One of: 0, 45, 90.
`enforce_symmetry`	`logical` | Return a symmetrical plot axes-wise or continuous color scale-wise, when applicable.
`legend.title`	`character` | Title for the legend.
`na.value`	`character` | Color value for NA.
`viridis.palette`	`character` | A capital letter from A to H or the scale name as in scale_fill_viridis.
`viridis.direction`	`numeric` | Either 1 or -1. Controls how the gradient of viridis scale is formed.
`verbose`	`logical` | Whether to show extra comments, warnings,etc.
`min.cutoff`, `max.cutoff`	`numeric` | Set the min/max ends of the color scale. Any cell/group with a value lower than min.cutoff will turn into min.cutoff and any cell with a value higher than max.cutoff will turn into max.cutoff. In FeaturePlots, provide as many values as features. Use NAs to skip a feature.
`number.breaks`	`numeric` | Controls the number of breaks in continuous color scales of ggplot2-based plots.
`diverging.palette`	`character` | Type of symmetrical color palette to use. Out of the diverging palettes defined in `brewer.pal`.
`diverging.direction`	`numeric` | Either 1 or -1. Direction of the divering palette. This basically flips the two ends.
`sequential.palette`	`character` | Type of sequential color palette to use. Out of the sequential palettes defined in `brewer.pal`.
`sequential.direction`	`numeric` | Direction of the sequential color scale. Either 1 or -1.
`use_viridis`	`logical` | Whether to use viridis color scales.
`return_object`	`logical` | Returns the Seurat object with the modifications performed in the function. Nomally, this contains a new assay with the data that can then be used for any other visualization desired.

`grid.color`	`character` | Color of the grid in the plot. In heatmaps, color of the border of the cells.
`border.color`	`character` | Color for the border of the heatmap body.
`flip`	`logical` | Whether to invert the axis of the displayed plot.

`plot.title.face`, `plot.subtitle.face`, `plot.caption.face`, `axis.title.face`, `axis.text.face`, `legend.title.face`, `legend.text.face`

`character` | Controls the style of the font for the corresponding theme element. One of:

- `plain`: For normal text.
- `italic`: For text in itallic.
- `bold`: For text in bold.
- `bold.italic`: For text both in itallic and bold.

Value

A list containing Feature Plots for different chromosome regions and corresponding dot plots by groups..

Examples

```
 # Check Suggests.
value <- SCpubr:::check_suggests(function_name = "do_CopyNumberVariantPlot", passive = TRUE)

 if (isTRUE(value)){
   # Consult the full documentation in https://enblacar.github.io/SCpubr-book/

   # This function expects that you have run inferCNV on your
   # own and you have access to the output object.

   # Define your Seurat object.
   sample <- readRDS(system.file("extdata/seurat_dataset_example.rds",
                                 package = "SCpubr"))

   # Define your inferCNV object.
   infercnv_object <- readRDS(system.file("extdata/infercnv_object_example.rds",
                                          package = "SCpubr"))

   # Get human chromosome locations.
   chromosome_locations = SCpubr::human_chr_locations

   # Compute for a all chromosomes.
   p <- SCpubr::do_CopyNumberVariantPlot(sample = sample,
                                         infercnv_object = infercnv_object,
                                         using_metacells = FALSE,
                                         chromosome_locations = chromosome_locations)

 } else if (base::isFALSE(value)){
   message("This function can not be used without its suggested packages.")
   message("Check out which ones are needed using `SCpubr::state_dependencies()`.")
 }
```

`do_CorrelationPlot` *Create correlation matrix heatmaps.*

Description

Create correlation matrix heatmaps.

Usage

```
do_CorrelationPlot(
  sample = NULL,
  input_gene_list = NULL,
  cluster = TRUE,
  remove.diagonal = TRUE,
  mode = "hvg",
  assay = NULL,
  group.by = NULL,
  legend.title = "Pearson coef.",
  enforce_symmetry = ifelse(mode == "hvg", TRUE, FALSE),
  font.size = 14,
  font.type = "sans",
  na.value = "grey75",
  legend.width = 1,
  legend.length = 20,
  legend.framewidth = 0.5,
  legend.tickwidth = 0.5,
  legend.framecolor = "grey50",
  legend.tickcolor = "white",
  legend.type = "colorbar",
  legend.position = "bottom",
  min.cutoff = NA,
  max.cutoff = NA,
  number.breaks = 5,
  plot.title = NULL,
  plot.subtitle = NULL,
  plot.caption = NULL,
  diverging.palette = "RdBu",
  diverging.direction = -1,
  use_viridis = FALSE,
  viridis.palette = "G",
  viridis.direction = -1,
  sequential.palette = "YlGnBu",
  sequential.direction = 1,
  axis.text.x.angle = 45,
  grid.color = "white",
```

```
  border.color = "black",
  plot.title.face = "bold",
  plot.subtitle.face = "plain",
  plot.caption.face = "italic",
  axis.title.face = "bold",
  axis.text.face = "plain",
  legend.title.face = "bold",
  legend.text.face = "plain"
)
```

Arguments

`sample`	`Seurat` | A Seurat object, generated by CreateSeuratObject.
`input_gene_list`	`named_list` | Named list of lists of genes to be used as input.
`cluster`	`logical` | Whether to cluster the elements in the heatmap or not.
`remove.diagonal`	`logical` | Whether to convert diagnoal to NA. Normally this value would be 1, heavily shifting the color scale.
`mode`	`character` | Different types of correlation matrices can be computed. Right now, the only possible value is "hvg", standing for Highly Variable Genes. The sample is subset for the HVG and the data is re-scaled. Scale data is used for the correlation.
`assay`	`character` | Assay to use. Defaults to the current assay.
`group.by`	`character` | Metadata variable to group the output by. Has to be a character of factor column.
`legend.title`	`character` | Title for the legend.
`enforce_symmetry`	`logical` | Return a symmetrical plot axes-wise or continuous color scale-wise, when applicable.
`font.size`	`numeric` | Overall font size of the plot. All plot elements will have a size relationship with this font size.
`font.type`	`character` | Base font family for the plot. One of: • `mono`: Mono spaced font. • `serif`: Serif font family. • `sans`: Default font family.
`na.value`	`character` | Color value for NA.
`legend.length, legend.width`	`numeric` | Length and width of the legend. Will adjust automatically depending on legend side.
`legend.framewidth, legend.tickwidth`	`numeric` | Width of the lines of the box in the legend.
`legend.framecolor`	`character` | Color of the lines of the box in the legend.

`legend.tickcolor`	`character` | Color of the ticks of the box in the legend.
`legend.type`	`character` | Type of legend to display. One of: • `normal`: Default legend displayed by **ggplot2**. • `colorbar`: Redefined colorbar legend, using guide_colorbar.
`legend.position`	`character` | Position of the legend in the plot. One of: • `top`: Top of the figure. • `bottom`: Bottom of the figure. • `left`: Left of the figure. • `right`: Right of the figure. • `none`: No legend is displayed.
`min.cutoff`, `max.cutoff`	`numeric` | Set the min/max ends of the color scale. Any cell/group with a value lower than min.cutoff will turn into min.cutoff and any cell with a value higher than max.cutoff will turn into max.cutoff. In FeaturePlots, provide as many values as features. Use NAs to skip a feature.
`number.breaks`	`numeric` | Controls the number of breaks in continuous color scales of ggplot2-based plots.
`plot.title`, `plot.subtitle`, `plot.caption`	`character` | Title, subtitle or caption to use in the plot.
`diverging.palette`	`character` | Type of symmetrical color palette to use. Out of the diverging palettes defined in `brewer.pal`.
`diverging.direction`	`numeric` | Either 1 or -1. Direction of the divering palette. This basically flips the two ends.
`use_viridis`	`logical` | Whether to use viridis color scales.
`viridis.palette`	`character` | A capital letter from A to H or the scale name as in scale_fill_viridis.
`viridis.direction`	`numeric` | Either 1 or -1. Controls how the gradient of viridis scale is formed.
`sequential.palette`	`character` | Type of sequential color palette to use. Out of the sequential palettes defined in `brewer.pal`.
`sequential.direction`	`numeric` | Direction of the sequential color scale. Either 1 or -1.
`axis.text.x.angle`	`numeric` | Degree to rotate the X labels. One of: 0, 45, 90.
`grid.color`	`character` | Color of the grid in the plot. In heatmaps, color of the border of the cells.
`border.color`	`character` | Color for the border of the heatmap body.

`plot.title.face`, `plot.subtitle.face`, `plot.caption.face`, `axis.title.face`, `axis.text.face`, `legend.title.face`, `legend.text.face`
: `character` | Controls the style of the font for the corresponding theme element. One of:

- `plain`: For normal text.
- `italic`: For text in itallic.
- `bold`: For text in bold.
- `bold.italic`: For text both in itallic and bold.

Value

A ggplot2 object.

Examples

```
# Check Suggests.
value <- SCpubr:::check_suggests(function_name = "do_CorrelationPlot", passive = TRUE)

if (isTRUE(value)){
  # Consult the full documentation in https://enblacar.github.io/SCpubr-book/

  # Define your Seurat object.
 sample <- readRDS(system.file("extdata/seurat_dataset_example.rds", package = "SCpubr"))

  # Default values.
  p <- SCpubr::do_CorrelationPlot(sample = sample)
  p

} else if (base::isFALSE(value)){
  message("This function can not be used without its suggested packages.")
  message("Check out which ones are needed using `SCpubr::state_dependencies()`.")
}
```

`do_DimPlot` *Wrapper for DimPlot.*

Description

Wrapper for DimPlot.

Usage

```
do_DimPlot(
  sample,
  reduction = NULL,
  group.by = NULL,
```

```
  split.by = NULL,
  colors.use = NULL,
  shuffle = TRUE,
  order = NULL,
  raster = FALSE,
  pt.size = 1,
  label = FALSE,
  label.color = "black",
  label.fill = "white",
  label.size = 4,
  label.box = TRUE,
  repel = FALSE,
  cells.highlight = NULL,
  idents.highlight = NULL,
  idents.keep = NULL,
  sizes.highlight = 1,
  ncol = NULL,
  plot.title = NULL,
  plot.subtitle = NULL,
  plot.caption = NULL,
  legend.title = NULL,
  legend.position = "bottom",
  legend.title.position = "top",
  legend.ncol = NULL,
  legend.nrow = NULL,
  legend.icon.size = 4,
  legend.byrow = FALSE,
  raster.dpi = 2048,
  dims = c(1, 2),
  font.size = 14,
  font.type = "sans",
  na.value = "grey75",
  plot_cell_borders = TRUE,
  border.size = 2,
  border.color = "black",
  border.density = 1,
  plot_marginal_distributions = FALSE,
  marginal.type = "density",
  marginal.size = 5,
  marginal.group = TRUE,
  plot.axes = FALSE,
  plot_density_contour = FALSE,
  contour.position = "bottom",
  contour.color = "grey90",
  contour.lineend = "butt",
  contour.linejoin = "round",
  contour_expand_axes = 0.25,
  plot.title.face = "bold",
```

```
  plot.subtitle.face = "plain",
  plot.caption.face = "italic",
  axis.title.face = "bold",
  axis.text.face = "plain",
  legend.title.face = "bold",
  legend.text.face = "plain"
)
```

Arguments

`sample`	`Seurat` | A Seurat object, generated by CreateSeuratObject.
`reduction`	`character` | Reduction to use. Can be the canonical ones such as "umap", "pca", or any custom ones, such as "diffusion". If you are unsure about which reductions you have, use `Seurat::Reductions(sample)`. Defaults to "umap" if present or to the last computed reduction if the argument is not provided.
`group.by`	`character` | Metadata variable to group the output by. Has to be a character of factor column.
`split.by`	`character` | Secondary metadata variable to further group (split) the output by. Has to be a character of factor column.
`colors.use`	`named_vector` | Named vector of valid color representations (either name of HEX codes) with as many named colors as unique values of group.by. If group.by is not provided, defaults to the unique values of Idents. If not provided, a color scale will be set by default.
`shuffle`	`logical` | Whether to shuffle the cells or not, so that they are not plotted cluster-wise. Recommended.
`order`	`character` | Vector of identities to be plotted. Either one with all identities or just some, which will be plotted last.
`raster`	`logical` | Whether to raster the resulting plot. This is recommendable if plotting a lot of cells.
`pt.size`	`numeric` | Size of the dots.
`label`	`logical` | Whether to plot the cluster labels in the UMAP. The cluster labels will have the same color as the cluster colors.
`label.color`	`character` | Color of the labels in the plot.
`label.fill`	`character` | Color to fill the labels. Has to be a single color, that will be used for all labels. If `NULL`, the colors of the clusters will be used instead.
`label.size`	`numeric` | Size of the labels in the plot.
`label.box`	`logical` | Whether to plot the plot labels as `geom_text` (FALSE) or `geom_label` (TRUE).
`repel`	`logical` | Whether to repel the text labels.
`cells.highlight, idents.highlight`	`character` | Vector of cells/identities to focus into. The identities have to much those in `Seurat::Idents(sample)` The rest of the cells will be grayed out. Both parameters can be used at the same time.

`idents.keep`	`character` | Vector of identities to keep. This will effectively set the rest of the cells that do not match the identities provided to NA, therefore coloring them according to na.value parameter.
`sizes.highlight`	`numeric` | Point size of highlighted cells using cells.highlight parameter.
`ncol`	`numeric` | Number of columns used in the arrangement of the output plot using "split.by" parameter.
`plot.title`, `plot.subtitle`, `plot.caption`	`character` | Title, subtitle or caption to use in the plot.
`legend.title`	`character` | Title for the legend.
`legend.position`	`character` | Position of the legend in the plot. One of: • `top`: Top of the figure. • `bottom`: Bottom of the figure. • `left`: Left of the figure. • `right`: Right of the figure. • none: No legend is displayed.
`legend.title.position`	`character` | Position for the title of the legend. One of: • `top`: Top of the legend. • `bottom`: Bottom of the legend. • `left`: Left of the legend. • `right`: Right of the legend.
`legend.ncol`	`numeric` | Number of columns in the legend.
`legend.nrow`	`numeric` | Number of rows in the legend.
`legend.icon.size`	`numeric` | Size of the icons in legend.
`legend.byrow`	`logical` | Whether the legend is filled by row or not.
`raster.dpi`	`numeric` | Pixel resolution for rasterized plots. Defaults to 1024. Only activates on Seurat versions higher or equal than 4.1.0.
`dims`	`numeric` | Vector of 2 numerics indicating the dimensions to plot out of the selected reduction. Defaults to c(1, 2) if not specified.
`font.size`	`numeric` | Overall font size of the plot. All plot elements will have a size relationship with this font size.
`font.type`	`character` | Base font family for the plot. One of: • mono: Mono spaced font. • `serif`: Serif font family. • sans: Default font family.
`na.value`	`character` | Color value for NA.
`plot_cell_borders`	`logical` | Whether to plot border around cells.
`border.size`	`numeric` | Width of the border of the cells.

`border.color`	`character` | Color for the border of the heatmap body.
`border.density`	`numeric` | Controls the number of cells used when `plot_cell_borders = TRUE`. Value between 0 and 1. It computes a 2D kernel density and based on this cells that have a density below the specified quantile will be used to generate the cluster contour. The lower this number, the less cells will be selected, thus reducing the overall size of the plot but also potentially preventing all the contours to be properly drawn.
`plot_marginal_distributions`	`logical` | Whether to plot marginal distributions on the figure or not.
`marginal.type`	`character` | One of: • `density`: Compute density plots on the margins. • `histogram`: Compute histograms on the margins. • `boxplot`: Compute boxplot on the margins. • `violin`: Compute violin plots on the margins. • `densigram`: Compute densigram plots on the margins.
`marginal.size`	`numeric` | Size ratio between the main and marginal plots. A value of 5 means that the main plot is 5 times bigger than the marginal plots.
`marginal.group`	`logical` | Whether to group the marginal distribution by group.by or current identities.
`plot.axes`	`logical` | Whether to plot axes or not.
`plot_density_contour`	`logical` | Whether to plot density contours in the UMAP.
`contour.position`	`character` | Whether to plot density contours on top or at the bottom of the visualization layers, thus overlapping the clusters/cells or not.
`contour.color`	`character` | Color of the density lines.
`contour.lineend`	`character` | Line end style (round, butt, square).
`contour.linejoin`	`character` | Line join style (round, mitre, bevel).
`contour_expand_axes`	`numeric` | To make the contours fit the plot, the limits of the X and Y axis are expanding a given percentage from the min and max values for each axis. This controls such percentage.
`plot.title.face`, `plot.subtitle.face`, `plot.caption.face`, `axis.title.face`, `axis.text.face`, `legend.title.face`, `legend.text.face`	`character` | Controls the style of the font for the corresponding theme element. One of: • `plain`: For normal text. • `italic`: For text in itallic. • `bold`: For text in bold. • `bold.italic`: For text both in itallic and bold.

Value

A ggplot2 object containing a DimPlot.

Examples

```
# Check Suggests.
value <- SCpubr:::check_suggests(function_name = "do_DimPlot", passive = TRUE)

if (isTRUE(value)){
  # Consult the full documentation in https://enblacar.github.io/SCpubr-book/

  # Define your Seurat object.
 sample <- readRDS(system.file("extdata/seurat_dataset_example.rds", package = "SCpubr"))

  # Basic DimPlot.
  p <- SCpubr::do_DimPlot(sample = sample)


  # Restrict the amount of identities displayed.
  p <- SCpubr::do_DimPlot(sample = sample,
                          idents.keep = c("1", "3", "5"))

  # Group by another variable rather than `Seurat::Idents(sample)`
  p <- SCpubr::do_DimPlot(sample = sample,
                          group.by = "seurat_clusters")

  # Split the output in as many plots as unique identities.
  p <- SCpubr::do_DimPlot(sample = sample,
                          split.by = "seurat_clusters")

  # Highlight given identities
  p <- SCpubr::do_DimPlot(sample,
                          idents.highlight = c("1", "3"))

} else if (base::isFALSE(value)){
  message("This function can not be used without its suggested packages.")
  message("Check out which ones are needed using `SCpubr::state_dependencies()`.")
}
```

`do_DotPlot` *This function is a wrapper for DotPlot. It provides most of its functionalities while adding extra. You can*

Description

This function is a wrapper for DotPlot. It provides most of its functionalities while adding extra. You can

Usage

```
do_DotPlot(
  sample,
  features,
  assay = NULL,
  group.by = NULL,
  scale = FALSE,
  legend.title = NULL,
  legend.type = "colorbar",
  legend.position = "bottom",
  legend.framewidth = 0.5,
  legend.tickwidth = 0.5,
  legend.length = 20,
  legend.width = 1,
  legend.framecolor = "grey50",
  legend.tickcolor = "white",
  colors.use = NULL,
  dot.scale = 6,
  plot.title = NULL,
  plot.subtitle = NULL,
  plot.caption = NULL,
  xlab = NULL,
  ylab = NULL,
  font.size = 14,
  font.type = "sans",
  cluster = FALSE,
  flip = FALSE,
  axis.text.x.angle = 45,
  scale.by = "size",
  use_viridis = FALSE,
  viridis.palette = "G",
  viridis.direction = -1,
  sequential.palette = "YlGnBu",
  sequential.direction = 1,
  na.value = "grey75",
  dot_border = TRUE,
  plot.grid = TRUE,
  grid.color = "grey75",
  grid.type = "dashed",
  number.breaks = 5,
  plot.title.face = "bold",
  plot.subtitle.face = "plain",
  plot.caption.face = "italic",
  axis.title.face = "bold",
  axis.text.face = "plain",
  legend.title.face = "bold",
  legend.text.face = "plain"
)
```

Arguments

`sample`	`Seurat` | A Seurat object, generated by CreateSeuratObject.
`features`	`character` | Features to represent.
`assay`	`character` | Assay to use. Defaults to the current assay.
`group.by`	`character` | Metadata variable to group the output by. Has to be a character of factor column.
`scale`	`logical` | Whether the data should be scaled or not. Non-scaled data allows for comparison across genes. Scaled data allows for an easier comparison along the same gene.
`legend.title`	`character` | Title for the legend.
`legend.type`	`character` | Type of legend to display. One of: • `normal`: Default legend displayed by **ggplot2**. • `colorbar`: Redefined colorbar legend, using guide_colorbar.
`legend.position`	`character` | Position of the legend in the plot. One of: • `top`: Top of the figure. • `bottom`: Bottom of the figure. • `left`: Left of the figure. • `right`: Right of the figure. • `none`: No legend is displayed.
`legend.framewidth`, `legend.tickwidth`	`numeric` | Width of the lines of the box in the legend.
`legend.length`, `legend.width`	`numeric` | Length and width of the legend. Will adjust automatically depending on legend side.
`legend.framecolor`	`character` | Color of the lines of the box in the legend.
`legend.tickcolor`	`character` | Color of the ticks of the box in the legend.
`colors.use`	`named_vector` | Named vector of valid color representations (either name of HEX codes) with as many named colors as unique values of group.by. If group.by is not provided, defaults to the unique values of Idents. If not provided, a color scale will be set by default.
`dot.scale`	`numeric` | Scale the size of the dots.
`plot.title`, `plot.subtitle`, `plot.caption`	`character` | Title, subtitle or caption to use in the plot.
`xlab`, `ylab`	`character` | Titles for the X and Y axis.
`font.size`	`numeric` | Overall font size of the plot. All plot elements will have a size relationship with this font size.
`font.type`	`character` | Base font family for the plot. One of: • `mono`: Mono spaced font. • `serif`: Serif font family.

	• `sans`: Default font family.
`cluster`	`logical` | Whether to cluster the identities based on the expression of the features.
`flip`	`logical` | Whether to invert the axis of the displayed plot.
`axis.text.x.angle`	`numeric` | Degree to rotate the X labels. One of: 0, 45, 90.
`scale.by`	`character` | How to scale the size of the dots. One of: • `radius`: use radius aesthetic. • `size`: use size aesthetic.
`use_viridis`	`logical` | Whether to use viridis color scales.
`viridis.palette`	`character` | A capital letter from A to H or the scale name as in scale_fill_viridis.
`viridis.direction`	`numeric` | Either 1 or -1. Controls how the gradient of viridis scale is formed.
`sequential.palette`	`character` | Type of sequential color palette to use. Out of the sequential palettes defined in `brewer.pal`.
`sequential.direction`	`numeric` | Direction of the sequential color scale. Either 1 or -1.
`na.value`	`character` | Color value for NA.
`dot_border`	`logical` | Whether to plot a border around dots.
`plot.grid`	`logical` | Whether to plot grid lines.
`grid.color`	`character` | Color of the grid in the plot. In heatmaps, color of the border of the cells.
`grid.type`	`character` | One of the possible linetype options: • `blank`. • `solid`. • `dashed`. • `dotted`. • `dotdash`. • `longdash`. • `twodash`.
`number.breaks`	`numeric` | Controls the number of breaks in continuous color scales of ggplot2-based plots.
`plot.title.face`, `plot.subtitle.face`, `plot.caption.face`, `axis.title.face`, `axis.text.face`, `legend.title.face`, `legend.text.face`	`character` | Controls the style of the font for the corresponding theme element. One of: • `plain`: For normal text. • `italic`: For text in itallic. • `bold`: For text in bold. • `bold.italic`: For text both in itallic and bold.

Value

A ggplot2 object containing a Dot Plot.

Examples

```
# Check Suggests.
value <- SCpubr:::check_suggests(function_name = "do_DotPlot", passive = TRUE)

if (isTRUE(value)){
  # Define your Seurat object.
 sample <- readRDS(system.file("extdata/seurat_dataset_example.rds", package = "SCpubr"))

  # Basic Dot plot.
  p <- SCpubr::do_DotPlot(sample = sample,
                          features = "EPC1")

  # Querying multiple features.
  genes <- rownames(sample)[1:14]
  p <- SCpubr::do_DotPlot(sample = sample,
                          features = genes)

  # Inverting the axes.
  p <- SCpubr::do_DotPlot(sample = sample,
                          features = genes,
                          cluster = TRUE,
                          plot.title = "Clustered",
                          flip = TRUE)

  # Modifying default colors.
  # Two colors to generate a gradient.
  p <- SCpubr::do_DotPlot(sample = sample,
                          features = genes,
                          colors.use = c("#001219", "#e9d8a6"))

  # Querying multiple features as a named list - splitting by each item in list.
  # Genes have to be unique.
  genes <- list("Naive CD4+ T" = rownames(sample)[1:2],
                "EPC1+ Mono" = rownames(sample)[3:4],
                "Memory CD4+" = rownames(sample)[5],
                "B" = rownames(sample)[6],
                "CD8+ T" = rownames(sample)[7],
                "FCGR3A+ Mono" = rownames(sample)[8:9],
                "NK" = rownames(sample)[10:11],
                "DC" = rownames(sample)[12:13],
                "Platelet" = rownames(sample)[14])

  p <- SCpubr::do_DotPlot(sample = sample,
                          features = genes)

  # Clustering the identities.
  p <- SCpubr::do_DotPlot(sample = sample,
                          features = genes,
```

```
                              cluster = TRUE,
                              plot.title = "Clustered")
} else if (base::isFALSE(value)){
  message("This function can not be used without its suggested packages.")
  message("Check out which ones are needed using `SCpubr::state_dependencies()`.")
}
```

`do_EnrichmentHeatmap` *Create enrichment scores heatmaps.*

Description

This function computes the enrichment scores for the cells using AddModuleScore and then aggregates the scores by the metadata variables provided by the user and displays it as a heatmap, computed by Heatmap.

Usage

```
do_EnrichmentHeatmap(
  sample,
  input_gene_list,
  features.order = NULL,
  groups.order = NULL,
  cluster = TRUE,
  scale_scores = TRUE,
  assay = NULL,
  slot = NULL,
  reduction = NULL,
  group.by = NULL,
  verbose = FALSE,
  na.value = "grey75",
  legend.position = "bottom",
  use_viridis = FALSE,
  viridis.palette = "G",
  viridis.direction = 1,
  legend.framewidth = 0.5,
  legend.tickwidth = 0.5,
  legend.length = 20,
  legend.width = 1,
  legend.framecolor = "grey50",
  legend.tickcolor = "white",
  legend.type = "colorbar",
  font.size = 14,
  font.type = "sans",
  axis.text.x.angle = 45,
  enforce_symmetry = FALSE,
```

```
  nbin = 24,
  ctrl = 100,
  flavor = "Seurat",
  legend.title = NULL,
  ncores = 1,
  storeRanks = TRUE,
  min.cutoff = NA,
  max.cutoff = NA,
  pt.size = 1,
  plot_cell_borders = TRUE,
  border.size = 2,
  return_object = FALSE,
  number.breaks = 5,
  sequential.palette = "YlGnBu",
  diverging.palette = "RdBu",
  diverging.direction = -1,
  sequential.direction = 1,
  flip = FALSE,
  grid.color = "white",
  border.color = "black",
  plot.title.face = "bold",
  plot.subtitle.face = "plain",
  plot.caption.face = "italic",
  axis.title.face = "bold",
  axis.text.face = "plain",
  legend.title.face = "bold",
  legend.text.face = "plain"
)
```

Arguments

`sample`	`Seurat` | A Seurat object, generated by CreateSeuratObject.
`input_gene_list`	`named_list` | Named list of lists of genes to be used as input.
`features.order`	`character` | Should the gene sets be ordered in a specific way? Provide it as a vector of characters with the same names as the names of the gene sets.
`groups.order`	`named_list` | Should the groups in theheatmaps be ordered in a specific way? Provide it as a named list (as many lists as values in `group.by`) with the order for each of the elements in the groups.
`cluster`	`logical` | Whether to perform clustering of rows and columns.
`scale_scores`	`logical` | Whether to transform the scores to a range of 0-1 for plotting.
`assay`	`character` | Assay to use. Defaults to the current assay.
`slot`	`character` | Data slot to use. Only one of: counts, data, scale.data. Defaults to "data".
`reduction`	`character` | Reduction to use. Can be the canonical ones such as "umap", "pca", or any custom ones, such as "diffusion". If you are unsure about which re-

	ductions you have, use `Seurat::Reductions(sample)`. Defaults to "umap" if present or to the last computed reduction if the argument is not provided.
`group.by`	`character` | Metadata variable to group the output by. Has to be a character of factor column.
`verbose`	`logical` | Whether to show extra comments, warnings,etc.
`na.value`	`character` | Color value for NA.
`legend.position`	`character` | Position of the legend in the plot. One of: • `top`: Top of the figure. • `bottom`: Bottom of the figure. • `left`: Left of the figure. • `right`: Right of the figure. • `none`: No legend is displayed.
`use_viridis`	`logical` | Whether to use viridis color scales.
`viridis.palette`	`character` | A capital letter from A to H or the scale name as in scale_fill_viridis.
`viridis.direction`	`numeric` | Either 1 or -1. Controls how the gradient of viridis scale is formed.
`legend.framewidth`, `legend.tickwidth`	`numeric` | Width of the lines of the box in the legend.
`legend.length`, `legend.width`	`numeric` | Length and width of the legend. Will adjust automatically depending on legend side.
`legend.framecolor`	`character` | Color of the lines of the box in the legend.
`legend.tickcolor`	`character` | Color of the ticks of the box in the legend.
`legend.type`	`character` | Type of legend to display. One of: • `normal`: Default legend displayed by **ggplot2**. • `colorbar`: Redefined colorbar legend, using guide_colorbar.
`font.size`	`numeric` | Overall font size of the plot. All plot elements will have a size relationship with this font size.
`font.type`	`character` | Base font family for the plot. One of: • `mono`: Mono spaced font. • `serif`: Serif font family. • `sans`: Default font family.
`axis.text.x.angle`	`numeric` | Degree to rotate the X labels. One of: 0, 45, 90.
`enforce_symmetry`	`logical` | Whether the geyser and feature plot has a symmetrical color scale.
`nbin`	`numeric` | Number of bins to use in AddModuleScore.
`ctrl`	`numeric` | Number of genes in the control set to use in AddModuleScore.

`flavor`	`character` | One of: Seurat, UCell. Compute the enrichment scores using AddModuleScore or AddModuleScore_UCell.
`legend.title`	`character` | Title for the legend.
`ncores`	`numeric` | Number of cores used to run UCell scoring.
`storeRanks`	`logical` | Whether to store the ranks for faster UCell scoring computations. Might require large amounts of RAM.
`min.cutoff`, `max.cutoff`	`numeric` | Set the min/max ends of the color scale. Any cell/group with a value lower than min.cutoff will turn into min.cutoff and any cell with a value higher than max.cutoff will turn into max.cutoff. In FeaturePlots, provide as many values as features. Use NAs to skip a feature.
`pt.size`	`numeric` | Size of the dots.
`plot_cell_borders`	`logical` | Whether to plot border around cells.
`border.size`	`numeric` | Width of the border of the cells.
`return_object`	`logical` | Return the Seurat object with the enrichment scores stored.
`number.breaks`	`numeric` | Controls the number of breaks in continuous color scales of ggplot2-based plots.
`sequential.palette`	`character` | Type of sequential color palette to use. Out of the sequential palettes defined in `brewer.pal`.
`diverging.palette`	`character` | Type of symmetrical color palette to use. Out of the diverging palettes defined in `brewer.pal`.
`diverging.direction`	`numeric` | Either 1 or -1. Direction of the divering palette. This basically flips the two ends.
`sequential.direction`	`numeric` | Direction of the sequential color scale. Either 1 or -1.
`flip`	`logical` | Whether to invert the axis of the displayed plot.
`grid.color`	`character` | Color of the grid in the plot. In heatmaps, color of the border of the cells.
`border.color`	`character` | Color for the border of the heatmap body.
`plot.title.face`, `plot.subtitle.face`, `plot.caption.face`, `axis.title.face`, `axis.text.face`, `legend.title.face`, `legend.text.face`	`character` | Controls the style of the font for the corresponding theme element. One of: • `plain`: For normal text. • `italic`: For text in itallic. • `bold`: For text in bold. • `bold.italic`: For text both in itallic and bold.

Value

A ggplot2 object.

Examples

```
 # Check Suggests.
value <- SCpubr:::check_suggests(function_name = "do_EnrichmentHeatmap", passive = TRUE)

 if (isTRUE(value)){
   # Consult the full documentation in https://enblacar.github.io/SCpubr-book/

   # Define your Seurat object.
  sample <- readRDS(system.file("extdata/seurat_dataset_example.rds", package = "SCpubr"))

   # Genes have to be unique.
   genes <- list("A" = rownames(sample)[1:5],
                 "B" = rownames(sample)[6:10],
                 "C" = rownames(sample)[11:15])

   # Default parameters.
   p <- SCpubr::do_EnrichmentHeatmap(sample = sample,
                                     input_gene_list = genes,
                                     nbin = 1,
                                     ctrl = 10)
   p

 } else if (base::isFALSE(value)){
   message("This function can not be used without its suggested packages.")
   message("Check out which ones are needed using `SCpubr::state_dependencies()`.")
 }
```

`do_ExpressionHeatmap` *Create heatmaps of averaged expression by groups.*

Description

This function generates a heatmap with averaged expression values by the unique groups of the metadata variables provided by the user.

Usage

```
do_ExpressionHeatmap(
  sample,
  features,
  group.by = NULL,
  assay = NULL,
  cluster = TRUE,
  features.order = NULL,
  groups.order = NULL,
  slot = "data",
  legend.title = "Avg. Expression",
  na.value = "grey75",
```

```
  legend.position = "bottom",
  legend.width = 1,
  legend.length = 20,
  legend.framewidth = 0.5,
  legend.tickwidth = 0.5,
  legend.framecolor = "grey50",
  legend.tickcolor = "white",
  legend.type = "colorbar",
  font.size = 14,
  font.type = "sans",
  axis.text.x.angle = 45,
  enforce_symmetry = FALSE,
  min.cutoff = NA,
  max.cutoff = NA,
  diverging.palette = "RdBu",
  diverging.direction = -1,
  sequential.palette = "YlGnBu",
  sequential.direction = 1,
  number.breaks = 5,
  use_viridis = FALSE,
  viridis.palette = "G",
  viridis.direction = -1,
  flip = FALSE,
  grid.color = "white",
  border.color = "black",
  plot.title.face = "bold",
  plot.subtitle.face = "plain",
  plot.caption.face = "italic",
  axis.title.face = "bold",
  axis.text.face = "plain",
  legend.title.face = "bold",
  legend.text.face = "plain"
)
```

Arguments

`sample`	`Seurat` | A Seurat object, generated by CreateSeuratObject.
`features`	`character` | Features to represent.
`group.by`	`character` | Metadata variable to group the output by. Has to be a character of factor column.
`assay`	`character` | Assay to use. Defaults to the current assay.
`cluster`	`logical` | Whether to perform clustering of rows and columns.
`features.order`	`character` | Should the gene sets be ordered in a specific way? Provide it as a vector of characters with the same names as the names of the gene sets.
`groups.order`	`named_list` | Should the groups in theheatmaps be ordered in a specific way? Provide it as a named list (as many lists as values in `group.by`) with the order for each of the elements in the groups.

`slot`	`character` | Data slot to use. Only one of: counts, data, scale.data. Defaults to "data".
`legend.title`	`character` | Title for the legend.
`na.value`	`character` | Color value for NA.
`legend.position`	`character` | Position of the legend in the plot. One of: • `top`: Top of the figure. • `bottom`: Bottom of the figure. • `left`: Left of the figure. • `right`: Right of the figure. • `none`: No legend is displayed.
`legend.length`, `legend.width`	`numeric` | Length and width of the legend. Will adjust automatically depending on legend side.
`legend.framewidth`, `legend.tickwidth`	`numeric` | Width of the lines of the box in the legend.
`legend.framecolor`	`character` | Color of the lines of the box in the legend.
`legend.tickcolor`	`character` | Color of the ticks of the box in the legend.
`legend.type`	`character` | Type of legend to display. One of: • `normal`: Default legend displayed by **ggplot2**. • `colorbar`: Redefined colorbar legend, using guide_colorbar.
`font.size`	`numeric` | Overall font size of the plot. All plot elements will have a size relationship with this font size.
`font.type`	`character` | Base font family for the plot. One of: • `mono`: Mono spaced font. • `serif`: Serif font family. • `sans`: Default font family.
`axis.text.x.angle`	`numeric` | Degree to rotate the X labels. One of: 0, 45, 90.
`enforce_symmetry`	`logical` | Return a symmetrical plot axes-wise or continuous color scale-wise, when applicable.
`min.cutoff`, `max.cutoff`	`numeric` | Set the min/max ends of the color scale. Any cell/group with a value lower than min.cutoff will turn into min.cutoff and any cell with a value higher than max.cutoff will turn into max.cutoff. In FeaturePlots, provide as many values as features. Use NAs to skip a feature.
`diverging.palette`	`character` | Type of symmetrical color palette to use. Out of the diverging palettes defined in `brewer.pal`.

`diverging.direction`	`numeric` | Either 1 or -1. Direction of the divering palette. This basically flips the two ends.
`sequential.palette`	`character` | Type of sequential color palette to use. Out of the sequential palettes defined in `brewer.pal`.
`sequential.direction`	`numeric` | Direction of the sequential color scale. Either 1 or -1.
`number.breaks`	`numeric` | Controls the number of breaks in continuous color scales of ggplot2-based plots.
`use_viridis`	`logical` | Whether to use viridis color scales.
`viridis.palette`	`character` | A capital letter from A to H or the scale name as in scale_fill_viridis.
`viridis.direction`	`numeric` | Either 1 or -1. Controls how the gradient of viridis scale is formed.
`flip`	`logical` | Whether to invert the axis of the displayed plot.
`grid.color`	`character` | Color of the grid in the plot. In heatmaps, color of the border of the cells.
`border.color`	`character` | Color for the border of the heatmap body.
`plot.title.face`, `plot.subtitle.face`, `plot.caption.face`, `axis.title.face`, `axis.text.face`, `legend.title.face`, `legend.text.face`	`character` | Controls the style of the font for the corresponding theme element. One of: • `plain`: For normal text. • `italic`: For text in itallic. • `bold`: For text in bold. • `bold.italic`: For text both in itallic and bold.

Value

A ggplot2 object.

Examples

```
# Check Suggests.
value <- SCpubr:::check_suggests(function_name = "do_ExpressionHeatmap", passive = TRUE)

if (isTRUE(value)){
  # Consult the full documentation in https://enblacar.github.io/SCpubr-book/

  # Define your Seurat object.
 sample <- readRDS(system.file("extdata/seurat_dataset_example.rds", package = "SCpubr"))

  # Define list of genes.
  genes <- rownames(sample)[1:10]

  # Default parameters.
```

```
    p <- SCpubr::do_ExpressionHeatmap(sample = sample,
                                      features = genes,
                                      viridis.direction = -1)

    p
  } else if (base::isFALSE(value)){
    message("This function can not be used without its suggested packages.")
    message("Check out which ones are needed using `SCpubr::state_dependencies()`.")
  }
```

`do_FeaturePlot` *Wrapper for FeaturePlot.*

Description

Wrapper for FeaturePlot.

Usage

```
do_FeaturePlot(
  sample,
  features,
  assay = NULL,
  reduction = NULL,
  slot = NULL,
  order = FALSE,
  group.by = NULL,
  group.by.colors.use = NULL,
  group.by.legend = NULL,
  group.by.show.dots = TRUE,
  group.by.dot.size = 8,
  group.by.cell_borders = FALSE,
  group.by.cell_borders.alpha = 0.1,
  split.by = NULL,
  idents.keep = NULL,
  cells.highlight = NULL,
  idents.highlight = NULL,
  dims = c(1, 2),
  enforce_symmetry = FALSE,
  symmetry.type = "absolute",
  symmetry.center = NA,
  pt.size = 1,
  font.size = 14,
  font.type = "sans",
  legend.title = NULL,
  legend.type = "colorbar",
  legend.position = "bottom",
```

```
  legend.framewidth = 0.5,
  legend.tickwidth = 0.5,
  legend.length = 20,
  legend.width = 1,
  legend.framecolor = "grey50",
  legend.tickcolor = "white",
  legend.ncol = NULL,
  legend.nrow = NULL,
  legend.byrow = FALSE,
  plot.title = NULL,
  plot.subtitle = NULL,
  plot.caption = NULL,
  individual.titles = NULL,
  individual.subtitles = NULL,
  individual.captions = NULL,
  ncol = NULL,
  use_viridis = FALSE,
  viridis.palette = "G",
  viridis.direction = 1,
  raster = FALSE,
  raster.dpi = 1024,
  plot_cell_borders = TRUE,
  border.size = 2,
  border.color = "black",
  border.density = 1,
  na.value = "grey75",
  verbose = TRUE,
  plot.axes = FALSE,
  min.cutoff = rep(NA, length(features)),
  max.cutoff = rep(NA, length(features)),
  plot_density_contour = FALSE,
  contour.position = "bottom",
  contour.color = "grey90",
  contour.lineend = "butt",
  contour.linejoin = "round",
  contour_expand_axes = 0.25,
  label = FALSE,
  label.color = "black",
  label.size = 4,
  number.breaks = 5,
  diverging.palette = "RdBu",
  diverging.direction = -1,
  sequential.palette = "YlGnBu",
  sequential.direction = 1,
  plot.title.face = "bold",
  plot.subtitle.face = "plain",
  plot.caption.face = "italic",
  axis.title.face = "bold",
```

```
    axis.text.face = "plain",
    legend.title.face = "bold",
    legend.text.face = "plain"
  )
```

Arguments

`sample`	`Seurat` | A Seurat object, generated by CreateSeuratObject.
`features`	`character` | Features to represent.
`assay`	`character` | Assay to use. Defaults to the current assay.
`reduction`	`character` | Reduction to use. Can be the canonical ones such as "umap", "pca", or any custom ones, such as "diffusion". If you are unsure about which reductions you have, use `Seurat::Reductions(sample)`. Defaults to "umap" if present or to the last computed reduction if the argument is not provided.
`slot`	`character` | Data slot to use. Only one of: counts, data, scale.data. Defaults to "data".
`order`	`logical` | Whether to order the cells based on expression.
`group.by`	`character` | Metadata variable based on which cells are grouped. This will effectively introduce a big dot in the center of each cluster, colored using a categorical color scale or with the values provided by the user in `group.by.colors.use`. It will also displays a legend.
`group.by.colors.use`	`character` | Colors to use for the group dots.
`group.by.legend`	`character` | Title for the legend when `group.by` is used. Use `NA` to disable it and `NULL` to use the default column title provided in `group.by`.
`group.by.show.dots`	`logical` | Controls whether to place in the middle of the groups.
`group.by.dot.size`	`numeric` | Size of the dots placed in the middle of the groups.
`group.by.cell_borders`	`logical` | Plots another border around the cells displaying the same color code of the dots displayed with `group.by`. Legend is shown always with alpha = 1 regardless of the alpha settings.
`group.by.cell_borders.alpha`	`numeric` | Controls the transparency of the new borders drawn by `group.by.cell_borders`.
`split.by`	`character` | Secondary metadata variable to further group (split) the output by. Has to be a character of factor column.
`idents.keep`	`character` | Vector of identities to plot. The gradient scale will also be subset to only the values of such identities.
`cells.highlight`, `idents.highlight`	`character` | Vector of cells/identities to focus into. The identities have to much those in `Seurat::Idents(sample)` The rest of the cells will be grayed out. Both parameters can be used at the same time.

`dims`	`numeric` | Vector of 2 numerics indicating the dimensions to plot out of the selected reduction. Defaults to c(1, 2) if not specified.
`enforce_symmetry`	`logical` | Return a symmetrical plot axes-wise or continuous color scale-wise, when applicable.
`symmetry.type`	`character` | Type of symmetry to be enforced. One of: • `absolute`: The highest absolute value will be taken into a account to generate the color scale. Works after `min.cutoff` and `max.cutoff`. • `centered`: Centers the scale around the provided value in `symmetry.center`. Works after `min.cutoff` and `max.cutoff`.
`symmetry.center`	`numeric` | Value upon which the scale will be centered.
`pt.size`	`numeric` | Size of the dots.
`font.size`	`numeric` | Overall font size of the plot. All plot elements will have a size relationship with this font size.
`font.type`	`character` | Base font family for the plot. One of: • `mono`: Mono spaced font. • `serif`: Serif font family. • `sans`: Default font family.
`legend.title`	`character` | Title for the legend.
`legend.type`	`character` | Type of legend to display. One of: • `normal`: Default legend displayed by **ggplot2**. • `colorbar`: Redefined colorbar legend, using guide_colorbar.
`legend.position`	`character` | Position of the legend in the plot. One of: • `top`: Top of the figure. • `bottom`: Bottom of the figure. • `left`: Left of the figure. • `right`: Right of the figure. • `none`: No legend is displayed.
`legend.framewidth`, `legend.tickwidth`	`numeric` | Width of the lines of the box in the legend.
`legend.length`, `legend.width`	`numeric` | Length and width of the legend. Will adjust automatically depending on legend side.
`legend.framecolor`	`character` | Color of the lines of the box in the legend.
`legend.tickcolor`	`character` | Color of the ticks of the box in the legend.
`legend.ncol`	`numeric` | Number of columns in the legend.
`legend.nrow`	`numeric` | Number of rows in the legend.
`legend.byrow`	`logical` | Whether the legend is filled by row or not.

`plot.title`, `plot.subtitle`, `plot.caption`	`character` | Title, subtitle or caption to use in the plot.
`individual.titles`, `individual.subtitles`, `individual.captions`	`character` | Titles or subtitles. for each feature if needed. Either NULL or a vector of equal length of features.
`ncol`	`numeric` | Number of columns used in the arrangement of the output plot using "split.by" parameter.
`use_viridis`	`logical` | Whether to use viridis color scales.
`viridis.palette`	`character` | A capital letter from A to H or the scale name as in scale_fill_viridis.
`viridis.direction`	`numeric` | Either 1 or -1. Controls how the gradient of viridis scale is formed.
`raster`	`logical` | Whether to raster the resulting plot. This is recommendable if plotting a lot of cells.
`raster.dpi`	`numeric` | Pixel resolution for rasterized plots. Defaults to 1024. Only activates on Seurat versions higher or equal than 4.1.0.
`plot_cell_borders`	`logical` | Whether to plot border around cells.
`border.size`	`numeric` | Width of the border of the cells.
`border.color`	`character` | Color for the border of the heatmap body.
`border.density`	`numeric` | Controls the number of cells used when `plot_cell_borders = TRUE`. Value between 0 and 1. It computes a 2D kernel density and based on this cells that have a density below the specified quantile will be used to generate the cluster contour. The lower this number, the less cells will be selected, thus reducing the overall size of the plot but also potentially preventing all the contours to be properly drawn.
`na.value`	`character` | Color value for NA.
`verbose`	`logical` | Whether to show extra comments, warnings,etc.
`plot.axes`	`logical` | Whether to plot axes or not.
`min.cutoff`, `max.cutoff`	`numeric` | Set the min/max ends of the color scale. Any cell/group with a value lower than min.cutoff will turn into min.cutoff and any cell with a value higher than max.cutoff will turn into max.cutoff. In FeaturePlots, provide as many values as features. Use NAs to skip a feature.
`plot_density_contour`	`logical` | Whether to plot density contours in the UMAP.
`contour.position`	`character` | Whether to plot density contours on top or at the bottom of the visualization layers, thus overlapping the clusters/cells or not.
`contour.color`	`character` | Color of the density lines.
`contour.lineend`	`character` | Line end style (round, butt, square).
`contour.linejoin`	`character` | Line join style (round, mitre, bevel).

`contour_expand_axes`	`numeric` | To make the contours fit the plot, the limits of the X and Y axis are expanding a given percentage from the min and max values for each axis. This controls such percentage.
`label`	`logical` | Whether to plot the cluster labels in the UMAP. The cluster labels will have the same color as the cluster colors.
`label.color`	`character` | Color of the labels in the plot.
`label.size`	`numeric` | Size of the labels in the plot.
`number.breaks`	`numeric` | Controls the number of breaks in continuous color scales of ggplot2-based plots.
`diverging.palette`	`character` | Type of symmetrical color palette to use. Out of the diverging palettes defined in `brewer.pal`.
`diverging.direction`	`numeric` | Either 1 or -1. Direction of the divering palette. This basically flips the two ends.
`sequential.palette`	`character` | Type of sequential color palette to use. Out of the sequential palettes defined in `brewer.pal`.
`sequential.direction`	`numeric` | Direction of the sequential color scale. Either 1 or -1.
`plot.title.face`, `plot.subtitle.face`, `plot.caption.face`, `axis.title.face`, `axis.text.face`, `legend.title.face`, `legend.text.face`	`character` | Controls the style of the font for the corresponding theme element. One of: • `plain`: For normal text. • `italic`: For text in itallic. • `bold`: For text in bold. • `bold.italic`: For text both in itallic and bold.

Value

A ggplot2 object containing a Feature Plot.

Examples

```
# Check Suggests.
value <- SCpubr:::check_suggests(function_name = "do_FeaturePlot", passive = TRUE)

if (isTRUE(value)){
  # Consult the full documentation in https://enblacar.github.io/SCpubr-book/

  # Define your Seurat object.
 sample <- readRDS(system.file("extdata/seurat_dataset_example.rds", package = "SCpubr"))

  # Regular FeaturePlot.
  p <- SCpubr::do_FeaturePlot(sample = sample,
```

```
                                features = "nCount_RNA")

    # FeaturePlot with a subset of identities
    # (in Seurat::Idents(sample)) maintaining the original UMAP shape.
    idents.use <- levels(sample)[!(levels(sample) %in% c("2", "5", "8"))]
    p <- SCpubr::do_FeaturePlot(sample = sample,
                                idents.highlight = idents.use,
                                features = c("EPC1"))

    # Splitting the FeaturePlot by a variable and
    # maintaining the color scale and the UMAP shape.
    p <- SCpubr::do_FeaturePlot(sample = sample,
                                features = "EPC1",
                                split.by = "seurat_clusters")

  } else if (base::isFALSE(value)){
    message("This function can not be used without its suggested packages.")
    message("Check out which ones are needed using `SCpubr::state_dependencies()`.")
  }
```

`do_FunctionalAnnotationPlot` *Compute functional annotation plots using GO or KEGG ontologies*

Description

Compute functional annotation plots using GO or KEGG ontologies

Usage

```
do_FunctionalAnnotationPlot(
  genes,
  org.db,
  organism = "hsa",
  database = "GO",
  GO_ontology = "BP",
  min.overlap = NULL,
  p.adjust.cutoff = 0.05,
  pAdjustMethod = "BH",
  minGSSize = 10,
  maxGSSize = 500,
  font.size = 10,
  font.type = "sans",
  axis.text.x.angle = 45,
  xlab = NULL,
  ylab = NULL,
  plot.title = NULL,
```

```
  plot.subtitle = NULL,
  plot.caption = NULL,
  legend.type = "colorbar",
  legend.position = "bottom",
  legend.framewidth = 0.5,
  legend.tickwidth = 0.5,
  legend.length = 10,
  legend.width = 1,
  legend.framecolor = "grey50",
  legend.tickcolor = "white",
  number.breaks = 5,
  return_matrix = FALSE,
  grid.color = "white",
  border.color = "black",
  plot.title.face = "bold",
  plot.subtitle.face = "plain",
  plot.caption.face = "italic",
  axis.title.face = "bold",
  axis.text.face = "plain",
  legend.title.face = "bold",
  legend.text.face = "plain"
)
```

Arguments

`genes`	`character` | Vector of gene symbols to query for functional annotation.
`org.db`	`OrgDB` | Database object to use for the query.
`organism`	`character` | Supported KEGG organism.
`database`	`character` | Database to run the analysis on. One of: • `GO`. • `KEGG`.
`GO_ontology`	`character` | GO ontology to use. One of: • `BP`: For **B**iological **P**rocess. • `MF`: For **M**olecular **F**unction. • `CC`: For **C**ellular **C**omponent.
`min.overlap`	`numeric` | Filter the output result to the terms which are supported by this many genes.
`p.adjust.cutoff`	`numeric` | Significance cutoff used to filter non-significant terms.
`pAdjustMethod`	`character` | Method to adjust for multiple testing. One of: • `holm`. • `hochberg`. • `hommel`. • `bonferroni`. • `BH`.

	• `BY`. • `fdr`. • `none`.
`minGSSize`	`numeric` | Minimal size of genes annotated by Ontology term for testing.
`maxGSSize`	`numeric` | Maximal size of genes annotated for testing.
`font.size`	`numeric` | Overall font size of the plot. All plot elements will have a size relationship with this font size.
`font.type`	`character` | Base font family for the plot. One of: • `mono`: Mono spaced font. • `serif`: Serif font family. • `sans`: Default font family.
`axis.text.x.angle`	`numeric` | Degree to rotate the X labels. One of: 0, 45, 90.
`xlab`, `ylab`	`character` | Titles for the X and Y axis.
`plot.title`, `plot.subtitle`, `plot.caption`	`character` | Title, subtitle or caption to use in the plot.
`legend.type`	`character` | Type of legend to display. One of: • `normal`: Default legend displayed by **ggplot2**. • `colorbar`: Redefined colorbar legend, using guide_colorbar.
`legend.position`	`character` | Position of the legend in the plot. One of: • `top`: Top of the figure. • `bottom`: Bottom of the figure. • `left`: Left of the figure. • `right`: Right of the figure. • `none`: No legend is displayed.
`legend.framewidth`, `legend.tickwidth`	`numeric` | Width of the lines of the box in the legend.
`legend.length`, `legend.width`	`numeric` | Length and width of the legend. Will adjust automatically depending on legend side.
`legend.framecolor`	`character` | Color of the lines of the box in the legend.
`legend.tickcolor`	`character` | Color of the ticks of the box in the legend.
`number.breaks`	`numeric` | Controls the number of breaks in continuous color scales of ggplot2-based plots.
`return_matrix`	`logical` | Returns the matrices with the enriched Terms for further use.
`grid.color`	`character` | Color of the grid in the plot. In heatmaps, color of the border of the cells.
`border.color`	`character` | Color for the border of the heatmap body.

`plot.title.face`, `plot.subtitle.face`, `plot.caption.face`, `axis.title.face`, `axis.text.face`, `legend.title.face`, `legend.text.face`

`character` | Controls the style of the font for the corresponding theme element. One of:

- `plain`: For normal text.
- `italic`: For text in itallic.
- `bold`: For text in bold.
- `bold.italic`: For text both in itallic and bold.

Value

A list containing a heatmap of the presence/absence of the genes in the enriched term, as well as a bar plot, dot plot and tree plot of the enriched terms.

Examples

```
 # Check Suggests.
value <- SCpubr:::check_suggests(function_name = "do_FunctionalAnnotationPlot", passive = TRUE)

 if (isTRUE(value)){
   # Consult the full documentation in https://enblacar.github.io/SCpubr-book/

   # Need to load this library or equivalent.
   suppressMessages(library("org.Hs.eg.db"))

   # Define list of genes to query.
   genes.use <- c("CCR7", "CD14", "LYZ",
                  "S100A4", "MS4A1",
                  "MS4A7", "GNLY", "NKG7", "FCER1A",
                  "CST3", "PPBP")

   # Compute the grouped GO terms.
   out <- SCpubr::do_FunctionalAnnotationPlot(genes = genes.use,
                                              org.db = org.Hs.eg.db)

 } else if (base::isFALSE(value)){
   message("This function can not be used without its suggested packages.")
   message("Check out which ones are needed using `SCpubr::state_dependencies()`.")
 }
```

`do_GeyserPlot` *Generate a Geyser plot.*

Description

A Geyser plot is a custom plot in which we plot continuous values on the Y axis grouped by a categorical value in the X. This is plotted as a dot plot, jittered so that the dots span all the way to the other groups. On top of this, the mean and .66 and .95 of the data is plotted, depicting the overall distribution of the dots. The cells can, then, be colored by a continuous variable (same as Y axis or different) or a categorical one (same as X axis or different).

Usage

```
do_GeyserPlot(
  sample,
  features,
  assay = NULL,
  slot = "data",
  group.by = NULL,
  split.by = NULL,
  enforce_symmetry = FALSE,
  scale_type = "continuous",
  order = TRUE,
  plot_cell_borders = TRUE,
  jitter = 0.45,
  pt.size = 1,
  border.size = 2,
  border.color = "black",
  legend.position = "bottom",
  legend.width = 1,
  legend.length = 20,
  legend.framewidth = 0.5,
  legend.tickwidth = 0.5,
  legend.framecolor = "grey50",
  legend.tickcolor = "white",
  legend.type = "colorbar",
  font.size = 14,
  font.type = "sans",
  axis.text.x.angle = 45,
  viridis.palette = "G",
  viridis.direction = 1,
  colors.use = NULL,
  na.value = "grey75",
  legend.ncol = NULL,
  legend.nrow = NULL,
  legend.icon.size = 4,
  legend.byrow = FALSE,
  legend.title = NULL,
  plot.title = NULL,
  plot.subtitle = NULL,
  plot.caption = NULL,
  xlab = "Groups",
  ylab = feature,
  flip = FALSE,
  min.cutoff = rep(NA, length(features)),
  max.cutoff = rep(NA, length(features)),
  number.breaks = 5,
```

```
  diverging.palette = "RdBu",
  diverging.direction = -1,
  sequential.palette = "YlGnBu",
  sequential.direction = -1,
  use_viridis = TRUE,
  plot.title.face = "bold",
  plot.subtitle.face = "plain",
  plot.caption.face = "italic",
  axis.title.face = "bold",
  axis.text.face = "plain",
  legend.title.face = "bold",
  legend.text.face = "plain"
)
```

Arguments

`sample`	`Seurat` | A Seurat object, generated by CreateSeuratObject.
`features`	`character` | Features to represent.
`assay`	`character` | Assay to use. Defaults to the current assay.
`slot`	`character` | Data slot to use. Only one of: counts, data, scale.data. Defaults to "data".
`group.by`	`character` | Metadata variable to group the output by. Has to be a character of factor column.
`split.by`	`character` | Secondary metadata variable to further group (split) the output by. Has to be a character of factor column.
`enforce_symmetry`	`logical` | Return a symmetrical plot axes-wise or continuous color scale-wise, when applicable.
`scale_type`	`character` | Type of color scale to use. One of: • `categorical`: Use a categorical color scale based on the values of "group.by". • `continuous`: Use a continuous color scale based on the values of "feature".
`order`	`logical` | Whether to order the groups by the median of the data (highest to lowest).
`plot_cell_borders`	`logical` | Whether to plot border around cells.
`jitter`	`numeric` | Amount of jitter in the plot along the X axis. The lower the value, the more compacted the dots are.
`pt.size`	`numeric` | Size of the dots.
`border.size`	`numeric` | Width of the border of the cells.
`border.color`	`character` | Color for the border of the heatmap body.
`legend.position`	`character` | Position of the legend in the plot. One of: • `top`: Top of the figure. • `bottom`: Bottom of the figure.

- `left`: Left of the figure.
- `right`: Right of the figure.
- `none`: No legend is displayed.

`legend.length`, `legend.width`	`numeric` | Length and width of the legend. Will adjust automatically depending on legend side.
`legend.framewidth`, `legend.tickwidth`	`numeric` | Width of the lines of the box in the legend.
`legend.framecolor`	`character` | Color of the lines of the box in the legend.
`legend.tickcolor`	`character` | Color of the ticks of the box in the legend.
`legend.type`	`character` | Type of legend to display. One of: • `normal`: Default legend displayed by **ggplot2**. • `colorbar`: Redefined colorbar legend, using guide_colorbar.
`font.size`	`numeric` | Overall font size of the plot. All plot elements will have a size relationship with this font size.
`font.type`	`character` | Base font family for the plot. One of: • `mono`: Mono spaced font. • `serif`: Serif font family. • `sans`: Default font family.
`axis.text.x.angle`	`numeric` | Degree to rotate the X labels. One of: 0, 45, 90.
`viridis.palette`	`character` | A capital letter from A to H or the scale name as in scale_fill_viridis.
`viridis.direction`	`numeric` | Either 1 or -1. Controls how the gradient of viridis scale is formed.
`colors.use`	`character` | Named vector of colors to use. Has to match the unique values of group.by when scale_type is set to categorical.
`na.value`	`character` | Color value for NA.
`legend.ncol`	`numeric` | Number of columns in the legend.
`legend.nrow`	`numeric` | Number of rows in the legend.
`legend.icon.size`	`numeric` | Size of the icons in legend.
`legend.byrow`	`logical` | Whether the legend is filled by row or not.
`legend.title`	`character` | Title for the legend.
`plot.title`, `plot.subtitle`, `plot.caption`	`character` | Title, subtitle or caption to use in the plot.
`xlab`, `ylab`	`character` | Titles for the X and Y axis.
`flip`	`logical` | Whether to invert the axis of the displayed plot.

`min.cutoff`, `max.cutoff`
: `numeric` | Set the min/max ends of the color scale. Any cell/group with a value lower than min.cutoff will turn into min.cutoff and any cell with a value higher than max.cutoff will turn into max.cutoff. In FeaturePlots, provide as many values as features. Use NAs to skip a feature.

`number.breaks`
: `numeric` | Controls the number of breaks in continuous color scales of ggplot2-based plots.

`diverging.palette`
: `character` | Type of symmetrical color palette to use. Out of the diverging palettes defined in `brewer.pal`.

`diverging.direction`
: `numeric` | Either 1 or -1. Direction of the divering palette. This basically flips the two ends.

`sequential.palette`
: `character` | Type of sequential color palette to use. Out of the sequential palettes defined in `brewer.pal`.

`sequential.direction`
: `numeric` | Direction of the sequential color scale. Either 1 or -1.

`use_viridis`
: `logical` | Whether to use viridis color scales.

`plot.title.face`, `plot.subtitle.face`, `plot.caption.face`, `axis.title.face`, `axis.text.face`, `legend.title.face`, `legend.text.face`
: `character` | Controls the style of the font for the corresponding theme element. One of:

  - `plain`: For normal text.
  - `italic`: For text in itallic.
  - `bold`: For text in bold.
  - `bold.italic`: For text both in itallic and bold.

Details

Special thanks to Christina Blume for coming up with the name of the plot.

Value

Either a plot of a list of plots, depending on the number of features provided.

Examples

```
# Check Suggests.
value <- SCpubr::check_suggests(function_name = "do_GeyserPlot", passive = TRUE)

if (isTRUE(value)){
  # Consult the full documentation in https://enblacar.github.io/SCpubr-book/

  # Define your Seurat object.
 sample <- readRDS(system.file("extdata/seurat_dataset_example.rds", package = "SCpubr"))

  # Geyser plot with categorical color scale.
```

```
    p <- SCpubr::do_GeyserPlot(sample = sample,
                               features = "nCount_RNA",
                               scale_type = "categorical")
    p

    # Geyser plot with continuous color scale.
    p <- SCpubr::do_GeyserPlot(sample = sample,
                               features = "nCount_RNA",
                               scale_type = "continuous")


    p

  } else if (base::isFALSE(value)){
    message("This function can not be used without its suggested packages.")
    message("Check out which ones are needed using `SCpubr::state_dependencies()`.")
  }
```

`do_GroupedGOTermPlot` *Compute an overview of the GO terms associated with the input list of genes.*

Description

Compute an overview of the GO terms associated with the input list of genes.

Usage

```
do_GroupedGOTermPlot(
  genes,
  org.db,
  levels.use = NULL,
  GO_ontology = "BP",
  min.overlap = 3,
  flip = TRUE,
  colors.use = c(Present = "#1e3d59", Absent = "#bccbcd"),
  legend.position = "bottom",
  reverse.levels = TRUE,
  axis.text.x.angle = 45,
  font.size = 10,
  font.type = "sans",
  plot.title = paste0("GO | ", GO_ontology),
  plot.subtitle = NULL,
  plot.caption = NULL,
  verbose = FALSE,
  return_matrices = FALSE,
  grid.color = "white",
```

```
    border.color = "black",
    plot.title.face = "bold",
    plot.subtitle.face = "plain",
    plot.caption.face = "italic",
    axis.title.face = "bold",
    axis.text.face = "plain",
    legend.title.face = "bold",
    legend.text.face = "plain"
  )
```

Arguments

<table>
<tr><td><code>genes</code></td><td><code>character</code> | Vector of gene symbols to query for functional annotation.</td></tr>
<tr><td><code>org.db</code></td><td><code>OrgDB</code> | Database object to use for the query.</td></tr>
<tr><td><code>levels.use</code></td><td><code>numeric</code> | Vector of numerics corresponding to the GO ontology levels to plot. If NULL will compute all recursively until there are no results.</td></tr>
<tr><td><code>GO_ontology</code></td><td><code>character</code> | GO ontology to use. One of:
• <code>BP</code>: For Biological Process.
• <code>MF</code>: For Molecular Function.
• <code>CC</code>: For Cellular Component.</td></tr>
<tr><td><code>min.overlap</code></td><td><code>numeric</code> | Filter the output result to the terms which are supported by this many genes.</td></tr>
<tr><td><code>flip</code></td><td><code>logical</code> | Whether to invert the axis of the displayed plot.</td></tr>
<tr><td><code>colors.use</code></td><td><code>character</code> | Named vector with two colors assigned to the names <code>Present</code> and <code>Absent</code>.</td></tr>
<tr><td><code>legend.position</code></td><td><code>character</code> | Position of the legend in the plot. One of:
• <code>top</code>: Top of the figure.
• <code>bottom</code>: Bottom of the figure.
• <code>left</code>: Left of the figure.
• <code>right</code>: Right of the figure.
• <code>none</code>: No legend is displayed.</td></tr>
<tr><td><code>reverse.levels</code></td><td><code>logical</code> | Whether to place the higher levels first when computing the joint heatmap.</td></tr>
<tr><td><code>axis.text.x.angle</code></td><td><code>numeric</code> | Degree to rotate the X labels. One of: 0, 45, 90.</td></tr>
<tr><td><code>font.size</code></td><td><code>numeric</code> | Overall font size of the plot. All plot elements will have a size relationship with this font size.</td></tr>
<tr><td><code>font.type</code></td><td><code>character</code> | Base font family for the plot. One of:
• <code>mono</code>: Mono spaced font.
• <code>serif</code>: Serif font family.
• <code>sans</code>: Default font family.</td></tr>
<tr><td><code>plot.title, plot.subtitle, plot.caption</code></td><td><code>character</code> | Title, subtitle or caption to use in the plot.</td></tr>
</table>

`verbose`	`logical` | Whether to show extra comments, warnings,etc.
`return_matrices`	`logical` | Returns the matrices of grouped GO terms.
`grid.color`	`character` | Color of the grid in the plot. In heatmaps, color of the border of the cells.
`border.color`	`character` | Color for the border of the heatmap body.
`plot.title.face`, `plot.subtitle.face`, `plot.caption.face`, `axis.title.face`, `axis.text.face`, `legend.title.face`, `legend.text.face`	`character` | Controls the style of the font for the corresponding theme element. One of: • `plain`: For normal text. • `italic`: For text in itallic. • `bold`: For text in bold. • `bold.italic`: For text both in itallic and bold.

Value

A list containing all the matrices for the respective GO levels and all the individual and combined heatmaps.

Examples

```
 # Check Suggests.
value <- SCpubr::check_suggests(function_name = "do_GroupedGOTermPlot", passive = TRUE)

 if (isTRUE(value)){
   # Consult the full documentation in https://enblacar.github.io/SCpubr-book/

   # Need to load this library or equivalent.
   suppressMessages(library("org.Hs.eg.db"))

   # Define list of genes to query.
   genes.use <- c("CCR7", "CD14", "LYZ",
                  "S100A4", "MS4A1",
                  "MS4A7", "GNLY", "NKG7", "FCER1A",
                  "CST3", "PPBP")

   # Compute the grouped GO terms.
   out <- SCpubr::do_GroupedGOTermPlot(genes = genes.use,
                                       org.db = org.Hs.eg.db)
 } else if (base::isFALSE(value)){
   message("This function can not be used without its suggested packages.")
   message("Check out which ones are needed using `SCpubr::state_dependencies()`.")
 }
```

`do_GroupwiseDEPlot` *Compute a heatmap with the results of a group-wise DE analysis.*

Description

Compute a heatmap with the results of a group-wise DE analysis.

Usage

```
do_GroupwiseDEPlot(
  sample,
  de_genes,
  group.by = NULL,
  number.breaks = 5,
  top_genes = 5,
  use_viridis = FALSE,
  viridis.direction = -1,
  viridis.palette.pvalue = "C",
  viridis.palette.logfc = "E",
  viridis.palette.expression = "G",
  sequential.direction = 1,
  sequential.palette.pvalue = "YlGn",
  sequential.palette.logfc = "YlOrRd",
  sequential.palette.expression = "YlGnBu",
  assay = NULL,
  slot = "data",
  legend.position = "bottom",
  legend.width = 1,
  legend.length = 20,
  legend.framewidth = 0.5,
  legend.tickwidth = 0.5,
  legend.framecolor = "grey50",
  legend.tickcolor = "white",
  legend.type = "colorbar",
  font.size = 14,
  font.type = "sans",
  axis.text.x.angle = 45,
  min.cutoff = NA,
  max.cutoff = NA,
  na.value = "grey75",
  grid.color = "white",
  border.color = "black",
  plot.title.face = "bold",
  plot.subtitle.face = "plain",
  plot.caption.face = "italic",
  axis.title.face = "bold",
  axis.text.face = "plain",
```

```
  legend.title.face = "bold",
  legend.text.face = "plain"
)
```

Arguments

`sample`	`Seurat` | A Seurat object, generated by CreateSeuratObject.
`de_genes`	`tibble` | DE genes matrix resulting of running `Seurat::FindAllMarkers()`.
`group.by`	`character` | Metadata variable to group the output by. Has to be a character of factor column.
`number.breaks`	`numeric` | Controls the number of breaks in continuous color scales of ggplot2-based plots.
`top_genes`	`numeric` | Top N differentially expressed (DE) genes by group to retrieve.
`use_viridis`	`logical` | Whether to use viridis color scales.
`viridis.direction`	`numeric` | Either 1 or -1. Controls how the gradient of viridis scale is formed.
`viridis.palette.pvalue`, `viridis.palette.logfc`, `viridis.palette.expression`	`character` | Viridis color palettes for the p-value, logfc and expression heatmaps. A capital letter from A to H or the scale name as in scale_fill_viridis.
`sequential.direction`	`numeric` | Direction of the sequential color scale. Either 1 or -1.
`sequential.palette.pvalue`, `sequential.palette.expression`, `sequential.palette.logfc`	`character` | Sequential palettes for p-value, logfc and expression heatmaps. Type of sequential color palette to use. Out of the sequential palettes defined in `brewer.pal`.
`assay`	`character` | Assay to use. Defaults to the current assay.
`slot`	`character` | Data slot to use. Only one of: counts, data, scale.data. Defaults to "data".
`legend.position`	`character` | Position of the legend in the plot. One of: • `top`: Top of the figure. • `bottom`: Bottom of the figure. • `left`: Left of the figure. • `right`: Right of the figure. • none: No legend is displayed.
`legend.length`, `legend.width`	`numeric` | Length and width of the legend. Will adjust automatically depending on legend side.
`legend.framewidth`, `legend.tickwidth`	`numeric` | Width of the lines of the box in the legend.
`legend.framecolor`	`character` | Color of the lines of the box in the legend.

`legend.tickcolor`	`character` | Color of the ticks of the box in the legend.
`legend.type`	`character` | Type of legend to display. One of: • `normal`: Default legend displayed by **ggplot2**. • `colorbar`: Redefined colorbar legend, using guide_colorbar.
`font.size`	`numeric` | Overall font size of the plot. All plot elements will have a size relationship with this font size.
`font.type`	`character` | Base font family for the plot. One of: • `mono`: Mono spaced font. • `serif`: Serif font family. • `sans`: Default font family.
`axis.text.x.angle`	`numeric` | Degree to rotate the X labels. One of: 0, 45, 90.
`min.cutoff`, `max.cutoff`	`numeric` | Set the min/max ends of the color scale. Any cell/group with a value lower than min.cutoff will turn into min.cutoff and any cell with a value higher than max.cutoff will turn into max.cutoff. In FeaturePlots, provide as many values as features. Use NAs to skip a feature.
`na.value`	`character` | Color value for NA.
`grid.color`	`character` | Color of the grid in the plot. In heatmaps, color of the border of the cells.
`border.color`	`character` | Color for the border of the heatmap body.
`plot.title.face`, `plot.subtitle.face`, `plot.caption.face`, `axis.title.face`, `axis.text.face`, `legend.title.face`, `legend.text.face`	`character` | Controls the style of the font for the corresponding theme element. One of: • `plain`: For normal text. • `italic`: For text in itallic. • `bold`: For text in bold. • `bold.italic`: For text both in itallic and bold.

Value

A heatmap composed of 3 main panels: -log10(adjusted p-value), log2(FC) and mean expression by cluster.

Examples

```
# Check Suggests.
value <- SCpubr::check_suggests(function_name = "do_GroupwiseDEPlot", passive = TRUE)

if (isTRUE(value)){
  # Consult the full documentation in https://enblacar.github.io/SCpubr-book/

  # Define your Seurat object.
```

```
   sample <- readRDS(system.file("extdata/seurat_dataset_example.rds", package = "SCpubr"))

    # Compute DE genes and transform to a tibble.
    de_genes <- readRDS(system.file("extdata/de_genes_example.rds", package = "SCpubr"))

    # Default output.
    p <- SCpubr::do_GroupwiseDEPlot(sample = sample,
                                    de_genes = de_genes)

    p

  } else if (base::isFALSE(value)){
    message("This function can not be used without its suggested packages.")
    message("Check out which ones are needed using `SCpubr::state_dependencies()`.")
  }
```

`do_NebulosaPlot` *Wrapper for Nebulosa::plot_density in Seurat.*

Description

Wrapper for Nebulosa::plot_density in Seurat.

Usage

```
do_NebulosaPlot(
  sample,
  features,
  slot = NULL,
  dims = c(1, 2),
  pt.size = 1,
  reduction = NULL,
  combine = TRUE,
  method = c("ks", "wkde"),
  joint = FALSE,
  return_only_joint = FALSE,
  plot.title = NULL,
  plot.subtitle = NULL,
  plot.caption = NULL,
  legend.type = "colorbar",
  legend.framewidth = 0.5,
  legend.tickwidth = 0.5,
  legend.length = 20,
  legend.width = 1,
  legend.framecolor = "grey50",
  legend.tickcolor = "white",
  font.size = 14,
```

```
    font.type = "sans",
    legend.position = "bottom",
    plot_cell_borders = TRUE,
    border.size = 2,
    border.color = "black",
    viridis.palette = "G",
    viridis.direction = 1,
    verbose = TRUE,
    na.value = "grey75",
    plot.axes = FALSE,
    number.breaks = 5,
    use_viridis = FALSE,
    sequential.palette = "YlGnBu",
    sequential.direction = 1,
    plot.title.face = "bold",
    plot.subtitle.face = "plain",
    plot.caption.face = "italic",
    axis.title.face = "bold",
    axis.text.face = "plain",
    legend.title.face = "bold",
    legend.text.face = "plain"
  )
```

Arguments

`sample`	`Seurat` | A Seurat object, generated by CreateSeuratObject.
`features`	`character` | Features to represent.
`slot`	`character` | Data slot to use. Only one of: counts, data, scale.data. Defaults to "data".
`dims`	`numeric` | Vector of 2 numerics indicating the dimensions to plot out of the selected reduction. Defaults to c(1, 2) if not specified.
`pt.size`	`numeric` | Size of the dots.
`reduction`	`character` | Reduction to use. Can be the canonical ones such as "umap", "pca", or any custom ones, such as "diffusion". If you are unsure about which reductions you have, use `Seurat::Reductions(sample)`. Defaults to "umap" if present or to the last computed reduction if the argument is not provided.
`combine`	`logical` | Whether to create a single plot out of multiple features.
`method`	Kernel density estimation method: • `ks`: Computes density using the `kde` function from the `ks` package. • `wkde`: Computes density using a modified version of the `kde2d` function from the `MASS` package to allow weights. Bandwidth selection from the `ks` package is used instead.
`joint`	`logical` | Whether to plot different features as joint density.
`return_only_joint`	`logical` | Whether to only return the joint density panel.

`plot.title`, `plot.subtitle`, `plot.caption`	`character` | Title, subtitle or caption to use in the plot.
`legend.type`	`character` | Type of legend to display. One of: • `normal`: Default legend displayed by **ggplot2**. • `colorbar`: Redefined colorbar legend, using guide_colorbar.
`legend.framewidth`, `legend.tickwidth`	`numeric` | Width of the lines of the box in the legend.
`legend.length`, `legend.width`	`numeric` | Length and width of the legend. Will adjust automatically depending on legend side.
`legend.framecolor`	`character` | Color of the lines of the box in the legend.
`legend.tickcolor`	`character` | Color of the ticks of the box in the legend.
`font.size`	`numeric` | Overall font size of the plot. All plot elements will have a size relationship with this font size.
`font.type`	`character` | Base font family for the plot. One of: • `mono`: Mono spaced font. • `serif`: Serif font family. • `sans`: Default font family.
`legend.position`	`character` | Position of the legend in the plot. One of: • `top`: Top of the figure. • `bottom`: Bottom of the figure. • `left`: Left of the figure. • `right`: Right of the figure. • `none`: No legend is displayed.
`plot_cell_borders`	`logical` | Whether to plot border around cells.
`border.size`	`numeric` | Width of the border of the cells.
`border.color`	`character` | Color for the border of the heatmap body.
`viridis.palette`	`character` | A capital letter from A to H or the scale name as in scale_fill_viridis.
`viridis.direction`	`numeric` | Either 1 or -1. Controls how the gradient of viridis scale is formed.
`verbose`	`logical` | Whether to show extra comments, warnings,etc.
`na.value`	`character` | Color value for NA.
`plot.axes`	`logical` | Whether to plot axes or not.
`number.breaks`	`numeric` | Controls the number of breaks in continuous color scales of ggplot2-based plots.
`use_viridis`	`logical` | Whether to use viridis color scales.

`sequential.palette`
: `character` | Type of sequential color palette to use. Out of the sequential palettes defined in `brewer.pal`.

`sequential.direction`
: `numeric` | Direction of the sequential color scale. Either 1 or -1.

`plot.title.face`, `plot.subtitle.face`, `plot.caption.face`, `axis.title.face`, `axis.text.face`, `legend.title.face`, `legend.text.face`
: `character` | Controls the style of the font for the corresponding theme element. One of:

  - `plain`: For normal text.
  - `italic`: For text in itallic.
  - `bold`: For text in bold.
  - `bold.italic`: For text both in itallic and bold.

Value

A ggplot2 object containing a Nebulosa plot.

Examples

```
# Check Suggests.
value <- SCpubr:::check_suggests(function_name = "do_NebulosaPlot", passive = TRUE)

if (isTRUE(value)){
  # Consult the full documentation in https://enblacar.github.io/SCpubr-book/

  # Define your Seurat object.
 sample <- readRDS(system.file("extdata/seurat_dataset_example.rds", package = "SCpubr"))

  # Basic Nebulosa plot.
  p <- SCpubr::do_NebulosaPlot(sample = sample,
                               features = "EPC1")

  # Compute joint density.
  p <- SCpubr::do_NebulosaPlot(sample = sample,
                               features = c("EPC1", "TOX2"),
                               joint = TRUE)

} else if (base::isFALSE(value)){
  message("This function can not be used without its suggested packages.")
  message("Check out which ones are needed using `SCpubr::state_dependencies()`.")
}
```

`do_PathwayActivityPlot` *Plot Pathway Activities from decoupleR using Progeny prior knowledge.*

Description

Plot Pathway Activities from decoupleR using Progeny prior knowledge.

Usage

```
do_PathwayActivityPlot(
  sample,
  activities,
  group.by = NULL,
  split.by = NULL,
  slot = "scale.data",
  statistic = "norm_wmean",
  pt.size = 1,
  border.size = 2,
  na.value = "grey75",
  legend.position = "bottom",
  legend.width = 1,
  legend.length = 20,
  legend.framewidth = 0.5,
  legend.tickwidth = 0.5,
  legend.framecolor = "grey50",
  legend.tickcolor = "white",
  legend.type = "colorbar",
  font.size = 14,
  font.type = "sans",
  axis.text.x.angle = 45,
  enforce_symmetry = TRUE,
  min.cutoff = NA,
  max.cutoff = NA,
  number.breaks = 5,
  diverging.palette = "RdBu",
  diverging.direction = -1,
  use_viridis = FALSE,
  viridis.palette = "G",
  viridis.direction = -1,
  sequential.palette = "YlGnBu",
  sequential.direction = 1,
  flip = FALSE,
  return_object = FALSE,
  grid.color = "white",
  border.color = "black",
  plot.title.face = "bold",
  plot.subtitle.face = "plain",
  plot.caption.face = "italic",
  axis.title.face = "bold",
  axis.text.face = "plain",
  legend.title.face = "bold",
  legend.text.face = "plain"
```

```
)
```

Arguments

`sample`	`Seurat` | A Seurat object, generated by CreateSeuratObject.
`activities`	`tibble` | Result of running decoupleR method with progeny regulon prior knowledge.
`group.by`	`character` | Metadata variable to group the output by. Has to be a character of factor column.
`split.by`	`character` | Secondary metadata variable to further group (split) the output by. Has to be a character of factor column.
`slot`	`character` | Data slot to use. Only one of: counts, data, scale.data. Defaults to "data".
`statistic`	`character` | DecoupleR statistic to use. One of: • `wmean`: For weighted mean. • `norm_wmean`: For normalized weighted mean. • `corr_wmean`: For corrected weighted mean.
`pt.size`	`numeric` | Size of the dots.
`border.size`	`numeric` | Width of the border of the cells.
`na.value`	`character` | Color value for NA.
`legend.position`	`character` | Position of the legend in the plot. One of: • `top`: Top of the figure. • `bottom`: Bottom of the figure. • `left`: Left of the figure. • `right`: Right of the figure. • `none`: No legend is displayed.
`legend.length`, `legend.width`	`numeric` | Length and width of the legend. Will adjust automatically depending on legend side.
`legend.framewidth`, `legend.tickwidth`	`numeric` | Width of the lines of the box in the legend.
`legend.framecolor`	`character` | Color of the lines of the box in the legend.
`legend.tickcolor`	`character` | Color of the ticks of the box in the legend.
`legend.type`	`character` | Type of legend to display. One of: • `normal`: Default legend displayed by **ggplot2**. • `colorbar`: Redefined colorbar legend, using guide_colorbar.
`font.size`	`numeric` | Overall font size of the plot. All plot elements will have a size relationship with this font size.
`font.type`	`character` | Base font family for the plot. One of:

- `mono`: Mono spaced font.
- `serif`: Serif font family.
- `sans`: Default font family.

`axis.text.x.angle`	`numeric` | Degree to rotate the X labels. One of: 0, 45, 90.
`enforce_symmetry`	`logical` | Return a symmetrical plot axes-wise or continuous color scale-wise, when applicable.
`min.cutoff`, `max.cutoff`	`numeric` | Set the min/max ends of the color scale. Any cell/group with a value lower than min.cutoff will turn into min.cutoff and any cell with a value higher than max.cutoff will turn into max.cutoff. In FeaturePlots, provide as many values as features. Use NAs to skip a feature.
`number.breaks`	`numeric` | Controls the number of breaks in continuous color scales of ggplot2-based plots.
`diverging.palette`	`character` | Type of symmetrical color palette to use. Out of the diverging palettes defined in `brewer.pal`.
`diverging.direction`	`numeric` | Either 1 or -1. Direction of the divering palette. This basically flips the two ends.
`use_viridis`	`logical` | Whether to use viridis color scales.
`viridis.palette`	`character` | A capital letter from A to H or the scale name as in scale_fill_viridis.
`viridis.direction`	`numeric` | Either 1 or -1. Controls how the gradient of viridis scale is formed.
`sequential.palette`	`character` | Type of sequential color palette to use. Out of the sequential palettes defined in `brewer.pal`.
`sequential.direction`	`numeric` | Direction of the sequential color scale. Either 1 or -1.
`flip`	`logical` | Whether to invert the axis of the displayed plot.
`return_object`	`logical` | Returns the Seurat object with the modifications performed in the function. Nomally, this contains a new assay with the data that can then be used for any other visualization desired.
`grid.color`	`character` | Color of the grid in the plot. In heatmaps, color of the border of the cells.
`border.color`	`character` | Color for the border of the heatmap body.
`plot.title.face`, `plot.subtitle.face`, `plot.caption.face`, `axis.title.face`, `axis.text.face`, `legend.title.face`, `legend.text.face`	`character` | Controls the style of the font for the corresponding theme element. One of: • `plain`: For normal text. • `italic`: For text in itallic. • `bold`: For text in bold. • `bold.italic`: For text both in itallic and bold.

Value

A ggplot2 object.

Examples

```
 # Check Suggests.
value <- SCpubr:::check_suggests(function_name = "do_PathwayActivityPlot", passive = TRUE)

 if (isTRUE(value)){
   # Consult the full documentation in https://enblacar.github.io/SCpubr-book/

   # Define your Seurat object.
   sample <- readRDS(system.file("extdata/seurat_dataset_example.rds",
                                 package = "SCpubr"))

   # Define your activities object.
   progeny_activities <- readRDS(system.file("extdata/progeny_activities_example.rds",
                                             package = "SCpubr"))

   # General heatmap.
   out <- SCpubr::do_PathwayActivityPlot(sample = sample,
                                         activities = progeny_activities)
   p <- out$heatmaps$average_scores
   p

 } else if (base::isFALSE(value)){
   message("This function can not be used without its suggested packages.")
   message("Check out which ones are needed using `SCpubr::state_dependencies()`.")
 }
```

`do_RidgePlot` *Create ridge plots.*

Description

This function computes ridge plots based on the **ggridges** package.

Usage

```
do_RidgePlot(
  sample,
  feature,
  group.by = NULL,
  split.by = NULL,
  assay = "SCT",
  slot = "data",
  continuous_scale = FALSE,
  legend.title = NULL,
```

```
  legend.ncol = NULL,
  legend.nrow = NULL,
  legend.byrow = FALSE,
  legend.position = NULL,
  legend.width = 1,
  legend.length = 20,
  legend.framewidth = 0.5,
  legend.tickwidth = 0.5,
  legend.framecolor = "grey50",
  legend.tickcolor = "white",
  legend.type = "colorbar",
  colors.use = NULL,
  font.size = 14,
  font.type = "sans",
  axis.text.x.angle = 45,
  plot.title = NULL,
  plot.subtitle = NULL,
  plot.caption = NULL,
  xlab = NULL,
  ylab = NULL,
  compute_quantiles = FALSE,
  compute_custom_quantiles = FALSE,
  quantiles = c(0.25, 0.5, 0.75),
  compute_distribution_tails = FALSE,
  prob_tails = 0.025,
  color_by_probabilities = FALSE,
  use_viridis = TRUE,
  viridis.palette = "G",
  viridis.direction = 1,
  sequential.palette = "YlGnBu",
  sequential.direction = 1,
  plot.grid = TRUE,
  grid.color = "grey75",
  grid.type = "dashed",
  flip = FALSE,
  number.breaks = 5,
  plot.title.face = "bold",
  plot.subtitle.face = "plain",
  plot.caption.face = "italic",
  axis.title.face = "bold",
  axis.text.face = "plain",
  legend.title.face = "bold",
  legend.text.face = "plain"
)
```

Arguments

`sample`	`Seurat` | A Seurat object, generated by CreateSeuratObject.

`feature`	`character` | Feature to represent.
`group.by`	`character` | Metadata variable to group the output by. Has to be a character of factor column.
`split.by`	`character` | Secondary metadata variable to further group (split) the output by. Has to be a character of factor column.
`assay`	`character` | Assay to use. Defaults to the current assay.
`slot`	`character` | Data slot to use. Only one of: counts, data, scale.data. Defaults to "data".
`continuous_scale`	`logical` | Whether to color the ridges depending on a categorical or continuous scale.
`legend.title`	`character` | Title for the legend.
`legend.ncol`	`numeric` | Number of columns in the legend.
`legend.nrow`	`numeric` | Number of rows in the legend.
`legend.byrow`	`logical` | Whether the legend is filled by row or not.
`legend.position`	`character` | Position of the legend in the plot. One of: • `top`: Top of the figure. • `bottom`: Bottom of the figure. • `left`: Left of the figure. • `right`: Right of the figure. • `none`: No legend is displayed.
`legend.length`, `legend.width`	`numeric` | Length and width of the legend. Will adjust automatically depending on legend side.
`legend.framewidth`, `legend.tickwidth`	`numeric` | Width of the lines of the box in the legend.
`legend.framecolor`	`character` | Color of the lines of the box in the legend.
`legend.tickcolor`	`character` | Color of the ticks of the box in the legend.
`legend.type`	`character` | Type of legend to display. One of: • `normal`: Default legend displayed by **ggplot2**. • `colorbar`: Redefined colorbar legend, using guide_colorbar.
`colors.use`	`character` | Named vector of colors to use. Has to match the unique values of group.by or color.by (if used) when scale_type is set to categorical.
`font.size`	`numeric` | Overall font size of the plot. All plot elements will have a size relationship with this font size.
`font.type`	`character` | Base font family for the plot. One of: • `mono`: Mono spaced font. • `serif`: Serif font family. • `sans`: Default font family.

`axis.text.x.angle`	`numeric` | Degree to rotate the X labels. One of: 0, 45, 90.
`plot.title`, `plot.subtitle`, `plot.caption`	`character` | Title, subtitle or caption to use in the plot.
`xlab`, `ylab`	`character` | Titles for the X and Y axis.
`compute_quantiles`	`logical` | Whether to compute quantiles of the distribution and color the ridge plots by them.
`compute_custom_quantiles`	`logical` | Whether to compute custom quantiles.
`quantiles`	`numeric` | Numeric vector of quantiles.
`compute_distribution_tails`	`logical` | Whether to compute distribution tails and color them.
`prob_tails`	`numeric` | The accumulated probability that the tails should contain.
`color_by_probabilities`	`logical` | Whether to color the ridges depending on the probability.
`use_viridis`	`logical` | Whether to use viridis color scales.
`viridis.palette`	`character` | A capital letter from A to H or the scale name as in scale_fill_viridis.
`viridis.direction`	`numeric` | Either 1 or -1. Controls how the gradient of viridis scale is formed.
`sequential.palette`	`character` | Type of sequential color palette to use. Out of the sequential palettes defined in `brewer.pal`.
`sequential.direction`	`numeric` | Direction of the sequential color scale. Either 1 or -1.
`plot.grid`	`logical` | Whether to plot grid lines.
`grid.color`	`character` | Color of the grid in the plot. In heatmaps, color of the border of the cells.
`grid.type`	`character` | One of the possible linetype options: • `blank`. • `solid`. • `dashed`. • `dotted`. • `dotdash`. • `longdash`. • `twodash`.
`flip`	`logical` | Whether to invert the axis of the displayed plot.
`number.breaks`	`numeric` | Controls the number of breaks in continuous color scales of ggplot2-based plots.
`plot.title.face`, `plot.subtitle.face`, `plot.caption.face`, `axis.title.face`, `axis.text.face`, `legend.title.face`, `legend.text.face`	`character` | Controls the style of the font for the corresponding theme element. One of:

- `plain`: For normal text.
- `italic`: For text in itallic.
- `bold`: For text in bold.
- `bold.italic`: For text both in itallic and bold.

Value

A ggplot2 object.

Examples

```
# Check Suggests.
value <- SCpubr:::check_suggests(function_name = "do_RidgePlot", passive = TRUE)

if (isTRUE(value)){
  # Consult the full documentation in https://enblacar.github.io/SCpubr-book/

  # Define your Seurat object.
 sample <- readRDS(system.file("extdata/seurat_dataset_example.rds", package = "SCpubr"))

  # Compute the most basic ridge plot.
  p <- SCpubr::do_RidgePlot(sample = sample,
                            feature = "nFeature_RNA")
  p

  # Use continuous color scale.
  p <- SCpubr::do_RidgePlot(sample = sample,
                            feature = "nFeature_RNA",
                            continuous_scale = TRUE,
                            viridis.direction = 1)
  p

  # Draw quantiles of the distribution.
  p <- SCpubr::do_RidgePlot(sample = sample,
                            feature = "nFeature_RNA",
                            continuous_scale = TRUE,
                            compute_quantiles = TRUE,
                            compute_custom_quantiles = TRUE)
  p

  # Draw probability tails.
  p <- SCpubr::do_RidgePlot(sample = sample,
                            feature = "nFeature_RNA",
                            continuous_scale = TRUE,
                            compute_quantiles = TRUE,
                            compute_distribution_tails = TRUE)
  p

  # Draw probability tails.
  p <- SCpubr::do_RidgePlot(sample = sample,
                            feature = "nFeature_RNA",
                            continuous_scale = TRUE,
```

```
                              compute_quantiles = TRUE,
                              color_by_probabilities = TRUE)
    p
  } else if (base::isFALSE(value)){
    message("This function can not be used without its suggested packages.")
    message("Check out which ones are needed using `SCpubr::state_dependencies()`.")
  }
```

`do_TermEnrichmentPlot` *Display the enriched terms for a given list of genes.*

Description

Display the enriched terms for a given list of genes.

Usage

```
do_TermEnrichmentPlot(
  enriched_terms,
  nchar_wrap = 20,
  nterms = 10,
  font.size = 14,
  font.type = "sans",
  plot.title = NULL,
  plot.subtitle = NULL,
  plot.caption = NULL,
  legend.position = "bottom",
  legend.type = "colorbar",
  colors.use = NULL,
  text_labels_size = 4,
  legend.length = 30,
  legend.width = 1,
  legend.framewidth = 0.5,
  legend.tickwidth = 0.5,
  legend.framecolor = "grey50",
  legend.tickcolor = "white",
  plot.title.face = "bold",
  plot.subtitle.face = "plain",
  plot.caption.face = "italic",
  axis.title.face = "bold",
  axis.text.face = "plain",
  legend.title.face = "bold",
  legend.text.face = "plain"
)
```

Arguments

`enriched_terms`	`list` | List containing the output(s) of running Enrichr.
`nchar_wrap`	`numeric` | Number of characters to use as a limit to wrap the term names. The higher this value, the longer the lines would be for each term in the plots. Defaults to 60.
`nterms`	`numeric` | Number of terms to report for each database. Terms are arranged by adjusted p-value and selected from lowest to highest. Defaults to 5. • `Enrichr`. • `FlyEnrichr`. • `WormEnrichr`. • `YeastEnrichr`. • `FishEnrichr`.
`font.size`	`numeric` | Overall font size of the plot. All plot elements will have a size relationship with this font size.
`font.type`	`character` | Base font family for the plot. One of: • `mono`: Mono spaced font. • `serif`: Serif font family. • `sans`: Default font family.
`plot.title`, `plot.subtitle`, `plot.caption`	`character` | Title, subtitle or caption to use in the plot.
`legend.position`	`character` | Position of the legend in the plot. One of: • `top`: Top of the figure. • `bottom`: Bottom of the figure. • `left`: Left of the figure. • `right`: Right of the figure. • `none`: No legend is displayed.
`legend.type`	`character` | Type of legend to display. One of: • `normal`: Default legend displayed by **ggplot2**. • `colorbar`: Redefined colorbar legend, using guide_colorbar.
`colors.use`	`character` | Character vector of 2 colors (low and high ends of the color scale) to generate the gradient.
`text_labels_size`	`numeric` | Controls how big or small labels are in the plot.
`legend.length`, `legend.width`	`numeric` | Length and width of the legend. Will adjust automatically depending on legend side.
`legend.framewidth`, `legend.tickwidth`	`numeric` | Width of the lines of the box in the legend.
`legend.framecolor`	`character` | Color of the lines of the box in the legend.

`legend.tickcolor`
: `character` | Color of the ticks of the box in the legend.

`plot.title.face`, `plot.subtitle.face`, `plot.caption.face`, `axis.title.face`, `axis.text.face`, `legend.title.face`, `legend.text.face`
: `character` | Controls the style of the font for the corresponding theme element. One of:

  - `plain`: For normal text.
  - `italic`: For text in itallic.
  - `bold`: For text in bold.
  - `bold.italic`: For text both in itallic and bold.

Value

A ggplot2 object with enriched terms.

Examples

```
 # Check Suggests.
value <- SCpubr:::check_suggests(function_name = "do_TermEnrichmentPlot", passive = TRUE)

 if (isTRUE(value)){
   # Consult the full documentation in https://enblacar.github.io/SCpubr-book/

   # Define your enriched terms.
  enriched_terms <- readRDS(system.file("extdata/enriched_terms_example.rds", package = "SCpubr"))
   enriched_terms$GO_Cellular_Component_2021 <- NULL
   enriched_terms$Azimuth_Cell_Types_2021 <- NULL

   # Default plot.
   p <- SCpubr::do_TermEnrichmentPlot(enriched_terms = enriched_terms)
   p
 } else if (base::isFALSE(value)){
   message("This function can not be used without its suggested packages.")
   message("Check out which ones are needed using `SCpubr::state_dependencies()`.")
 }
```

`do_TFActivityPlot` *Plot TF Activities from decoupleR using Dorothea prior knowledge.*

Description

Plot TF Activities from decoupleR using Dorothea prior knowledge.

Usage

```
do_TFActivityPlot(
  sample,
  activities,
  n_tfs = 25,
  slot = "scale.data",
  statistic = "norm_wmean",
  tfs.use = NULL,
  group.by = NULL,
  split.by = NULL,
  na.value = "grey75",
  legend.position = "bottom",
  legend.width = 1,
  legend.length = 20,
  legend.framewidth = 0.5,
  legend.tickwidth = 0.5,
  legend.framecolor = "grey50",
  legend.tickcolor = "white",
  legend.type = "colorbar",
  font.size = 14,
  font.type = "sans",
  axis.text.x.angle = 45,
  enforce_symmetry = TRUE,
  diverging.palette = "RdBu",
  diverging.direction = -1,
  use_viridis = FALSE,
  viridis.palette = "G",
  viridis.direction = -1,
  sequential.palette = "YlGnBu",
  sequential.direction = 1,
  min.cutoff = NA,
  max.cutoff = NA,
  number.breaks = 5,
  flip = FALSE,
  return_object = FALSE,
  grid.color = "white",
  border.color = "black",
  plot.title.face = "bold",
  plot.subtitle.face = "plain",
  plot.caption.face = "italic",
  axis.title.face = "bold",
  axis.text.face = "plain",
  legend.title.face = "bold",
  legend.text.face = "plain"
)
```

Arguments

`sample`	`Seurat` | A Seurat object, generated by CreateSeuratObject.
`activities`	`tibble` | Result of running decoupleR method with dorothea regulon prior knowledge.
`n_tfs`	`numeric` | Number of top regulons to consider for downstream analysis.
`slot`	`character` | Data slot to use. Only one of: counts, data, scale.data. Defaults to "data".
`statistic`	`character` | DecoupleR statistic to use. One of: • `wmean`: For weighted mean. • `norm_wmean`: For normalized weighted mean. • `corr_wmean`: For corrected weighted mean.
`tfs.use`	`character` | Restrict the analysis to given regulons.
`group.by`	`character` | Metadata variable to group the output by. Has to be a character of factor column.
`split.by`	`character` | Secondary metadata variable to further group (split) the output by. Has to be a character of factor column.
`na.value`	`character` | Color value for NA.
`legend.position`	`character` | Position of the legend in the plot. One of: • `top`: Top of the figure. • `bottom`: Bottom of the figure. • `left`: Left of the figure. • `right`: Right of the figure. • `none`: No legend is displayed.
`legend.length`, `legend.width`	`numeric` | Length and width of the legend. Will adjust automatically depending on legend side.
`legend.framewidth`, `legend.tickwidth`	`numeric` | Width of the lines of the box in the legend.
`legend.framecolor`	`character` | Color of the lines of the box in the legend.
`legend.tickcolor`	`character` | Color of the ticks of the box in the legend.
`legend.type`	`character` | Type of legend to display. One of: • `normal`: Default legend displayed by **ggplot2**. • `colorbar`: Redefined colorbar legend, using guide_colorbar.
`font.size`	`numeric` | Overall font size of the plot. All plot elements will have a size relationship with this font size.
`font.type`	`character` | Base font family for the plot. One of: • `mono`: Mono spaced font. • `serif`: Serif font family.

- `sans`: Default font family.

`axis.text.x.angle`	`numeric` | Degree to rotate the X labels. One of: 0, 45, 90.
`enforce_symmetry`	`logical` | Whether the geyser and feature plot has a symmetrical color scale.
`diverging.palette`	`character` | Type of symmetrical color palette to use. Out of the diverging palettes defined in `brewer.pal`.
`diverging.direction`	`numeric` | Either 1 or -1. Direction of the divering palette. This basically flips the two ends.
`use_viridis`	`logical` | Whether to use viridis color scales.
`viridis.palette`	`character` | A capital letter from A to H or the scale name as in scale_fill_viridis.
`viridis.direction`	`numeric` | Either 1 or -1. Controls how the gradient of viridis scale is formed.
`sequential.palette`	`character` | Type of sequential color palette to use. Out of the sequential palettes defined in `brewer.pal`.
`sequential.direction`	`numeric` | Direction of the sequential color scale. Either 1 or -1.
`min.cutoff`, `max.cutoff`	`numeric` | Set the min/max ends of the color scale. Any cell/group with a value lower than min.cutoff will turn into min.cutoff and any cell with a value higher than max.cutoff will turn into max.cutoff. In FeaturePlots, provide as many values as features. Use NAs to skip a feature.
`number.breaks`	`numeric` | Controls the number of breaks in continuous color scales of ggplot2-based plots.
`flip`	`logical` | Whether to invert the axis of the displayed plot.
`return_object`	`logical` | Returns the Seurat object with the modifications performed in the function. Nomally, this contains a new assay with the data that can then be used for any other visualization desired.
`grid.color`	`character` | Color of the grid in the plot. In heatmaps, color of the border of the cells.
`border.color`	`character` | Color for the border of the heatmap body.
`plot.title.face`, `plot.subtitle.face`, `plot.caption.face`, `axis.title.face`, `axis.text.face`, `legend.title.face`, `legend.text.face`	`character` | Controls the style of the font for the corresponding theme element. One of: • `plain`: For normal text. • `italic`: For text in itallic. • `bold`: For text in bold. • `bold.italic`: For text both in itallic and bold.

Value

A ggplot2 object.

Examples

```
# Check Suggests.
value <- SCpubr:::check_suggests(function_name = "do_TFActivityPlot", passive = TRUE)

if (isTRUE(value)){
  # Consult the full documentation in https://enblacar.github.io/SCpubr-book/

  # Define your Seurat object.
  sample <- readRDS(system.file("extdata/seurat_dataset_example.rds",
                                package = "SCpubr"))

  # Define your activities object.
 dorothea_activities <- readRDS(system.file("extdata/dorothea_activities_example.rds",
                                            package = "SCpubr"))

  # General heatmap.
  out <- SCpubr::do_TFActivityPlot(sample = sample,
                                   activities = dorothea_activities)
  p <- out$heatmaps$average_scores
  p

} else if (base::isFALSE(value)){
  message("This function can not be used without its suggested packages.")
  message("Check out which ones are needed using `SCpubr::state_dependencies()`.")
}
```

---

do_ViolinPlot *Wrapper for VlnPlot.*

Description

Wrapper for VlnPlot.

Usage

```
do_ViolinPlot(
  sample,
  features,
  assay = NULL,
  slot = NULL,
  group.by = NULL,
  split.by = NULL,
  colors.use = NULL,
  pt.size = 0,
```

```
  line_width = 0.5,
  y_cut = rep(NA, length(features)),
  plot_boxplot = TRUE,
  boxplot_width = 0.2,
  legend.position = "bottom",
  plot.title = NULL,
  plot.subtitle = NULL,
  plot.caption = NULL,
  xlab = rep(NA, length(features)),
  ylab = rep(NA, length(features)),
  font.size = 14,
  font.type = "sans",
  axis.text.x.angle = 45,
  plot.grid = TRUE,
  grid.color = "grey75",
  grid.type = "dashed",
  flip = FALSE,
  ncol = NULL,
  share.y.lims = FALSE,
  legend.title = NULL,
  legend.title.position = "top",
  legend.ncol = NULL,
  legend.nrow = NULL,
  legend.byrow = FALSE,
  plot.title.face = "bold",
  plot.subtitle.face = "plain",
  plot.caption.face = "italic",
  axis.title.face = "bold",
  axis.text.face = "plain",
  legend.title.face = "bold",
  legend.text.face = "plain"
)
```

Arguments

`sample`	`Seurat` | A Seurat object, generated by CreateSeuratObject.
`features`	`character` | Features to represent.
`assay`	`character` | Assay to use. Defaults to the current assay.
`slot`	`character` | Data slot to use. Only one of: counts, data, scale.data. Defaults to "data".
`group.by`	`character` | Metadata variable to group the output by. Has to be a character of factor column.
`split.by`	`character` | Secondary metadata variable to further group (split) the output by. Has to be a character of factor column.
`colors.use`	`named_vector` | Named vector of valid color representations (either name of HEX codes) with as many named colors as unique values of group.by. If group.by is not provided, defaults to the unique values of Idents. If not provided, a color scale will be set by default.

`pt.size`	`numeric` | Size of points in the Violin plot.
`line_width`	`numeric` | Width of the lines drawn in the plot. Defaults to 1.
`y_cut`	`numeric` | Vector with the values in which the Violins should be cut. Only works for one feature.
`plot_boxplot`	`logical` | Whether to plot a Box plot inside the violin or not.
`boxplot_width`	`numeric` | Width of the boxplots. Defaults to 0.2.
`legend.position`	`character` | Position of the legend in the plot. One of: • `top`: Top of the figure. • `bottom`: Bottom of the figure. • `left`: Left of the figure. • `right`: Right of the figure. • `none`: No legend is displayed.
`plot.title`, `plot.subtitle`, `plot.caption`	`character` | Title, subtitle or caption to use in the plot.
`xlab`, `ylab`	`character` | Titles for the X and Y axis.
`font.size`	`numeric` | Overall font size of the plot. All plot elements will have a size relationship with this font size.
`font.type`	`character` | Base font family for the plot. One of: • `mono`: Mono spaced font. • `serif`: Serif font family. • `sans`: Default font family.
`axis.text.x.angle`	`numeric` | Degree to rotate the X labels. One of: 0, 45, 90.
`plot.grid`	`logical` | Whether to plot grid lines.
`grid.color`	`character` | Color of the grid in the plot. In heatmaps, color of the border of the cells.
`grid.type`	`character` | One of the possible linetype options: • `blank`. • `solid`. • `dashed`. • `dotted`. • `dotdash`. • `longdash`. • `twodash`.
`flip`	`logical` | Whether to invert the axis of the displayed plot.
`ncol`	`numeric` | Number of columns used in the arrangement of the output plot using "split.by" parameter.
`share.y.lims`	`logical` | When querying multiple features, force the Y axis of all of them to be on the same range of values (this being the max and min of all features combined).

`legend.title`	`character` | Title for the legend.
`legend.title.position`	`character` | Position for the title of the legend. One of: • `top`: Top of the legend. • `bottom`: Bottom of the legend. • `left`: Left of the legend. • `right`: Right of the legend.
`legend.ncol`	`numeric` | Number of columns in the legend.
`legend.nrow`	`numeric` | Number of rows in the legend.
`legend.byrow`	`logical` | Whether the legend is filled by row or not.
`plot.title.face`, `plot.subtitle.face`, `plot.caption.face`, `axis.title.face`, `axis.text.face`, `legend.title.face`, `legend.text.face`	`character` | Controls the style of the font for the corresponding theme element. One of: • `plain`: For normal text. • `italic`: For text in itallic. • `bold`: For text in bold. • `bold.italic`: For text both in itallic and bold.

Value

A ggplot2 object containing a Violin Plot.

Examples

```
# Check Suggests.
value <- SCpubr:::check_suggests(function_name = "do_ViolinPlot", passive = TRUE)

if (isTRUE(value)){
  # Consult the full documentation in https://enblacar.github.io/SCpubr-book/

  # Define your Seurat object.
 sample <- readRDS(system.file("extdata/seurat_dataset_example.rds", package = "SCpubr"))

  # Basic violin plot.
  p <- SCpubr::do_ViolinPlot(sample = sample,
                             feature = "nCount_RNA")
  p

  # Remove the box plots.
  p <- SCpubr::do_ViolinPlot(sample = sample,
                             feature = "nCount_RNA",
                             plot_boxplot = FALSE)
  p

} else if (base::isFALSE(value)){
  message("This function can not be used without its suggested packages.")
  message("Check out which ones are needed using `SCpubr::state_dependencies()`.")
```

```
}
```

`do_VolcanoPlot` *Compute a Volcano plot out of DE genes.*

Description

Compute a Volcano plot out of DE genes.

Usage

```
do_VolcanoPlot(
  sample,
  de_genes,
  pval_cutoff = 0.05,
  FC_cutoff = 2,
  pt.size = 2,
  border.size = 1.5,
  border.color = "black",
  font.size = 14,
  font.type = "sans",
  plot.title = NULL,
  plot.subtitle = NULL,
  plot.caption = NULL,
  plot_lines = TRUE,
  line_color = "grey75",
  line_size = 0.5,
  add_gene_tags = TRUE,
  order_tags_by = "both",
  n_genes = 5,
  use_labels = FALSE,
  colors.use = "steelblue",
  plot.title.face = "bold",
  plot.subtitle.face = "plain",
  plot.caption.face = "italic",
  axis.title.face = "bold",
  axis.text.face = "plain",
  legend.title.face = "bold",
  legend.text.face = "plain"
)
```

Arguments

`sample`	`Seurat` | A Seurat object, generated by CreateSeuratObject.
`de_genes`	`tibble` | Output of `Seurat::FindMarkers()`.
`pval_cutoff`	`numeric` | Cutoff for the p-value.

`FC_cutoff`	`numeric` | Cutoff for the avg_log2FC.
`pt.size`	`numeric` | Size of the dots.
`border.size`	`numeric` | Width of the border of the cells.
`border.color`	`character` | Color for the border of the heatmap body.
`font.size`	`numeric` | Overall font size of the plot. All plot elements will have a size relationship with this font size.
`font.type`	`character` | Base font family for the plot. One of: • `mono`: Mono spaced font. • `serif`: Serif font family. • `sans`: Default font family.
`plot.title`, `plot.subtitle`, `plot.caption`	`character` | Title, subtitle or caption to use in the plot.
`plot_lines`	`logical` | Whether to plot the division lines.
`line_color`	`character` | Color for the lines.
`line_size`	`numeric` | Size of the lines in the plot.
`add_gene_tags`	`logical` | Whether to plot the top genes.
`order_tags_by`	`character` | Either "both", "pvalue" or "logfc".
`n_genes`	`numeric` | Number of top genes in each side to plot.
`use_labels`	`logical` | Whether to use labels instead of text for the tags.
`colors.use`	`character` | Color to generate a tetradic color scale with.
`plot.title.face`, `plot.subtitle.face`, `plot.caption.face`, `axis.title.face`, `axis.text.face`, `legend.title.face`, `legend.text.face`	`character` | Controls the style of the font for the corresponding theme element. One of: • `plain`: For normal text. • `italic`: For text in itallic. • `bold`: For text in bold. • `bold.italic`: For text both in itallic and bold.

Value

A volcano plot as a ggplot2 object.

Examples

```
# Check Suggests.
value <- SCpubr::check_suggests(function_name = "do_VolcanoPlot", passive = TRUE)

if (isTRUE(value)){
  # Consult the full documentation in https://enblacar.github.io/SCpubr-book/

  # Define your Seurat object.
 sample <- readRDS(system.file("extdata/seurat_dataset_example.rds", package = "SCpubr"))
```

```
    # Retrieve DE genes.
    de_genes <- readRDS(system.file("extdata/de_genes_example.rds", package = "SCpubr"))

    # Generate a volcano plot.
    p <- SCpubr::do_VolcanoPlot(sample = sample,
                                de_genes = de_genes)
    p

  } else if (base::isFALSE(value)){
    message("This function can not be used without its suggested packages.")
    message("Check out which ones are needed using `SCpubr::state_dependencies()`.")
  }
```

`human_chr_locations` *Chromosome arm locations for human genome GRCh38.*

Description

A tibble containing the chromosome, arm and start and end coordinates.

Usage

```
data(human_chr_locations)
```

Format

A tibble with 48 rows and 4 columns:

chr Chromosome.

arm Chromosome arm.

start Start coordinates.

end End coordinates.

`package_report` *Generate a status report of SCpubr and its dependencies.*

Description

This function generates a summary report of the installation status of SCpubr, which packages are still missing and which functions can or can not currently be used.

Usage

```
package_report(startup = FALSE, extended = FALSE)
```

Arguments

`startup`	`logical` | Whether the message should be displayed at startup, therefore, also containing welcoming messages and tips. If `FALSE`, only the report itself will be printed.
`extended`	`logical` | Whether the message should also include installed packages, current and available version, and which `SCpubr` functions can be used with the currently installed packages.

Value

None

Examples

```
# Print a package report.
SCpubr::package_report(startup = FALSE, extended = FALSE)
```

Index