

Package ‘Giotto’

November 7, 2019

Title Spatial single-cell transcriptomics pipeline.

Version 0.1.2

Description Pipeline to process, analyze and visualize (spatial) single-cell transcriptomic data.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

Depends data.table (>= 1.12.2),
ggplot2 (>= 3.1.1),
base (>= 3.5.1),
utils (>= 3.5.1),
R (>= 3.5.1)

Imports Rtsne (>= 0.15),
uwot (>= 0.0.0.9010),
multinet (>= 3.0.2),
FactoMineR (>= 1.34),
factoextra (>= 1.0.5),
cowplot (>= 0.9.4),
grDevices,
RColorBrewer,
jackstraw (>= 1.3),
dbscan (>= 1.1-3),
ggalluvial (>= 0.9.1),
scales (>= 1.0.0),
ComplexHeatmap (>= 1.20.0),
qvalue (>= 2.14.1),
lfa (>= 1.12.0),
igraph (>= 1.2.4.1),
plotly,
reticulate,
magrittr,
limma

Suggests knitr,
rmarkdown,
MAST,
scrna (>= 1.10.1)

biocViews

VignetteBuilder knitr

R topics documented:

addCellMetadata	5
addCellStatistics	6
addGeneMetadata	6
addGeneStatistics	7
addHMRF	8
addNetworkLayout	8
addStatistics	9
adjustGiottoMatrix	10
allCellCellcommunicationsScores	11
annotateGiotto	12
annotateSpatialNetwork	12
annotate_spatlocs_with_spatgrid_2D	13
annotate_spatlocs_with_spatgrid_3D	13
average_gene_gene_expression_in_groups	14
binGetSpatialGenes	15
calculateHVG	16
calculateMetaTable	17
calculateSpatialGenes	17
calculate_spatial_genes_python	19
cellProximityBarplot	20
cellProximityEnrichment	20
cellProximityHeatmap	21
cellProximityNetwork	22
cellProximityVisPlot	23
cellProximityVisPlot_2D_ggplot	24
cellProximityVisPlot_2D_plotly	26
cellProximityVisPlot_3D_plotly	27
clusterCells	28
createGiottoObject	30
createHeatmap_DT	31
createNearestNetwork	32
createSpatialGrid	33
createSpatialGrid_2D	34
createSpatialGrid_3D	35
createSpatialNetwork	36
create_average_detection_DT	37
create_average_DT	37
create_cell_type_random_cell_IDs	38
create_cluster_matrix	38
create_dimObject	39
decide_cluster_order	39
detectSpatialPatterns	40
direction_test_CPG	41
doHclust	41
doHMRF	42
doKmeans	43
doLeidenCluster	44
doLeidenSubCluster	45
doLouvainCluster	47
doLouvainCluster_community	48

doLouvainCluster_multinet	49
doLouvainSubCluster	50
doLouvainSubCluster_community	51
doLouvainSubCluster_multinet	52
doRandomWalkCluster	54
doSNNCluster	55
dt_to_matrix	56
exportGiottoViewer	56
exprOnlyCellCellcommunicationScores	57
extended_gini_fun	58
extractNearestNetwork	58
fDataDT	59
filterCombinations	59
filterDistributions	60
filterGiotto	61
findGiniMarkers	62
findGiniMarkers_one_vs_all	63
findMarkers	64
findMarkers_one_vs_all	65
findMastMarkers	66
findMastMarkers_one_vs_all	67
findScranMarkers	68
findScranMarkers_one_vs_all	68
find_grid_2D	69
find_grid_3D	70
find_grid_x	70
find_grid_y	70
find_grid_z	70
fish_function	71
fish_function2	71
FSV_show	71
GenePattern_show	72
getCellProximityGeneScores	73
getClusterSimilarity	74
getDendrogramSplits	74
getDistinctColors	75
getGeneToGeneSelection	76
get_cell_to_cell_sorted_name_conversion	77
get_interaction_gene_enrichment	77
get_specific_interaction_gene_enrichment	78
giotto-class	78
iterCluster	79
iterLeidenCluster	80
iterLouvainCluster	81
iterLouvainCluster_community	82
iterLouvainCluster_multinet	83
kmeans_binarize	85
loadHMRP	85
make_simulated_network	86
mergeClusters	86
mygini_fun	87
nnDT_to_kNN	87

node_clusters	88
normalizeGiotto	88
OR_function2	89
pDataDT	89
plotCPGscores	90
plotGTGscores	91
plotHeatmap	92
plotly_axis_scale_2D	93
plotly_axis_scale_3D	94
plotly_grid	94
plotly_network	95
plotMetaDataHeatmap	95
plotPCA	97
plotTSNE	97
plotUMAP	98
plot_network_layer_ggplot	99
plot_point_layer_ggplot	99
print.giotto	101
rank_binarize	101
runPCA	102
runTSNE	103
runUMAP	104
selectPatternGenes	105
select_expression_values	106
show,giotto-method	106
showClusterDendrogram	106
showClusterHeatmap	107
showCPGscores	108
showGeneExpressionProximityScore	109
showGTGscores	109
showIntExpressionProximityScore	110
showPattern	111
showPatternGenes	112
showProcessingSteps	113
showTopGeneToGene	113
signPCA	114
Spatial_AEH	115
Spatial_DE	116
specificCellCellcommunicationScores	116
split_dendrogram_in_two	117
stitchFieldCoordinates	118
subClusterCells	119
subsetGiotto	120
viewHMRResults	121
violinPlot	121
visDimGenePlot	122
visDimGenePlot_2D_ggplot	124
visDimGenePlot_3D_plotly	125
visDimPlot	126
visDimPlot_2D_ggplot	128
visDimPlot_2D_plotly	129
visDimPlot_3D_plotly	131

visForceLayoutPlot	132
visGenePlot	133
visGenePlot_2D_ggplot	134
visGenePlot_3D_plotly	136
visPlot	137
visPlot_2D_ggplot	139
visPlot_2D_plotly	140
visPlot_3D_plotly	141
visSpatDimGenePlot	143
visSpatDimGenePlot_2D	145
visSpatDimGenePlot_3D	147
visSpatDimPlot	149
visSpatDimPlot_2D	151
visSpatDimPlot_3D	152
write_giotto_viewer_annotation	154
write_giotto_viewer_dim_reduction	155
Index	156

addCellMetadata	<i>addCellMetadata</i>
-----------------	------------------------

Description

adds cell metadata to the giotto object

Usage

```
addCellMetadata(gobject, new_metadata, by_column = F,  
column_cell_ID = NULL)
```

Arguments

- gobject giotto object
- new_metadata new metadata to use
- by_column merge metadata based on cell_ID column in pDataDT
- column_cell_ID column name of new metadata to use if by_column = TRUE

Details

Description of how to add cell metadata ...

Value

giotto object

Examples

```
addCellMetadata(gobject)
```

addCellStatistics	<i>addCellStatistics</i>
-------------------	--------------------------

Description

adds cells statistics to the giotto object

Usage

```
addCellStatistics(gobject, expression_values = c("normalized", "scaled",
"custom"), detection_threshold = 0, return_gobject = TRUE)
```

Arguments

gobject	giotto object
expression_values	expression values to use
detection_threshold	detection threshold to consider a gene detected
return_gobject	boolean: return giotto object (default = TRUE)

Details

Details about cell statistics that are returned.

Value

giotto object if return_gobject = TRUE

Examples

```
addCellStatistics(gobject)
```

addGeneMetadata	<i>addGeneMetadata</i>
-----------------	------------------------

Description

adds gene metadata to the giotto object

Usage

```
addGeneMetadata(gobject, new_metadata, by_column = F,
column_gene_ID = NULL)
```

Arguments

gobject	giotto object
new_metadata	new metadata to use
by_column	merge metadata based on gene_ID column in fDataDT
column_cell_ID	column name of new metadata to use if by_column = TRUE

Details

Description of how to add gene metadata ...

Value

giotto object

Examples

```
addGeneMetadata(gobject)
```

addGeneStatistics

addGeneStatistics

Description

adds gene statistics to the giotto object

Usage

```
addGeneStatistics(gobject, expression_values = c("normalized", "scaled",  
"custom"), detection_threshold = 0, return_gobject = TRUE)
```

Arguments

<code>gobject</code>	giotto object
<code>expression_values</code>	expression values to use
<code>detection_threshold</code>	detection threshold to consider a gene detected
<code>return_gobject</code>	boolean: return giotto object (default = TRUE)

Details

Details about gene statistics that are returned.

Value

giotto object if `return_gobject = TRUE`

Examples

```
addGeneStatistics(gobject)
```

addHMRF

addHMRF

Description

Add selected results from doHMRF to the giotto object

Usage

```
addHMRF(gobject, HMRFoutput, k = NULL, betas_to_add = NULL,
        hmr_name = NULL)
```

Arguments

gobject	giotto object
HMRFoutput	HMRF output from doHMRF()
k	number of domains
betas_to_add	results from different betas that you want to add
name	specify a custom name

Details

Description ...

Value

giotto object

Examples

```
addHMRF(gobject)
```

addNetworkLayout

addNetworkLayout

Description

Add a network layout for a select nearest neighbor network

Usage

```
addNetworkLayout(gobject, nn_network_to_use = NULL,
                 network_name = NULL, layout_type = c("drl"), options_list = NULL,
                 layout_name = "layout", return_gobject = TRUE)
```


Arguments

<code>gobject</code>	giotto object
<code>nn_network_to_use</code>	kNN or sNN
<code>network_name</code>	name of NN network to be used
<code>layout_type</code>	layout algorithm to use
<code>options_list</code>	list of options for selected layout
<code>layout_name</code>	name for layout
<code>return_gobject</code>	boolean: return giotto object (default = TRUE)

Details

Description of layouts and options.

Value

giotto object with updated layout for selected NN network

Examples

```
addNetworkLayout(gobject)
```

<code>addStatistics</code>	<i>addStatistics</i>
----------------------------	----------------------

Description

adds genes and cells statistics to the giotto object

Usage

```
addStatistics(gobject, expression_values = c("normalized", "scaled",
      "custom"), detection_threshold = 0, return_gobject = TRUE)
```

Arguments

<code>gobject</code>	giotto object
<code>expression_values</code>	expression values to use
<code>detection_threshold</code>	detection threshold to consider a gene detected
<code>return_gobject</code>	boolean: return giotto object (default = TRUE)

Details

Details about gene and cell statistics that are returned.

Value

giotto object if `return_gobject = TRUE`, else a list with results

Examples

```
addStatistics(gobject)
```

adjustGiottoMatrix	<i>adjustGiottoMatrix</i>
--------------------	---------------------------

Description

normalize and/or scale expresion values of Giotto object

Usage

```
adjustGiottoMatrix(gobject, expression_values = c("normalized", "scaled",
"custom"), batch_columns = NULL, covariate_columns = NULL,
return_gobject = TRUE, update_slot = c("custom"))
```

Arguments

gobject	giotto object
expression_values	expression values to use
batch_columns	metadata columns that represent different batch
covariate_columns	metadata columns that represent covariates to regress out
return_gobject	boolean: return giotto object (default = TRUE)
update_slot	expression slot that will be updated (default = custom)

Details

Description of adjusting steps ...

Value

giotto object

Examples

```
adjustGiottoMatrix(gobject)
```

```
allCellCellcommunicationsScores  
    allCellCellcommunicationsScores
```

Description

All Cell-Cell communication scores based on spatial expression of interacting cells

Usage

```
allCellCellcommunicationsScores(gobject,  
    spatial_network_name = "spatial_network",  
    cluster_column = "cell_types", random_iter = 100, gene_set_1,  
    gene_set_2, log2FC_addendum = 0.1, min_observations = 2,  
    verbose = c("a little", "a lot", "none"))
```

Arguments

<code>gobject</code>	giotto object to use
<code>spatial_network_name</code>	spatial network to use for identifying interacting cells
<code>cluster_column</code>	cluster column with cell type information
<code>random_iter</code>	number of iterations
<code>gene_set_1</code>	first specific gene set from gene pairs
<code>gene_set_2</code>	second specific gene set from gene pairs
<code>log2FC_addendum</code>	addendum to add when calculating log2FC
<code>min_observations</code>	minimum number of interactions needed to be considered
<code>verbose</code>	verbose

Details

Details will follow.

Value

Cell-Cell communication scores for gene pairs based on spatial interaction

Examples

```
allCellCellcommunicationsScores(gobject)
```

annotateGiotto	<i>annotateGiotto</i>
----------------	-----------------------

Description

adds cell annotation to giotto object based on clustering

Usage

```
annotateGiotto(gobject, annotation_vector = NULL,
               cluster_column = NULL, name = "cell_types")
```

Arguments

gobject	giotto object
annotation_vector	named annotation vector (names = cluster ids)
cluster_column	cluster column to convert to annotation names
name	new name for annotation column

Details

Description of how to add cell metadata ...

Value

giotto object

Examples

```
annotateGiotto(gobject)
```

annotateSpatialNetwork	<i>annotateSpatialNetwork</i>
------------------------	-------------------------------

Description

Annotate spatial network with cell metadata information.

Usage

```
annotateSpatialNetwork(gobject, spatial_network_name = "spatial_network",
                       cluster_column)
```

Arguments

gobject	giotto object
spatial_network_name	name of spatial network to use
cluster_column	name of column to use for clusters

Value

annotated network in data.table format

Examples

```
annotateSpatialNetwork(gobject)
```

```
annotate_spatlocs_with_spatgrid_2D
```

```
annotate_spatlocs_with_spatgrid_2D
```

Description

annotate spatial locations with 2D spatial grid information

Usage

```
annotate_spatlocs_with_spatgrid_2D(spatloc, spatgrid)
```

Arguments

spatloc	spatial_locs slot from giotto object
spatgrid	selected spatial_grid slot from giotto object

Value

annotated spatial location data.table

Examples

```
annotate_spatlocs_with_spatgrid_2D()
```

```
annotate_spatlocs_with_spatgrid_3D
```

```
annotate_spatlocs_with_spatgrid_3D
```

Description

annotate spatial locations with 3D spatial grid information

Usage

```
annotate_spatlocs_with_spatgrid_3D(spatloc, spatgrid)
```

Arguments

spatloc	spatial_locs slot from giotto object
spatgrid	selected spatial_grid slot from giotto object

Value

annotated spatial location data.table

Examples

```
annotate_spatlocs_with_spatgrid_3D()
```

```
average_gene_gene_expression_in_groups
```

```
average_gene_gene_expression_in_groups
```

Description

calculate average expression per cluster

Usage

```
average_gene_gene_expression_in_groups(gobject,  
  cluster_column = "cell_types", gene_set_1, gene_set_2)
```

Arguments

<code>gobject</code>	giotto object to use
<code>cluster_column</code>	cluster column with cell type information
<code>gene_set_1</code>	first specific gene set from gene pairs
<code>gene_set_2</code>	second specific gene set from gene pairs

Details

Details will follow.

Value

data.table with average expression scores for each cluster

Examples

```
average_gene_gene_expression_in_groups(gobject)
```

binGetSpatialGenes	<i>binGetSpatialGenes</i>
--------------------	---------------------------

Description

compute genes that are spatially clustered

Usage

```
binGetSpatialGenes(gobject, bin_method = c("kmeans", "rank"),
  expression_values = c("normalized", "scaled", "custom"),
  spatial_network_name = "spatial_network", nstart = 3,
  iter_max = 10, percentage_rank = 10, do_fisher_test = F,
  community_expectation = 5, verbose = F)
```

Arguments

<code>gobject</code>	giotto object
<code>bin_method</code>	method to binarize gene expression
<code>expression_values</code>	expression values to use
<code>spatial_network_name</code>	name of spatial network to use (default = 'spatial_network')
<code>nstart</code>	kmeans: nstart parameter
<code>iter_max</code>	kmeans: iter.max parameter
<code>do_fisher_test</code>	perform fisher test
<code>community_expectation</code>	cell degree expectation in spatial communities
<code>verbose</code>	be verbose
<code>rank_percentage</code>	percentage of top cells for binarization

Details

Description of how we compute spatial genes.

Value

giotto object spatial genes appended to fDataDT

Examples

```
binGetSpatialGenes(gobject)
```

calculateHVG

calculateHVG

Description

compute highly variable genes

Usage

```
calculateHVG(gobject, expression_values = c("normalized", "scaled",
      "custom"), method = c("cov_loess", "cov_groups", "gini_loess"),
      reverse_log_scale = T, logbase = 2, expression_threshold = 0,
      nr_expression_groups = 20, zscore_threshold = 1.5, HVGname = "hvg",
      difference_in_variance = 1, show_plot = T, return_gobject = T)
```

Arguments

gobject	giotto object
expression_values	expression values to use
method	method to calculate highly variable genes
reverse_log_scale	reverse log-scale of expression values
logbase	if reverse_log_scale is TRUE, which log base was used?
expression_threshold	expression threshold to consider a gene detected
nr_expression_groups	number of expression groups for cov_groups
zscore_threshold	zscore to select hvg for cov_groups
HVGname	name for highly variable genes in cell metadata
difference_in_variance	minimum difference in variance required
show_plot	show plots
return_gobject	boolean: return giotto object (default = TRUE)

Details

Description of how we compute highly variable genes.

Value

giotto object highly variable genes appended to gene metadata (fDataDT)

Examples

```
calculateHVG(gobject)
```

calculateMetaTable	<i>calculateMetaTable</i>
--------------------	---------------------------

Description

calculates the average gene expression for one or more (combined) annotation columns.

Usage

```
calculateMetaTable(gobject, expression_values = c("normalized", "scaled",
  "custom"), metadata_cols = NULL, selected_genes = NULL)
```

Arguments

gobject	giotto object
expression_values	expression values to use
metadata_cols	annotation columns found in pDataDT(gobject)
selected_genes	subset of genes to use

Value

data.table with average expression values for each gene per (combined) annotation

Examples

```
calculateMetaTable(gobject)
```

calculateSpatialGenes	<i>calculateSpatialGenes</i>
-----------------------	------------------------------

Description

compute genes that are spatially clustered

Usage

```
calculateSpatialGenes(gobject, expression_values = c("normalized",
  "scaled", "custom"), method = c("kmeans", "gini", "rank"),
  spatial_network_name = "spatial_network", simulations = 10,
  detection_threshold = 0, loess_span = 0.2, pred_difference = 0.01,
  split_gene_groups = 10, show_plot = T, rank_percentage = 10,
  pvalue = 0.01, OddsRatio = 2, min_N = 20, max_N = 5000,
  SVname = "SV", show_genes = T, nr_genes = 20, return_gobject = T)
```

Arguments

<code>gobject</code>	giotto object
<code>expression_values</code>	expression values to use
<code>method</code>	method to calculate spatial genes
<code>spatial_network_name</code>	name of spatial network to use (default = 'spatial_network')
<code>detection_threshold</code>	detection threshold to consider a gene detected
<code>loess_span</code>	loess span for loess regression
<code>pred_difference</code>	minimum difference between observed and predicted
<code>split_gene_groups</code>	number of groups to split genes in
<code>show_plot</code>	show plots
<code>rank_percentage</code>	percentage of top cells for binarization
<code>pvalue</code>	minimum p-value
<code>OddsRatio</code>	minimum odds ratio
<code>min_N</code>	minimum number of cells that need to display high expression upon binarization
<code>max_N</code>	maximum number of cells that can display high expression upon binarization
<code>SVname</code>	name for identified spatial genes (default = 'SV')
<code>show_genes</code>	show top genes on plot
<code>nr_genes</code>	# of genes to plot if <code>show_genes = TRUE</code>
<code>return_gobject</code>	boolean: return giotto object (default = TRUE)

Details

Description of how we compute spatial genes.

Value

giotto object spatial genes appended to fDataDT

Examples

```
calculateSpatialGenes(gobject)
```

```
calculate_spatial_genes_python  
    calculate_spatial_genes_python
```

Description

Calculate spatial genes using distance matrix.

Usage

```
calculate_spatial_genes_python(gobject,  
    expression_values = c("normalized", "scaled", "custom"),  
    metric = "euclidean", subset_genes = NULL, rbp_p = 0.95,  
    examine_top = 0.3, python_path = NULL)
```

Arguments

<code>gobject</code>	giotto object
<code>expression_values</code>	expression values to use
<code>metric</code>	distance metric to use
<code>subset_genes</code>	only run on this subset of genes
<code>rbp_p</code>	fractional binarization threshold
<code>examine_top</code>	top fraction to evaluate with silhouette
<code>python_path</code>	specify specific path to python if required

Details

Description of how we compute spatial pattern genes.

Value

data.table with spatial scores

Examples

```
calculate_spatial_genes_python(gobject)
```

```
cellProximityBarplot  cellProximityBarplot
```

Description

Create barplot from cell-cell proximity scores

Usage

```
cellProximityBarplot(CPscore, min_orig_ints = 5, min_sim_ints = 5,
  p_val = 0.05)
```

Arguments

CPscore	CPscore, output from cellProximityEnrichment()
min_orig_ints	filter on minimum original cell-cell interactions
min_sim_ints	filter on minimum simulated cell-cell interactions
p_val	p-value

Details

This function creates a barplot that shows the spatial proximity enrichment or depletion of cell type pairs.

Value

ggplot barplot

Examples

```
cellProximityBarplot(CPscore)
```

```
cellProximityEnrichment
  cellProximityEnrichment
```

Description

Compute cell-cell interaction enrichment (observed vs expected)

Usage

```
cellProximityEnrichment(gobject,
  spatial_network_name = "spatial_network", cluster_column,
  number_of_simulations = 100)
```

Arguments

`gobject` giotto object
`spatial_network_name` name of spatial network to use
`cluster_column` name of column to use for clusters
`number_of_simulations` number of simulations to create expected observations

Details

Spatial proximity enrichment or depletion between pairs of cell types is calculated by calculating the observed over the expected frequency of cell-cell proximity interactions. The expected frequency is the average frequency calculated from a number of spatial network simulations. Each individual simulation is obtained by random permutations of the cell type labels of each node (cell) in the spatial network.

Value

List of cell Proximity scores (CPscores) in data.table format. The first data.table (`raw_sim_table`) shows the raw observations of both the original and simulated networks. The second data.table (`enrichm_res`) shows the enrichment results.

Examples

```
cellProximityEnrichment(gobject)
```

```
cellProximityHeatmap    cellProximityHeatmap
```

Description

Create heatmap from cell-cell proximity scores

Usage

```
cellProximityHeatmap(CPscore, scale = T, order_cell_types = T,  
  color_breaks = NULL, color_names = NULL)
```

Arguments

`CPscore` CPscore, output from `cellProximityEnrichment()`
`scale` scale cell-cell proximity interaction scores
`order_cell_types` order cell types based on enrichment correlation
`color_breaks` numerical vector of length 3 to represent min, mean and maximum
`color_names` character color vector of length 3

Details

This function creates a heatmap that shows the spatial proximity enrichment or depletion of cell type pairs.

Value

ggplot heatmap

Examples

```
cellProximityHeatmap(CPscore)
```

```
cellProximityNetwork  cellProximityNetwork
```

Description

Create network from cell-cell proximity scores

Usage

```
cellProximityNetwork(CPscore, remove_self_edges = FALSE,
  color_depletion = "blue", color_enrichment = "red",
  rescale_edge_weights = TRUE, edge_weight_range_depletion = c(0.1, 1),
  edge_weight_range_enrichment = c(1, 5), layout = "Fruchterman")
```

Arguments

CPscore	CPscore, output from cellProximityEnrichment()
remove_self_edges	remove enrichment/depletion edges with itself
color_depletion	color for depleted cell-cell interactions
color_enrichment	color for enriched cell-cell interactions
rescale_edge_weights	rescale edge weights (boolean)
edge_weight_range_depletion	numerical vector of length 2 to rescale depleted edge weights
edge_weight_range_enrichment	numerical vector of length 2 to rescale enriched edge weights
layout	layout algorithm to use to draw nodes and edges

Details

This function creates a network that shows the spatial proximity enrichment or depletion of cell type pairs.

Value

igraph plot

Examples

```
cellProximityNetwork(CPscore)
```

cellProximityVisPlot *cellProximityVisPlot*

Description

Visualize cell-cell interactions according to spatial coordinates

Usage

```
cellProximityVisPlot(gobject, interaction_name = NULL,
  cluster_column = NULL, sdimx = NULL, sdimy = NULL, sdimz = NULL,
  cell_color = NULL, cell_color_code = NULL, color_as_factor = T,
  show_other_cells = F, show_network = F, show_other_network = F,
  network_color = NULL, spatial_network_name = "spatial_network",
  show_grid = F, grid_color = NULL,
  spatial_grid_name = "spatial_grid", coord_fix_ratio = 1,
  show_legend = T, point_size_select = 2,
  point_select_border_col = "black", point_select_border_stroke = 0.05,
  point_size_other = 1, point_alpha_other = 0.3,
  point_other_border_col = "lightgrey",
  point_other_border_stroke = 0.01, axis_scale = c("cube", "real",
  "custom"), custom_ratio = NULL, x_ticks = NULL, y_ticks = NULL,
  z_ticks = NULL, plot_method = c("ggplot", "plotly"), ...)
```

Arguments

<code>gobject</code>	giotto object
<code>interaction_name</code>	cell-cell interaction name
<code>cluster_column</code>	cluster column with cell clusters
<code>sdimx</code>	x-axis dimension name (default = 'sdimx')
<code>sdimy</code>	y-axis dimension name (default = 'sdimy')
<code>sdimz</code>	z-axis dimension name (default = 'sdimz')
<code>cell_color</code>	color for cells (see details)
<code>cell_color_code</code>	named vector with colors
<code>color_as_factor</code>	convert color column to factor
<code>show_network</code>	show underlying spatial network
<code>network_color</code>	color of spatial network
<code>spatial_network_name</code>	name of spatial network to use
<code>show_grid</code>	show spatial grid
<code>grid_color</code>	color of spatial grid
<code>spatial_grid_name</code>	name of spatial grid to use

coord_fix_ratio	fix ratio between x and y-axis
show_legend	show legend
point_size_select	size of selected points
point_select_border_col	border color of selected points
point_select_border_stroke	stroke size of selected points
point_size_other	size of other points
point_other_border_col	border color of other points
point_other_border_stroke	stroke size of other points

Details

Description of parameters.

Value

ggplot or plotly

Examples

```
cellProximityVisPlot(gobject)
```

```
cellProximityVisPlot_2D_ggplot
  cellProximityVisPlot_2D_ggplot
```

Description

Visualize 2D cell-cell interactions according to spatial coordinates in ggplot mode

Usage

```
cellProximityVisPlot_2D_ggplot(gobject, interaction_name = NULL,
  cluster_column = NULL, sdimx = NULL, sdimy = NULL,
  cell_color = NULL, cell_color_code = NULL, color_as_factor = T,
  show_other_cells = F, show_network = F, show_other_network = F,
  network_color = NULL, spatial_network_name = "spatial_network",
  show_grid = F, grid_color = NULL,
  spatial_grid_name = "spatial_grid", coord_fix_ratio = 1,
  show_legend = T, point_size_select = 2,
  point_select_border_col = "black", point_select_border_stroke = 0.05,
  point_size_other = 1, point_alpha_other = 0.3,
  point_other_border_col = "lightgrey",
  point_other_border_stroke = 0.01, ...)
```


Arguments

<code>gobject</code>	giotto object
<code>interaction_name</code>	cell-cell interaction name
<code>cluster_column</code>	cluster column with cell clusters
<code>sdimx</code>	x-axis dimension name (default = 'sdimx')
<code>sdimy</code>	y-axis dimension name (default = 'sdimy')
<code>cell_color</code>	color for cells (see details)
<code>cell_color_code</code>	named vector with colors
<code>color_as_factor</code>	convert color column to factor
<code>show_other_cells</code>	decide if show cells not in network
<code>show_network</code>	show underlying spatial network
<code>network_color</code>	color of spatial network
<code>spatial_network_name</code>	name of spatial network to use
<code>show_grid</code>	show spatial grid
<code>grid_color</code>	color of spatial grid
<code>spatial_grid_name</code>	name of spatial grid to use
<code>coord_fix_ratio</code>	fix ratio between x and y-axis
<code>show_legend</code>	show legend
<code>point_size_select</code>	size of selected points
<code>point_select_border_col</code>	border color of selected points
<code>point_select_border_stroke</code>	stroke size of selected points
<code>point_size_other</code>	size of other points
<code>point_other_border_col</code>	border color of other points
<code>point_other_border_stroke</code>	stroke size of other points

Details

Description of parameters.

Value

ggplot

Examples

```
cellProximityVisPlot_2D_ggplot(gobject)
```

```
cellProximityVisPlot_2D_plotly
      cellProximityVisPlot_2D_plotly
```

Description

Visualize 2D cell-cell interactions according to spatial coordinates in plotly mode

Usage

```
cellProximityVisPlot_2D_plotly(gobject, interaction_name = NULL,
  cluster_column = NULL, sdimx = NULL, sdimy = NULL,
  cell_color = NULL, cell_color_code = NULL, color_as_factor = T,
  show_other_cells = F, show_network = F, show_other_network = F,
  network_color = NULL, spatial_network_name = "spatial_network",
  show_grid = F, grid_color = NULL,
  spatial_grid_name = "spatial_grid", show_legend = T,
  point_size_select = 2, point_size_other = 1,
  point_alpha_other = 0.3, axis_scale = c("cube", "real", "custom"),
  custom_ratio = NULL, x_ticks = NULL, y_ticks = NULL, ...)
```

Arguments

<code>gobject</code>	giotto object
<code>interaction_name</code>	cell-cell interaction name
<code>cluster_column</code>	cluster column with cell clusters
<code>sdimx</code>	x-axis dimension name (default = 'sdimx')
<code>sdimy</code>	y-axis dimension name (default = 'sdimy')
<code>cell_color</code>	color for cells (see details)
<code>cell_color_code</code>	named vector with colors
<code>color_as_factor</code>	convert color column to factor
<code>show_other_cells</code>	decide if show cells not in network
<code>show_network</code>	show underlying spatial network
<code>network_color</code>	color of spatial network
<code>spatial_network_name</code>	name of spatial network to use
<code>show_grid</code>	show spatial grid
<code>grid_color</code>	color of spatial grid
<code>spatial_grid_name</code>	name of spatial grid to use
<code>show_legend</code>	show legend
<code>point_size_select</code>	size of selected points
<code>coord_fix_ratio</code>	fix ratio between x and y-axis

Details

Description of parameters.

Value

plotly

Examples

```
cellProximityVisPlot_2D_plotly(gobject)
```

```
cellProximityVisPlot_3D_plotly
```

```
cellProximityVisPlot_3D_plotly
```

Description

Visualize 3D cell-cell interactions according to spatial coordinates in plotly mode

Usage

```
cellProximityVisPlot_3D_plotly(gobject, interaction_name = NULL,
  cluster_column = NULL, sdimx = NULL, sdimy = NULL, sdimz = NULL,
  cell_color = NULL, cell_color_code = NULL, color_as_factor = T,
  show_other_cells = F, show_network = F, show_other_network = F,
  network_color = NULL, spatial_network_name = "spatial_network",
  show_grid = F, grid_color = NULL,
  spatial_grid_name = "spatial_grid", show_legend = T,
  point_size_select = 2, point_size_other = 1,
  point_alpha_other = 0.5, axis_scale = c("cube", "real", "custom"),
  custom_ratio = NULL, x_ticks = NULL, y_ticks = NULL,
  z_ticks = NULL, ...)
```

Arguments

<code>gobject</code>	giotto object
<code>interaction_name</code>	cell-cell interaction name
<code>cluster_column</code>	cluster column with cell clusters
<code>sdimx</code>	x-axis dimension name (default = 'sdimx')
<code>sdimy</code>	y-axis dimension name (default = 'sdimy')
<code>sdimz</code>	z-axis dimension name (default = 'sdimz')
<code>cell_color</code>	color for cells (see details)
<code>cell_color_code</code>	named vector with colors
<code>color_as_factor</code>	convert color column to factor
<code>show_other_cells</code>	decide if show cells not in network

show_network show underlying spatial network
network_color color of spatial network
spatial_network_name
 name of spatial network to use
show_grid show spatial grid
grid_color color of spatial grid
spatial_grid_name
 name of spatial grid to use
show_legend show legend
point_size_select
 size of selected points
coord_fix_ratio
 fix ratio between x and y-axis

Details

Description of parameters.

Value

plotly

Examples

cellProximityVisPlot_3D_plotly(gobject)

clusterCells	<i>clusterCells</i>
--------------	---------------------

Description

cluster cells using a NN-network and community detection algorithms

Usage

```
clusterCells(gobject, cluster_method = c("leiden", "louvain_community",
"louvain_multinet", "randomwalk", "sNNclust", "kmeans", "hierarchical"),
name = "cluster_name", nn_network_to_use = "sNN",
network_name = "sNN.pca", pyth_leid_resolution = 1,
pyth_leid_weight_col = "weight",
pyth_leid_part_type = c("RBConfigurationVertexPartition",
"ModularityVertexPartition"), pyth_leid_init_memb = NULL,
pyth_leid_iterations = 1000, pyth_louv_resolution = 1,
pyth_louv_weight_col = NULL, python_louv_random = F,
python_path = NULL, louvain_gamma = 1, louvain_omega = 1,
walk_steps = 4, walk_clusters = 10, walk_weights = NA,
sNNclust_k = 20, sNNclust_eps = 4, sNNclust_minPts = 16,
borderPoints = TRUE, expression_values = c("normalized", "scaled",
"custom"), genes_to_use = NULL, dim_reduction_to_use = c("cells",
"pca", "umap", "tsne"), dim_reduction_name = "pca",
```

```

dimensions_to_use = 1:10, distance_method = c("original", "pearson",
"spearman", "euclidean", "maximum", "manhattan", "canberra", "binary",
"minkowski"), km_centers = 10, km_iter_max = 100, km_nstart = 1000,
km_algorithm = "Hartigan-Wong",
hc_agglomeration_method = c("ward.D2", "ward.D", "single", "complete",
"average", "mcquitty", "median", "centroid"), hc_k = 10, hc_h = NULL,
return_gobject = TRUE, set_seed = T, seed_number = 1234, ...)

```

Arguments

<code>gobject</code>	giotto object
<code>cluster_method</code>	community cluster method to use
<code>name</code>	name for new clustering result
<code>nn_network_to_use</code>	type of NN network to use (kNN vs sNN)
<code>network_name</code>	name of NN network to use
<code>pyth_leid_resolution</code>	resolution for leiden
<code>pyth_leid_weight_col</code>	column to use for weights
<code>pyth_leid_part_type</code>	partition type to use
<code>pyth_leid_init_memb</code>	initial membership
<code>pyth_leid_iterations</code>	number of iterations
<code>pyth_louv_resolution</code>	resolution for louvain
<code>pyth_louv_weight_col</code>	python louvain param: weight column
<code>python_louv_random</code>	python louvain param: random
<code>python_path</code>	specify specific path to python if required
<code>louvain_gamma</code>	louvain param: gamma or resolution
<code>louvain_omega</code>	louvain param: omega
<code>walk_steps</code>	randomwalk: number of steps
<code>walk_clusters</code>	randomwalk: number of clusters
<code>walk_weights</code>	randomwalk: weight column
<code>sNNclust_k</code>	SNNclust: k neighbors to use
<code>sNNclust_eps</code>	SNNclust: epsilon
<code>sNNclust_minPts</code>	SNNclust: min points
<code>borderPoints</code>	SNNclust: border points
<code>expression_values</code>	expression values to use
<code>genes_to_use</code>	= NULL,

dim_reduction_to_use	dimension reduction to use
dim_reduction_name	name of reduction 'pca',
dimensions_to_use	dimensions to use
distance_method	distance method
km_centers	kmeans centers
km_iter_max	kmeans iterations
km_nstart	kmeans random starting points
km_algorithm	kmeans algorithm
hc_agglomeration_method	hierarchical clustering method
hc_k	hierachical number of clusters
hc_h	hierarchical tree cutoff
return_gobject	boolean: return giotto object (default = TRUE)
set_seed	set seed
seed_number	number for seed
...	additional parameters

Details

Description of different clustering methods.

Value

giotto object appended with new cluster

Examples

```
clusterCells(gobject)
```

createGiottoObject	<i>create Giotto object</i>
--------------------	-----------------------------

Description

Function to create a giotto object

Usage

```
createGiottoObject(raw_exprs, spatial_locs = NULL, norm_expr = NULL,
  norm_scaled_expr = NULL, custom_expr = NULL, cell_metadata = NULL,
  gene_metadata = NULL, spatial_network = NULL,
  spatial_network_name = NULL, spatial_grid = NULL,
  spatial_grid_name = NULL, dimension_reduction = NULL,
  nn_network = NULL, offset_file = NULL)
```

Arguments

raw_exprs	matrix with raw expression counts [required]
spatial_locs	data.table with coordinates for cell centroids [required]
norm_expr	normalized expression values
norm_scaled_expr	scaled expression values
custom_expr	custom expression values
cell_metadata	cell metadata
gene_metadata	gene metadata
spatial_network	list of spatial network(s)
spatial_network_name	list of spatial network name(s)
spatial_grid	list of spatial grid(s)
spatial_grid_name	list of spatial grid name(s)
dimension_reduction	list of dimension reduction(s)
nn_network	list of nearest neighbor network(s)
offset_file	file used to stitch fields together (optional)

Value

giotto object

Examples

```
createGiottoObject(raw_exprs, spatial_locs)
```

createHeatmap_DT	<i>createHeatmap_DT</i>
------------------	-------------------------

Description

creates order for clusters

Usage

```
createHeatmap_DT(gobject, expression_values = c("normalized", "scaled",
"custom"), genes, cluster_column = NULL, cluster_order = c("size",
"correlation", "custom"), cluster_custom_order = NULL,
cluster_cor_method = "pearson", cluster_hclust_method = "ward.D",
gene_order = c("custom", "correlation"), gene_custom_order = NULL,
gene_cor_method = "pearson", gene_hclust_method = "complete")
```

Arguments

<code>gobject</code>	giotto object
<code>expression_values</code>	expression values to use
<code>genes</code>	genes to use
<code>cluster_column</code>	name of column to use for clusters
<code>cluster_order</code>	method to determine cluster order
<code>cluster_custom_order</code>	custom order for clusters
<code>cluster_cor_method</code>	method for cluster correlation
<code>cluster_hclust_method</code>	method for hierarchical clustering of clusters
<code>gene_order</code>	method to determine gene order
<code>gene_custom_order</code>	custom order for genes
<code>gene_cor_method</code>	method for gene correlation
<code>gene_hclust_method</code>	method for hierarchical clustering of genes

Details

Creates input data.tables for plotHeatmap function.

Value

list

Examples

```
createHeatmap_DT(gobject)
```

<code>createNearestNetwork</code>	<i>createNearestNetwork</i>
-----------------------------------	-----------------------------

Description

create a nearest neighbour network based on previously computed dimension reductions

Usage

```
createNearestNetwork(gobject, expression_values = c("normalized",
  "scaled", "custom"), type = c("sNN", "kNN"),
  dim_reduction_to_use = "pca", dim_reduction_name = "pca",
  dimensions_to_use = 1:10, genes_to_use = NULL, name = "sNN.pca",
  return_gobject = TRUE, k = 30, minimum_shared = 5,
  top_shared = 3, verbose = T, ...)
```


Arguments

<code>gobject</code>	giotto object
<code>expression_values</code>	expression values to use
<code>type</code>	kNN or sNN
<code>dim_reduction_to_use</code>	dimension reduction method to use
<code>dim_reduction_name</code>	name of dimension reduction set to use
<code>dimensions_to_use</code>	number of dimensions to use as input
<code>genes_to_use</code>	if <code>dim_reduction_to_use = NULL</code> , which genes to use
<code>name</code>	arbitrary name for NN network
<code>return_gobject</code>	boolean: return giotto object (default = TRUE)
<code>k</code>	number of k neighbors to use
<code>minimum_shared</code>	minimum shared neighbors
<code>top_shared</code>	keep at ...
<code>verbose</code>	be verbose
<code>...</code>	additional parameters

Details

Description of nearest neighbor network creation and filter steps.

Value

giotto object with updated NN network

Examples

```
createNearestNetwork(gobject)
```

<code>createSpatialGrid</code>	<i>createSpatialGrid2</i>
--------------------------------	---------------------------

Description

create a spatial grid

Usage

```
createSpatialGrid(gobject, sdimx_stepsize = NULL,
  sdimy_stepsize = NULL, sdimz_stepsize = NULL, minimum_padding = 1,
  name = "spatial_grid", return_gobject = TRUE)
```

Arguments

<code>gobject</code>	giotto object
<code>sdimx_stepsize</code>	stepsize along the x-axis
<code>sdimy_stepsize</code>	stepsize along the y-axis
<code>sdimz_stepsize</code>	stepsize along the z-axis
<code>minimum_padding</code>	minimum padding on the edges
<code>name</code>	name for spatial grid (default = 'spatial_grid')
<code>return_gobject</code>	boolean: return giotto object (default = TRUE)

Details

Creates a spatial grid with defined x, y (and z) dimensions.

Value

giotto object with updated spatial grid slot

Examples

```
createSpatialGrid2(gobject)
```

`createSpatialGrid_2D` *createSpatialGrid_2D*

Description

create a spatial grid

Usage

```
createSpatialGrid_2D(gobject, sdimx_stepsize = NULL,
  sdimy_stepsize = NULL, minimum_padding = 1, name = "spatial_grid",
  return_gobject = TRUE)
```

Arguments

<code>gobject</code>	giotto object
<code>sdimx_stepsize</code>	stepsize along the x-axis
<code>sdimy_stepsize</code>	stepsize along the y-axis
<code>minimum_padding</code>	minimum padding on the edges
<code>name</code>	name for spatial grid (default = 'spatial_grid')
<code>return_gobject</code>	boolean: return giotto object (default = TRUE)

Details

Creates a spatial grid with defined x, y (and z) dimensions.

Value

giotto object with updated spatial grid slot

Examples

```
createSpatialGrid_2D(gobject)
```

```
createSpatialGrid_3D    createSpatialGrid_3D
```

Description

create a spatial grid

Usage

```
createSpatialGrid_3D(gobject, sdimx_stepsize = NULL,
  sdimy_stepsize = NULL, sdimz_stepsize = NULL, minimum_padding = 1,
  name = "spatial_grid", return_gobject = TRUE)
```

Arguments

<code>gobject</code>	giotto object
<code>sdimx_stepsize</code>	stepsize along the x-axis
<code>sdimy_stepsize</code>	stepsize along the y-axis
<code>sdimz_stepsize</code>	stepsize along the z-axis
<code>minimum_padding</code>	minimum padding on the edges
<code>name</code>	name for spatial grid (default = 'spatial_grid')
<code>return_gobject</code>	boolean: return giotto object (default = TRUE)

Details

Creates a spatial grid with defined x, y (and z) dimensions.

Value

giotto object with updated spatial grid slot

Examples

```
createSpatialGrid_3D(gobject)
```

createSpatialNetwork	<i>createSpatialNetwork</i>
----------------------	-----------------------------

Description

Create a spatial network based on cell centroid distances.

Usage

```
createSpatialNetwork(gobject, k = 4, dimensions = "all",
  maximum_distance = NULL, minimum_k = 0, name = "spatial_network",
  verbose = F, return_gobject = TRUE)
```

Arguments

<code>gobject</code>	giotto object
<code>k</code>	number of nearest neighbors based on physical distance
<code>dimensions</code>	which spatial dimensions to use (default = all)
<code>maximum_distance</code>	distance cutoff for nearest neighbors to consider
<code>minimum_k</code>	minimum nearest neighbours if <code>maximum_distance</code> != NULL
<code>name</code>	name for spatial network (default = 'spatial_network')
<code>verbose</code>	verbose
<code>return_gobject</code>	boolean: return giotto object (default = TRUE)

Details

Creates a spatial network connecting single-cells based on their physical distance to each other. Number of neighbors can be determined by `k`, maximum distance from each cell with or without setting a minimum `k` for each cell.

Value

giotto object with updated spatial network slot

Examples

```
createSpatialNetwork(gobject)
```

```
create_average_detection_DT
      create_average_detection_DT
```

Description

calculates average gene detection for a cell metadata factor (e.g. cluster)

Usage

```
create_average_detection_DT(gobject, meta_data_name,
  expression_values = c("normalized", "scaled", "custom"),
  detection_threshold = 0)
```

Arguments

```
gobject          giotto object
meta_data_name   name of metadata column to use
expression_values which expression values to use
detection_threshold
                  detection threshold to consider a gene detected
```

Value

data.table with average gene expression values for each factor

```
create_average_DT      create_average_DT
```

Description

calculates average gene expression for a cell metadata factor (e.g. cluster)

Usage

```
create_average_DT(gobject, meta_data_name,
  expression_values = c("normalized", "scaled", "custom"))
```

Arguments

```
gobject          giotto object
meta_data_name   name of metadata column to use
expression_values which expression values to use
```

Value

data.table with average gene expression values for each factor

```
create_cell_type_random_cell_IDs
      create_cell_type_random_cell_IDs
```

Description

creates randomized cell ids within a selection of cell types

Usage

```
create_cell_type_random_cell_IDs(gobject, cluster_column = "cell_types",
      needed_cell_types)
```

Arguments

```
gobject          giotto object to use
cluster_column   cluster column with cell type information
needed_cell_types
                  vector of cell type names for which a random id will be found
```

Details

Details will follow.

Value

list of randomly sampled cell ids with same cell type composition

Examples

```
create_cell_type_random_cell_IDs(gobject)
```

```
create_cluster_matrix  create_cluster_matrix
```

Description

creates aggregated matrix for a given clustering

Usage

```
create_cluster_matrix(gobject, expression_values = c("normalized",
      "scaled", "custom"), cluster_column, gene_subset = NULL)
```

Examples

```
create_cluster_matrix(gobject)
```

create_dimObject	<i>create_dimObject</i>
------------------	-------------------------

Description

Creates an object that stores a dimension reduction output

Usage

```
create_dimObject(name = "test", reduction_method = NULL,
  coordinates = NULL, misc = NULL)
```

Arguments

name	arbitrary name for object
reduction_method	method used to reduce dimensions
coordinates	accepts the coordinates after dimension reduction
misc	any additional information will be added to this slot

Value

number of distinct colors

decide_cluster_order	<i>decide_cluster_order</i>
----------------------	-----------------------------

Description

creates order for clusters

Usage

```
decide_cluster_order(gobject, expression_values = c("normalized",
  "scaled", "custom"), genes, cluster_column = NULL,
  cluster_order = c("size", "correlation", "custom"),
  cluster_custom_order = NULL, cor_method = "pearson",
  hclust_method = "ward.D")
```

Arguments

gobject	giotto object
expression_values	expression values to use
genes	genes to use
cluster_column	name of column to use for clusters
cluster_order	method to determine cluster order
cluster_custom_order	custom order for clusters
cor_method	method for correlation
hclust_method	method for hierarchical clustering

Details

Calculates order for clusters.

Value

custom

Examples

```
decide_cluster_order(gobject)
```

detectSpatialPatterns *detectSpatialPatterns*

Description

Identify spatial patterns through PCA on average expression in a spatial grid.

Usage

```
detectSpatialPatterns(gobject, expression_values = c("normalized",
  "scaled", "custom"), spatial_grid_name = "spatial_grid",
  min_cells_per_grid = 4, scale_unit = F, ncp = 100, show_plot = T,
  PC_zscore = 1.5)
```

Arguments

<code>gobject</code>	giotto object
<code>expression_values</code>	expression values to use
<code>spatial_grid_name</code>	name of spatial grid to use (default = 'spatial_grid')
<code>min_cells_per_grid</code>	minimum number of cells in a grid to be considered
<code>scale_unit</code>	scale features
<code>ncp</code>	number of principal components to calculate
<code>show_plot</code>	show plots
<code>PC_zscore</code>	minimum z-score of variance explained by a PC

Details

Description of how we compute spatial pattern genes.

Value

spatial pattern object 'spatPatObj'

Examples

```
detectSpatialPatterns(gobject)
```

direction_test_CPG	<i>direction_test_CPG</i>
--------------------	---------------------------

Description

shows direction of change

Usage

```
direction_test(x, min_pval = 0.05)
```

Examples

```
direction_test_CPG()
```

doHclust	<i>doHclust</i>
----------	-----------------

Description

cluster cells using hierarchical clustering algorithm

Usage

```
doHclust(gobject, expression_values = c("normalized", "scaled",
    "custom"), genes_to_use = NULL, dim_reduction_to_use = c("cells",
    "pca", "umap", "tsne"), dim_reduction_name = "pca",
    dimensions_to_use = 1:10, distance_method = c("pearson", "spearman",
    "original", "euclidean", "maximum", "manhattan", "canberra", "binary",
    "minkowski"), agglomeration_method = c("ward.D2", "ward.D", "single",
    "complete", "average", "mcquitty", "median", "centroid"), k = 10,
    h = NULL, name = "hclust", return_gobject = TRUE, set_seed = T,
    seed_number = 1234)
```

Arguments

gobject	giotto object
expression_values	expression values to use
genes_to_use	subset of genes to use
dim_reduction_to_use	dimension reduction to use
dim_reduction_name	dimensions reduction name
dimensions_to_use	dimensions to use
distance_method	distance method

agglomeration_method	agglomeration method for hclust
k	number of final clusters
h	cut hierarchical tree at height = h
name	name for hierarchical clustering
return_gobject	boolean: return giotto object (default = TRUE)
set_seed	set seed
seed_number	number for seed
...	additional parameters

Details

Description on how to use Kmeans clustering method.

Value

giotto object appended with new cluster

Examples

```
doHclust(gobject)
```

doHMRF	<i>doHMRF</i>
--------	---------------

Description

Run HMRF

Usage

```
doHMRF(gobject, expression_values = c("normalized", "scaled", "custom"),
  spatial_network_name = "spatial_network", spatial_genes = NULL,
  dim_reduction_to_use = NULL, dim_reduction_name = "pca",
  dimensions_to_use = 1:10, name = "test", k = 10, betas = c(0, 2,
  50), tolerance = 1e-10, zscore = c("none", "rowcol", "colrow"),
  numinit = 100, python_path = NULL, output_folder = NULL)
```

Arguments

gobject	giotto object
expression_values	expression values to use
spatial_network_name	name of spatial network to use for HMRF
spatial_genes	spatial genes to use for HMRF
dim_reduction_to_use	use another dimension reduction set as input

dim_reduction_name	name of dimension reduction set to use
dimensions_to_use	number of dimensions to use as input
name	name of HMRF run
k	number of HMRF domains
betas	betas to test for
tolerance	tolerance
zscore	zscore
numinit	number of initializations
python_path	python path to use
output_folder	output folder to save results

Details

Description of HMRF parameters ...

Value

Creates a directory with results that can be viewed with viewHMRFresults

Examples

doHMRF(gobject)

doKmeans	<i>doKmeans</i>
----------	-----------------

Description

cluster cells using kmeans algorithm

Usage

```
doKmeans(gobject, expression_values = c("normalized", "scaled",
    "custom"), genes_to_use = NULL, dim_reduction_to_use = c("cells",
    "pca", "umap", "tsne"), dim_reduction_name = "pca",
    dimensions_to_use = 1:10, distance_method = c("original", "pearson",
    "spearman", "euclidean", "maximum", "manhattan", "canberra", "binary",
    "minkowski"), centers = 10, iter_max = 100, nstart = 1000,
    algorithm = "Hartigan-Wong", name = "kmeans",
    return_gobject = TRUE, set_seed = T, seed_number = 1234)
```

Arguments

<code>gobject</code>	giotto object
<code>expression_values</code>	expression values to use
<code>genes_to_use</code>	subset of genes to use
<code>dim_reduction_to_use</code>	dimension reduction to use
<code>dim_reduction_name</code>	dimensions reduction name
<code>dimensions_to_use</code>	dimensions to use
<code>distance_method</code>	distance method
<code>centers</code>	number of final clusters
<code>iter_max</code>	kmeans maximum iterations
<code>nstart</code>	kmeans nstart
<code>algorithm</code>	kmeans algorithm
<code>name</code>	name for kmeans clustering
<code>return_gobject</code>	boolean: return giotto object (default = TRUE)
<code>set_seed</code>	set seed
<code>seed_number</code>	number for seed
<code>...</code>	additional parameters

Details

Description on how to use Kmeans clustering method.

Value

giotto object appended with new cluster

Examples

```
doKmeans(gobject)
```

doLeidenCluster

doLeidenCluster

Description

cluster cells using a NN-network and the Leiden community detection algorithm

Usage

```
doLeidenCluster(gobject, name = "leiden_clus",
  nn_network_to_use = "sNN", network_name = "sNN.pca",
  python_path = NULL, resolution = 1, weight_col = "weight",
  partition_type = c("RBConfigurationVertexPartition",
    "ModularityVertexPartition"), init_membership = NULL,
  n_iterations = 1000, return_gobject = TRUE, set_seed = T,
  seed_number = 1234, ...)
```

Arguments

<code>gobject</code>	giotto object
<code>name</code>	name for cluster
<code>nn_network_to_use</code>	type of NN network to use (kNN vs sNN)
<code>network_name</code>	name of NN network to use
<code>python_path</code>	specify specific path to python if required
<code>resolution</code>	resolution
<code>weight_col</code>	weight column
<code>partition_type</code>	partition type to use
<code>init_membership</code>	initial membership of cells
<code>n_iterations</code>	number of iterations
<code>return_gobject</code>	boolean: return giotto object (default = TRUE)
<code>set_seed</code>	set seed
<code>seed_number</code>	number for seed
<code>...</code>	additional parameters

Details

Description of Leiden clustering method.

Value

giotto object appended with new cluster

Examples

```
doLeidenCluster(gobject)
```

<code>doLeidenSubCluster</code>	<i>doLeidenSubCluster</i>
---------------------------------	---------------------------

Description

subcluster cells using a NN-network and the Leiden algorithm

Usage

```
doLeidenSubCluster(gobject, name = "sub_pleiden_clus",
  cluster_column = NULL, selected_clusters = NULL,
  hvg_param = list(reverse_log_scale = T, difference_in_variance = 1,
  expression_values = "normalized"), hvg_min_perc_cells = 5,
  hvg_mean_expr_det = 1, use_all_genes_as_hvg = FALSE,
  min_nr_of_hvg = 5, pca_param = list(expression_values = "normalized",
  scale_unit = T), nn_param = list(dimensions_to_use = 1:20),
  k_neighbors = 10, resolution = 0.5, n_iterations = 500,
  python_path = NULL, nn_network_to_use = "sNN",
  network_name = "sNN.pca", return_gobject = TRUE, verbose = T, ...)
```

Arguments

<code>gobject</code>	giotto object
<code>name</code>	name for new clustering result
<code>cluster_column</code>	cluster column to subcluster
<code>selected_clusters</code>	only do subclustering on these clusters
<code>hvg_param</code>	parameters for calculateHVG
<code>hvg_min_perc_cells</code>	threshold for detection in min percentage of cells
<code>hvg_mean_expr_det</code>	threshold for mean expression level in cells with detection
<code>use_all_genes_as_hvg</code>	forces all genes to be HVG and to be used as input for PCA
<code>min_nr_of_hvg</code>	minimum number of HVG, or all genes will be used as input for PCA
<code>pca_param</code>	parameters for runPCA
<code>nn_param</code>	parameters for parameters for createNearestNetwork
<code>k_neighbors</code>	number of k for createNearestNetwork
<code>resolution</code>	resolution of Leiden clustering
<code>n_iterations</code>	number of iterations
<code>python_path</code>	specify specific path to python if required
<code>nn_network_to_use</code>	type of NN network to use (kNN vs sNN)
<code>network_name</code>	name of NN network to use
<code>return_gobject</code>	boolean: return giotto object (default = TRUE)
<code>verbose</code>	verbose
<code>...</code>	additional parameters

Details

Description of Leiden clustering method.

Value

giotto object appended with new cluster

Examples

```
doLeidenSubCluster(gobject)
```

doLouvainCluster	<i>doLouvainCluster</i>
------------------	-------------------------

Description

cluster cells using a NN-network and the Louvain algorithm.

Usage

```
doLouvainCluster(gobject, version = c("community", "multinet"),
  name = "louvain_clus", nn_network_to_use = "sNN",
  network_name = "sNN.pca", python_path = NULL, resolution = 1,
  weight_col = NULL, gamma = 1, omega = 1, louv_random = F,
  return_gobject = TRUE, set_seed = F, seed_number = 1234, ...)
```

Arguments

gobject	giotto object
version	implemented version of Louvain clustering to use
name	name for cluster
nn_network_to_use	type of NN network to use (kNN vs sNN)
network_name	name of NN network to use
python_path	specify specific path to python if required
resolution	resolution
gamma	gamma
omega	omega
return_gobject	boolean: return giotto object (default = TRUE)
set_seed	set seed
seed_number	number for seed
...	additional parameters

Details

Louvain clustering using the community or multinet implementation of the louvain clustering algorithm.

Value

giotto object appended with new cluster

Examples

```
doLouvainCluster(gobject)
```

```
doLouvainCluster_community
    doLouvainCluster_community
```

Description

cluster cells using a NN-network and the Louvain algorithm from the community module in Python

Usage

```
doLouvainCluster_community(gobject, name = "louvain_clus",
    nn_network_to_use = "sNN", network_name = "sNN.pca",
    python_path = NULL, resolution = 1, weight_col = NULL,
    louv_random = F, return_gobject = TRUE, set_seed = F,
    seed_number = 1234, ...)
```

Arguments

<code>gobject</code>	giotto object
<code>name</code>	name for cluster
<code>nn_network_to_use</code>	type of NN network to use (kNN vs sNN)
<code>network_name</code>	name of NN network to use
<code>python_path</code>	specify specific path to python if required
<code>resolution</code>	resolution
<code>weight_col</code>	weight column
<code>louv_random</code>	random
<code>return_gobject</code>	boolean: return giotto object (default = TRUE)
<code>set_seed</code>	set seed
<code>seed_number</code>	number for seed
<code>...</code>	additional parameters

Details

Description of Leiden clustering method.

Value

giotto object appended with new cluster

Examples

```
doLouvainCluster_community(gobject)
```

```
doLouvainCluster_multinet
doLouvainCluster_multinet
```

Description

cluster cells using a NN-network and the Louvain algorithm from the multinet package in R.

Usage

```
doLouvainCluster_multinet(gobject, name = "louvain_clus",
  nn_network_to_use = "sNN", network_name = "sNN.pca",
  weight_col = NULL, gamma = 1, omega = 1, return_gobject = TRUE,
  set_seed = F, seed_number = 1234, ...)
```

Arguments

<code>gobject</code>	giotto object
<code>name</code>	name for cluster
<code>nn_network_to_use</code>	type of NN network to use (kNN vs sNN)
<code>network_name</code>	name of NN network to use
<code>gamma</code>	gamma
<code>omega</code>	omega
<code>return_gobject</code>	boolean: return giotto object (default = TRUE)
<code>set_seed</code>	set seed
<code>seed_number</code>	number for seed
<code>...</code>	additional parameters
<code>python_path</code>	specify specific path to python if required

Details

See louvain algorithm from the multinet package in R.

Value

giotto object appended with new cluster

Examples

```
doLouvainCluster_multinet(gobject)
```

doLouvainSubCluster *doLouvainSubCluster*

Description

subcluster cells using a NN-network and the Louvain algorithm

Usage

```
doLouvainSubCluster(gobject, name = "sub_louvain_clus",
  version = c("community", "multinet"), cluster_column = NULL,
  selected_clusters = NULL, hvg_param = list(reverse_log_scale = T,
  difference_in_variance = 1, expression_values = "normalized"),
  hvg_min_perc_cells = 5, hvg_mean_expr_det = 1,
  use_all_genes_as_hvg = FALSE, min_nr_of_hvg = 5,
  pca_param = list(expression_values = "normalized", scale_unit = T),
  nn_param = list(dimensions_to_use = 1:20), k_neighbors = 10,
  resolution = 0.5, gamma = 1, omega = 1, python_path = NULL,
  nn_network_to_use = "sNN", network_name = "sNN.pca",
  return_gobject = TRUE, verbose = T, ...)
```

Arguments

<code>gobject</code>	giotto object
<code>name</code>	name for new clustering result
<code>version</code>	version of Louvain algorithm to use
<code>cluster_column</code>	cluster column to subcluster
<code>selected_clusters</code>	only do subclustering on these clusters
<code>hvg_param</code>	parameters for calculateHVG
<code>hvg_min_perc_cells</code>	threshold for detection in min percentage of cells
<code>hvg_mean_expr_det</code>	threshold for mean expression level in cells with detection
<code>use_all_genes_as_hvg</code>	forces all genes to be HVG and to be used as input for PCA
<code>min_nr_of_hvg</code>	minimum number of HVG, or all genes will be used as input for PCA
<code>pca_param</code>	parameters for runPCA
<code>nn_param</code>	parameters for parameters for createNearestNetwork
<code>k_neighbors</code>	number of k for createNearestNetwork
<code>resolution</code>	resolution for community algorithm
<code>gamma</code>	gamma
<code>omega</code>	omega
<code>python_path</code>	specify specific path to python if required
<code>nn_network_to_use</code>	type of NN network to use (kNN vs sNN)

network_name	name of NN network to use
return_gobject	boolean: return giotto object (default = TRUE)
verbose	verbose
...	additional parameters

Details

Description of Louvain clustering method.

Value

giotto object appended with new cluster

Examples

```
doLouvainSubCluster(gobject)
```

```
doLouvainSubCluster_community
doLouvainSubCluster_community
```

Description

subcluster cells using a NN-network and the Louvain community detection algorithm

Usage

```
doLouvainSubCluster_community(gobject, name = "sub_louvain_comm_clus",
  cluster_column = NULL, selected_clusters = NULL,
  hvg_param = list(reverse_log_scale = T, difference_in_variance = 1,
  expression_values = "normalized"), hvg_min_perc_cells = 5,
  hvg_mean_expr_det = 1, use_all_genes_as_hvg = FALSE,
  min_nr_of_hvg = 5, pca_param = list(expression_values = "normalized",
  scale_unit = T), nn_param = list(dimensions_to_use = 1:20),
  k_neighbors = 10, resolution = 0.5, python_path = NULL,
  nn_network_to_use = "sNN", network_name = "sNN.pca",
  return_gobject = TRUE, verbose = T, ...)
```

Arguments

gobject	giotto object
name	name for new clustering result
cluster_column	cluster column to subcluster
selected_clusters	only do subclustering on these clusters
hvg_param	parameters for calculateHVG
hvg_min_perc_cells	threshold for detection in min percentage of cells
hvg_mean_expr_det	threshold for mean expression level in cells with detection

use_all_genes_as_hvg	forces all genes to be HVG and to be used as input for PCA
min_nr_of_hvg	minimum number of HVG, or all genes will be used as input for PCA
pca_param	parameters for runPCA
nn_param	parameters for parameters for createNearestNetwork
k_neighbors	number of k for createNearestNetwork
resolution	resolution
python_path	specify specific path to python if required
nn_network_to_use	type of NN network to use (kNN vs sNN)
network_name	name of NN network to use
return_gobject	boolean: return giotto object (default = TRUE)
verbose	verbose
...	additional parameters

Details

Description of Leiden clustering method.

Value

giotto object appended with new cluster

Examples

```
doLouvainSubCluster_community(gobject)
```

```
doLouvainSubCluster_multinet
doLouvainSubCluster_multinet
```

Description

subcluster cells using a NN-network and the Louvain multinet detection algorithm

Usage

```
doLouvainSubCluster_multinet(gobject, name = "sub_louvain_mult_clus",
  cluster_column = NULL, selected_clusters = NULL,
  hvg_param = list(reverse_log_scale = T, difference_in_variance = 1,
  expression_values = "normalized"), hvg_min_perc_cells = 5,
  hvg_mean_expr_det = 1, use_all_genes_as_hvg = FALSE,
  min_nr_of_hvg = 5, pca_param = list(expression_values = "normalized",
  scale_unit = T), nn_param = list(dimensions_to_use = 1:20),
  k_neighbors = 10, gamma = 1, omega = 1,
  nn_network_to_use = "sNN", network_name = "sNN.pca",
  return_gobject = TRUE, verbose = T, ...)
```

Arguments

<code>gobject</code>	giotto object
<code>name</code>	name for new clustering result
<code>cluster_column</code>	cluster column to subcluster
<code>selected_clusters</code>	only do subclustering on these clusters
<code>hvg_param</code>	parameters for calculateHVG
<code>hvg_min_perc_cells</code>	threshold for detection in min percentage of cells
<code>hvg_mean_expr_det</code>	threshold for mean expression level in cells with detection
<code>use_all_genes_as_hvg</code>	forces all genes to be HVG and to be used as input for PCA
<code>min_nr_of_hvg</code>	minimum number of HVG, or all genes will be used as input for PCA
<code>pca_param</code>	parameters for runPCA
<code>nn_param</code>	parameters for parameters for createNearestNetwork
<code>k_neighbors</code>	number of k for createNearestNetwork
<code>gamma</code>	gamma
<code>omega</code>	omega
<code>nn_network_to_use</code>	type of NN network to use (kNN vs sNN)
<code>network_name</code>	name of NN network to use
<code>return_gobject</code>	boolean: return giotto object (default = TRUE)
<code>verbose</code>	verbose
<code>...</code>	additional parameters
<code>python_path</code>	specify specific path to python if required

Details

Description of Louvain clustering method.

Value

giotto object appended with new cluster

Examples

```
doLouvainSubCluster_multinet(gobject)
```

doRandomWalkCluster	<i>doRandomWalkCluster</i>
---------------------	----------------------------

Description

Cluster cells using a random walk approach.

Usage

```
doRandomWalkCluster(gobject, name = "random_walk_clus",  
  nn_network_to_use = "sNN", network_name = "sNN.pca",  
  walk_steps = 4, walk_clusters = 10, walk_weights = NA,  
  return_gobject = TRUE, set_seed = F, seed_number = 1234, ...)
```

Arguments

<code>gobject</code>	giotto object
<code>name</code>	name for cluster
<code>nn_network_to_use</code>	type of NN network to use (kNN vs sNN)
<code>network_name</code>	name of NN network to use
<code>walk_steps</code>	number of walking steps
<code>walk_clusters</code>	number of final clusters
<code>walk_weights</code>	cluster column defining the walk weights
<code>return_gobject</code>	boolean: return giotto object (default = TRUE)
<code>set_seed</code>	set seed
<code>seed_number</code>	number for seed
<code>...</code>	additional parameters

Details

See random walk algorithm from the igraph package in R.

Value

giotto object appended with new cluster

Examples

```
doRandomWalkCluster(gobject)
```

doSNNCluster

*doSNNCluster***Description**

Cluster cells using a SNN cluster approach.

Usage

```
doSNNCluster(gobject, name = "sNN_clus", nn_network_to_use = "kNN",
  network_name = "kNN.pca", k = 20, eps = 4, minPts = 16,
  borderPoints = TRUE, return_gobject = TRUE, set_seed = F,
  seed_number = 1234, ...)
```

Arguments

gobject	giotto object
name	name for cluster
nn_network_to_use	type of NN network to use (only works on kNN)
network_name	name of kNN network to use
k	Neighborhood size for nearest neighbor sparsification to create the shared NN graph.
eps	Two objects are only reachable from each other if they share at least eps nearest neighbors.
minPts	minimum number of points that share at least eps nearest neighbors for a point to be considered a core points.
borderPoints	should borderPoints be assigned to clusters like in DBSCAN?
return_gobject	boolean: return giotto object (default = TRUE)
set_seed	set seed
seed_number	number for seed
...	additional parameters

Details

See sNNclust algorithm from dbscan package

Value

giotto object appended with new cluster

Examples

```
doSNNCluster(gobject)
```

dt_to_matrix	<i>dt_to_matrix</i>
--------------	---------------------

Description

converts data.table to matrix

Usage

```
dt_to_matrix(x)
```

Examples

```
dt_to_matrix(x)
```

exportGiottoViewer	<i>exportGiottoViewer</i>
--------------------	---------------------------

Description

compute highly variable genes

Usage

```
exportGiottoViewer(gobject, output_directory = NULL, annotations,
  dim_reductions, dim_reduction_names,
  expression_values = c("normalized", "scaled", "custom"),
  dim_red_rounding = NULL, dim_red_rescale = c(-20, 20),
  expression_rounding = NULL, overwrite_dir = F, verbose = T)
```

Arguments

gobject	giotto object
output_directory	directory where to save the files
annotations	giotto cell annotations to view
dim_reductions	high level dimension reductions to view
dim_reduction_names	specific dimension reduction names
expression_values	expression values to use in Viewer
dim_red_rounding	numerical indicating how to round the coordinates
dim_red_rescale	numericals to rescale the coordinates
expression_rounding	numerical indicating how to round the expression data
overwrite_dir	overwrite files in the directory if it already existed
verbose	be verbose

Details

Giotto Viewer expects the results from Giotto Analyzer in a specific format, which is provided by this function.

Value

writes the necessary output to use in Giotto Viewer

Examples

```
exportGiottoViewer(gobject)
```

`exprOnlyCellCellcommunicationScores`

exprOnlyCellCellcommunicationScores

Description

Cell-Cell communication scores based on expression only

Usage

```
exprOnlyCellCellcommunicationScores(gobject,
  cluster_column = "cell_types", random_iter = 100, gene_set_1,
  gene_set_2, log2FC_addendum = 0.1, verbose = T)
```

Arguments

<code>gobject</code>	giotto object to use
<code>cluster_column</code>	cluster column with cell type information
<code>random_iter</code>	number of iterations
<code>gene_set_1</code>	first specific gene set from gene pairs
<code>gene_set_2</code>	second specific gene set from gene pairs
<code>log2FC_addendum</code>	addendum to add when calculating log2FC
<code>verbose</code>	verbose

Details

Details will follow.

Value

Cell-Cell communication scores for gene pairs based on expression only

Examples

```
exprOnlyCellCellcommunicationScores(gobject)
```

extended_gini_fun	<i>extended_gini_fun</i>
-------------------	--------------------------

Description

calculate gini coefficient on a minimum length vector

Usage

```
extended_gini_fun(x, weights = rep(1, length = length(x)),
  minimum_length = 16)
```

Value

gini coefficient

extractNearestNetwork	<i>extractNearestNetwork</i>
-----------------------	------------------------------

Description

Extracts a NN-network from a Giotto object as an igraph object

Usage

```
extractNearestNetwork(gobject, nn_network_to_use = "sNN",
  network_name = "sNN.pca")
```

Arguments

gobject	giotto object
nn_network_to_use	kNN or sNN
network_name	name of NN network to be used

Value

igraph object

Examples

```
extractNearestNetwork(gobject)
```

fDataDT

*fDataDT***Description**

show gene metadata

Usage

fDataDT(gobject)

Arguments

gobject giotto object

Value

data.table

Examples

pDataDT(gobject)

filterCombinations

*filterCombinations***Description**

Shows how many genes and cells are lost with combinations of thresholds.

Usage

```
filterCombinations(gobject, expression_values = c("raw", "normalized",
  "scaled", "custom"), expression_thresholds = c(1, 2),
  gene_det_in_min_cells = c(5, 50), min_det_genes_per_cell = c(200,
  400), scale_x_axis = "identity", x_axis_offset = 0,
  scale_y_axis = "identity", y_axis_offset = 0, show_plot = TRUE)
```

Arguments

gobject giotto object

expression_values
 expression values to use

expression_thresholds
 all thresholds to consider a gene expressed

gene_det_in_min_cells
 minimum number of cells that should express a gene to consider that gene further

min_det_genes_per_cell
 minimum number of expressed genes per cell to consider that cell further

scale_x_axis	ggplot transformation for x-axis (e.g. log2)
x_axis_offset	x-axis offset to be used together with the scaling transformation
scale_y_axis	ggplot transformation for y-axis (e.g. log2)
y_axis_offset	y-axis offset to be used together with the scaling transformation
show_plot	show plot

Details

Creates a scatterplot that visualizes the number of genes and cells that are lost with a specific combination of a gene and cell threshold given an arbitrary cutoff to call a gene expressed. This function can be used to make an informed decision at the filtering step with filterGiotto.

Value

list of data.table and ggplot object

Examples

```
filterCombinations(gobject)
```

filterDistributions	<i>filterDistributions</i>
---------------------	----------------------------

Description

show gene or cell filter distributions

Usage

```
filterDistributions(gobject, expression_values = c("raw", "normalized",
  "scaled", "custom"), expression_threshold = 1, detection = c("genes",
  "cells"), plot_type = c("histogram", "violin"), nr_bins = 30,
  fill_color = "lightblue", scale_axis = "identity", axis_offset = 0,
  show_plot = TRUE)
```

Arguments

gobject	giotto object
expression_values	expression values to use
expression_threshold	threshold to consider a gene expressed
detection	look at genes or cells
plot_type	type of plot
nr_bins	number of bins for histogram plot
fill_color	fill color for plots
scale_axis	ggplot transformation for axis (e.g. log2)
axis_offset	offset to be used together with the scaling transformation
show_plot	show plot

Value

ggplot object

Examples

```
filterDistributions(gobject)
```

filterGiotto	<i>filterGiotto</i>
--------------	---------------------

Description

filter Giotto object

Usage

```
filterGiotto(gobject, expression_values = c("raw", "normalized",
      "scaled", "custom"), expression_threshold = 1,
      gene_det_in_min_cells = 100, min_det_genes_per_cell = 100,
      verbose = F)
```

Arguments

gobject	giotto object
expression_values	expression values to use
expression_threshold	threshold to consider a gene expressed
gene_det_in_min_cells	minimum # of cells that need to express a gene
min_det_genes_per_cell	minimum # of genes that need to be detected in a cell
verbose	verbose

Value

giotto object

Examples

```
filterGiotto(gobject)
```

findGiniMarkers	<i>findGiniMarkers</i>
-----------------	------------------------

Description

Identify marker genes for selected clusters based on gini detection and expression scores.

Usage

```
findGiniMarkers(gobject, expression_values = c("normalized", "scaled",
  "custom"), cluster_column, subset_clusters = NULL, group_1 = NULL,
  group_2 = NULL, min_expr_gini_score = 0.5,
  min_det_gini_score = 0.5, detection_threshold = 0, rank_score = 1)
```

Arguments

gobject	giotto object
expression_values	gene expression values to use
cluster_column	clusters to use
subset_clusters	selection of clusters to compare
group_1	group 1 cluster IDs from cluster_column for pairwise comparison
group_2	group 2 cluster IDs from cluster_column for pairwise comparison
min_expr_gini_score	filter on minimum gini coefficient for expression
min_det_gini_score	filter minimum gini coefficient for detection
detection_threshold	detection threshold for gene expression
rank_score	rank scores to include

Details

Description of parameters.

Value

data.table with marker genes

Examples

```
findGiniMarkers(gobject)
```

```
findGiniMarkers_one_vs_all
      findGiniMarkers_one_vs_all
```

Description

Identify marker genes for all clusters based on gini detection and expression scores.

Usage

```
findGiniMarkers_one_vs_all(gobject, expression_values = c("normalized",
  "scaled", "custom"), cluster_column, subset_clusters = NULL,
  min_expr_gini_score = 0.5, min_det_gini_score = 0.5,
  detection_threshold = 0, min_genes = 10, verbose = TRUE)
```

Arguments

<code>gobject</code>	giotto object
<code>expression_values</code>	gene expression values to use
<code>cluster_column</code>	clusters to use
<code>subset_clusters</code>	selection of clusters to compare
<code>min_expr_gini_score</code>	filter on minimum gini coefficient on expression
<code>min_det_gini_score</code>	filter on minimum gini coefficient on detection
<code>detection_threshold</code>	detection threshold for gene expression
<code>min_genes</code>	minimum genes to keep per cluster, overrides pval and logFC
<code>verbose</code>	be verbose

Details

Description of parameters.

Value

data.table with marker genes

Examples

```
findGiniMarkers_one_vs_all(gobject)
```

findMarkers	<i>findMarkers</i>
-------------	--------------------

Description

Identify marker genes for selected clusters.

Usage

```
findMarkers(gobject, expression_values = c("normalized", "scaled",
      "custom"), cluster_column, method = c("scrn", "gini", "mast"),
      subset_clusters = NULL, group_1 = NULL, group_2 = NULL,
      min_expr_gini_score = 0.5, min_det_gini_score = 0.5,
      detection_threshold = 0, rank_score = 1, group_1_name = NULL,
      group_2_name = NULL, adjust_columns = NULL, ...)
```

Arguments

gobject	giotto object
expression_values	gene expression values to use
cluster_column	clusters to use
method	method to use to detect differentially expressed genes
subset_clusters	selection of clusters to compare
group_1	group 1 cluster IDs from cluster_column for pairwise comparison
group_2	group 2 cluster IDs from cluster_column for pairwise comparison
min_expr_gini_score	gini: filter on minimum gini coefficient for expression
min_det_gini_score	gini: filter minimum gini coefficient for detection
detection_threshold	gini: detection threshold for gene expression
rank_score	gini: rank scores to include
group_1_name	mast: custom name for group_1 clusters
group_2_name	mast: custom name for group_2 clusters
adjust_columns	mast: column in pDataDT to adjust for (e.g. detection rate)
...	additional parameters for the findMarkers function in scrn or zlm function in MAST

Details

Wrapper for findScranMarkers, findGiniMarkers and FindMastMarkers.

Value

data.table with marker genes

Examples

```
findMarkers(gobject)
```

```
findMarkers_one_vs_all
```

```
findMarkers_one_vs_all
```

Description

Identify marker genes for all clusters.

Usage

```
findMarkers_one_vs_all(gobject, expression_values = c("normalized",
  "scaled", "custom"), cluster_column, subset_clusters = NULL,
  method = c("scrn", "gini", "mast"), pval = 0.01, logFC = 0.5,
  min_genes = 10, min_expr_gini_score = 0.5,
  min_det_gini_score = 0.5, detection_threshold = 0, rank_score = 1,
  adjust_columns = NULL, verbose = TRUE, ...)
```

Arguments

<code>gobject</code>	giotto object
<code>expression_values</code>	gene expression values to use
<code>cluster_column</code>	clusters to use
<code>subset_clusters</code>	selection of clusters to compare
<code>method</code>	method to use to detect differentially expressed genes
<code>pval</code>	scrn & mast: filter on minimal p-value
<code>logFC</code>	scan & mast: filter on logFC
<code>min_genes</code>	minimum genes to keep per cluster, overrides pval and logFC
<code>min_expr_gini_score</code>	gini: filter on minimum gini coefficient for expression
<code>min_det_gini_score</code>	gini: filter minimum gini coefficient for detection
<code>detection_threshold</code>	gini: detection threshold for gene expression
<code>rank_score</code>	gini: rank scores to include
<code>adjust_columns</code>	mast: column in pDataDT to adjust for (e.g. detection rate)
<code>verbose</code>	be verbose
<code>...</code>	additional parameters for the findMarkers function in scrn or zlm function in MAST

Details

Wrapper for findScrnMarkers_one_vs_all, findGiniMarkers_one_vs_all and FindMastMarkers_one_vs_all.

Value

data.table with marker genes

Examples

```
findMarkers_one_vs_all(gobject)
```

findMastMarkers	<i>findMastMarkers</i>
-----------------	------------------------

Description

Identify marker genes for selected clusters based on the MAST package.

Usage

```
findMastMarkers(gobject, expression_values = c("normalized", "scaled",
"custom"), cluster_column, group_1 = NULL, group_1_name = NULL,
group_2 = NULL, group_2_name = NULL, adjust_columns = NULL, ...)
```

Arguments

gobject	giotto object
expression_values	gene expression values to use
cluster_column	clusters to use
group_1	group 1 cluster IDs from cluster_column for pairwise comparison
group_1_name	custom name for group_1 clusters
group_2	group 2 cluster IDs from cluster_column for pairwise comparison
group_2_name	custom name for group_2 clusters
adjust_columns	column in pDataDT to adjust for (e.g. detection rate)
...	additional parameters for the zlm function in MAST

Details

This is a minimal convenience wrapper around the MAST functions to detect differentially expressed genes.

Value

data.table with marker genes

Examples

```
findMastMarkers(gobject)
```

```
findMastMarkers_one_vs_all  
  findMastMarkers_one_vs_all
```

Description

Identify marker genes for all clusters based on the MAST package.

Usage

```
findMastMarkers_one_vs_all(gobject, expression_values = c("normalized",  
  "scaled", "custom"), cluster_column, subset_clusters = NULL,  
  adjust_columns = NULL, pval = 0.001, logFC = 1, min_genes = 10,  
  verbose = TRUE, ...)
```

Arguments

<code>gobject</code>	giotto object
<code>expression_values</code>	gene expression values to use
<code>cluster_column</code>	clusters to use
<code>subset_clusters</code>	selection of clusters to compare
<code>adjust_columns</code>	column in pDataDT to adjust for (e.g. detection rate)
<code>pval</code>	filter on minimal p-value
<code>logFC</code>	filter on logFC
<code>min_genes</code>	minimum genes to keep per cluster, overrides pval and logFC
<code>verbose</code>	be verbose
<code>...</code>	additional parameters for the zlm function in MAST

Details

This is a minimal convenience wrapper around the MAST functions to detect differentially expressed genes.

Value

data.table with marker genes

Examples

```
findMastMarkers_one_vs_all(gobject)
```

findScranMarkers	<i>findScranMarkers</i>
------------------	-------------------------

Description

Identify marker genes for selected clusters based on scran's implementation of findMarkers.

Usage

```
findScranMarkers(gobject, expression_values = c("normalized", "scaled",
  "custom"), cluster_column, subset_clusters = NULL, group_1 = NULL,
  group_2 = NULL, ...)
```

Arguments

gobject	giotto object
expression_values	gene expression values to use
cluster_column	clusters to use
subset_clusters	selection of clusters to compare
group_1	group 1 cluster IDs from cluster_column for pairwise comparison
group_2	group 2 cluster IDs from cluster_column for pairwise comparison
...	additional parameters for the findMarkers function in scran

Details

This is a minimal convenience wrapper around the findMarkers function from the scran package.

Value

data.table with marker genes

Examples

```
findScranMarkers(gobject)
```

findScranMarkers_one_vs_all	<i>findScranMarkers_one_vs_all</i>
-----------------------------	------------------------------------

Description

Identify marker genes for all clusters in a one vs all manner based on scran's implementation of findMarkers.

Usage

```
findScranMarkers_one_vs_all(gobject, expression_values = c("normalized",
  "scaled", "custom"), cluster_column, subset_clusters = NULL,
  pval = 0.01, logFC = 0.5, min_genes = 10, verbose = TRUE, ...)
```

Arguments

gobject	giotto object
expression_values	gene expression values to use
cluster_column	clusters to use
subset_clusters	subset of clusters to use
pval	filter on minimal p-value
logFC	filter on logFC
min_genes	minimum genes to keep per cluster, overrides pval and logFC
verbose	be verbose
...	additional parameters for the findMarkers function in scran

Details

This is a minimal convenience wrapper around the findMarkers function from the scran package.

Value

data.table with marker genes

Examples

```
findScranMarkers_one_vs_all(gobject)
```

find_grid_2D

find_grid_2D

Description

find grid location in 2D

Usage

```
find_grid_2D(grid_DT, x_loc, y_loc)
```

find_grid_3D	<i>find_grid_3D</i>
--------------	---------------------

Description

find grid location in 3D

Usage

```
find_grid_3D(grid_DT, x_loc, y_loc, z_loc)
```

find_grid_x	<i>find_grid_x</i>
-------------	--------------------

Description

find grid location on x-axis

Usage

```
find_grid_x(grid_DT, x_loc)
```

find_grid_y	<i>find_grid_y</i>
-------------	--------------------

Description

find grid location on y-axis

Usage

```
find_grid_y(grid_DT, y_loc)
```

find_grid_z	<i>find_grid_z</i>
-------------	--------------------

Description

find grid location on z-axis

Usage

```
find_grid_z(grid_DT, z_loc)
```

fish_function	<i>fish_function</i>
---------------	----------------------

Description

perform fisher exact test

Usage

```
fish_function(x_to, x_from)
```

fish_function2	<i>fish_function2</i>
----------------	-----------------------

Description

perform fisher exact test

Usage

```
fish_function2(A, B, C, D)
```

FSV_show	<i>FSV_show</i>
----------	-----------------

Description

Visualize spatial variable genes caculated by spatial_DE

Usage

```
FSV_show(results, ms_results = NULL, size = c(4, 2, 1),
  color = c("blue", "green", "red"), sig_alpha = 0.5,
  unsig_alpha = 0.5)
```

Arguments

results	results caculated by spatial_DE
ms_results	ms_results caculated by spatial_DE
size	indicate different levels of qval
color	indicate different SV features
sig_alpha	transparency of significant genes
unsig_alpha	transparency of insignificant genes

Details

Description of parameters.

Value

nothing

Examples

```
FSV_show(results)
```

GenePattern_show	<i>GenePattern_show</i>
------------------	-------------------------

Description

Visualize genes distribution patterns calculated by spatial_AEH

Usage

```
GenePattern_show(gobject = NULL, AEH_results = NULL, sdimx = NULL,
  sdimy = NULL, point_size = 3, point_alpha = 1,
  low_color = "blue", mid_color = "white", high_color = "red",
  midpoint = 0)
```

Arguments

gobject	giotto object
AEH_results	results from spatial_AEH
sdimx	x axis of spatial locus
sdimy	y axis of spatial locus
point_size	size of points to indicate cells
point_alpha	transparency of points to indicate cells
low_color	color to indicate low score level
mid_color	color to indicate middle score level
high_color	color to indicate high score level
midpoint	point to set mid_color

Details

Description of parameters.

Value

nothing

Examples

```
GenePattern_show(gobject,AEH_results)
```

```
getCellProximityGeneScores  
    getCellProximityGeneScores
```

Description

Compute cell-cell interaction enrichment (observed vs expected)

Usage

```
getCellProximityGeneScores(gobject,  
  spatial_network_name = "spatial_network",  
  cluster_column = "louvain_clus.1",  
  expression_values = c("normalized", "scaled", "custom"),  
  fold_change_addendum = 0.1, in_two_directions = TRUE,  
  exclude_selected_cells_from_test = F, verbose = T)
```

Arguments

gobject	giotto object
spatial_network_name	name of spatial network to use
cluster_column	name of column to use for clusters
expression_values	expression values to use
fold_change_addendum	constant to add when calculating log2 fold-change
in_two_directions	shows enrichment in both directions: cell1-cell2, cell2-cell1
exclude_selected_cells_from_test	exclude certain cells from test
verbose	verbose

Details

Give more details ...

Value

Cell Proximity Gene scores (CPGscores) in data.table format

Examples

```
getCellProximityGeneScores(gobject)
```

getClusterSimilarity *getClusterSimilarity*

Description

Creates data.table with pairwise correlation scores between each cluster.

Usage

```
getClusterSimilarity(gobject, expression_values = c("normalized",
  "scaled", "custom"), cluster_column, cor = c("pearson", "spearman"))
```

Arguments

gobject	giotto object
expression_values	expression values to use
cluster_column	name of column to use for clusters
cor	correlation score to calculate distance

Details

Creates data.table with pairwise correlation scores between each cluster and the group size (# of cells) for each cluster. This information can be used together with mergeClusters to combine very similar or small clusters into bigger clusters.

Value

data.table

Examples

```
getClusterSimilarity(gobject)
```

getDendrogramSplits *getDendrogramSplits*

Description

Split dendrogram at each node and keep the leave (label) information..

Usage

```
getDendrogramSplits(gobject, expression_values = c("normalized",
  "scaled", "custom"), cluster_column, cor = c("pearson", "spearman"),
  distance = "ward.D", h = NULL, h_color = "red", show_dend = TRUE,
  verbose = TRUE)
```

Arguments

<code>gobject</code>	giotto object
<code>expression_values</code>	expression values to use
<code>cluster_column</code>	name of column to use for clusters
<code>cor</code>	correlation score to calculate distance
<code>distance</code>	distance method to use for hierarchical clustering
<code>h</code>	height of horizontal lines to plot
<code>h_color</code>	color of horizontal lines
<code>show_dend</code>	show dendrogram
<code>verbose</code>	be verbose

Details

Creates a data.table with three columns and each row represents a node in the dendrogram. For each node the height of the node is given together with the two subdendrograms. This information can be used to determine in a hierarchical manner differentially expressed marker genes at each node.

Value

data.table object

Examples

```
getDendrogramSplits(gobject)
```

<code>getDistinctColors</code>	<i>getDistinctColors</i>
--------------------------------	--------------------------

Description

Returns a number of distinct colors based on the RGB scale

Usage

```
getDistinctColors(n)
```

Arguments

<code>n</code>	number of colors wanted
----------------	-------------------------

Value

number of distinct colors

getGeneToGeneSelection

getGeneToGeneSelection

Description

Compute gene-gene enrichment scores.

Usage

```
getGeneToGeneSelection(CPGscore, selected_genes = NULL,
  specific_genes_1 = NULL, specific_genes_2 = NULL, min_cells = 5,
  min_pval = 0.05, min_spat_diff = 0.2, min_log2_fc = 0.5,
  direction = c("both", "up", "down"), fold_change_addendum = 0.1,
  verbose = TRUE)
```

Arguments

CPGscore	CPGscore, output from getCellProximityGeneScores()
selected_genes	select subset of genes
specific_genes_1	specific source genes (see details)
specific_genes_2	specific target genes (see details)
min_cells	min number of cells threshold
min_pval	p-value threshold
min_spat_diff	spatial difference threshold
min_log2_fc	log2 fold-change threshold
direction	up or downregulation or both
fold_change_addendum	constant to add when calculating log2 fold-change
verbose	verbose

Details

Give more details ...

Value

Gene to gene scores in data.table format

Examples

```
getGeneToGeneSelection(CPGscore)
```

```
get_cell_to_cell_sorted_name_conversion
    get_cell_to_cell_sorted_name_conversion
```

Description

creates unified cell-cell interaction names

Usage

```
get_cell_to_cell_sorted_name_conversion(all_cell_types)
```

Examples

```
get_cell_to_cell_sorted_name_conversion()
```

```
get_interaction_gene_enrichment
    get_interaction_gene_enrichment
```

Description

Computes gene enrichment between all interactions

Usage

```
get_interaction_gene_enrichment(spatial_network,
    unified_int_col = "unified_int", source_col = "source_clus",
    source_IDs = "from", neighb_col = "neighb_clus", neighb_IDs = "to",
    expression_matrix, cell_annotation, annotation_ID = "uniq_ID",
    cell_type_col, do_diff_test = T,
    exclude_selected_cells_from_test = T, verbose = T)
```

Examples

```
get_interaction_gene_enrichment()
```

```
get_specific_interaction_gene_enrichment
    get_specific_interaction_gene_enrichment
```

Description

Computes gene enrichment between specified interaction

Usage

```
get_specific_interaction_gene_enrichment(sub_spatial_network,
    source_col = "source_clus", source_IDs = "from",
    neighb_col = "neighb_clus", neighb_IDs = "to", expression_matrix,
    interaction_name = "to_specify", cell_annotation,
    annotation_ID = "uniq_ID", cell_type_col, do_diff_test = T,
    exclude_selected_cells_from_test = T)
```

Examples

```
get_specific_interaction_gene_enrichment()
```

giotto-class	<i>S4 giotto Class</i>
--------------	------------------------

Description

Framework of giotto object

Slots

```
raw_exprs raw expression counts
norm_expr normalized expression counts
norm_scaled_expr normalized and scaled expression counts
custom_expr custom normalized counts
spatial_locs spatial location coordinates for cells
cell_metadata metadata for cells
gene_metadata metadata for genes
cell_ID unique cell IDs
gene_ID unique gene IDs
spatial_network spatial network in data.table/data.frame format
spatial_grid spatial grid in data.table/data.frame format
dimension_reduction slot to save dimension reduction coordinates
nn_network nearest neighbor network in igraph format
parameters slot to save parameters that have been used
offset_file offset file used to stitch together image fields
```

iterCluster	<i>iterCluster</i>
-------------	--------------------

Description

cluster cells iteratively

Usage

```
iterCluster(gobject, cluster_method = c("leiden", "louvain_community",
    "louvain_multinet"), nr_rounds = 5,
    hvg_param = list(reverse_log_scale = T, difference_in_variance = 1,
    expression_values = "normalized"), hvg_min_perc_cells = 5,
    hvg_mean_expr_det = 1, use_all_genes_as_hvg = FALSE,
    min_nr_of_hvg = 5, pca_param = list(expression_values = "normalized",
    scale_unit = T), nn_param = list(dimensions_to_use = 1:20),
    k_neighbors = 20, resolution = 1, gamma = 1, omega = 1,
    python_path = NULL, nn_network_to_use = "sNN",
    network_name = "sNN.pca", name = "iter_clus",
    return_gobject = TRUE, ...)
```

Arguments

gobject	giotto object
cluster_method	clustering algorithm to use
nr_rounds	number of iterative rounds
hvg_param	parameters for calculateHVG
hvg_min_perc_cells	threshold for detection in min percentage of cells
hvg_mean_expr_det	threshold for mean expression level in cells with detection
use_all_genes_as_hvg	forces all genes to be HVG and to be used as input for PCA
min_nr_of_hvg	minimum number of HVG, or all genes will be used as input for PCA
pca_param	parameters for runPCA
nn_param	parameters for parameters for runPCA
k_neighbors	k for nn-network
resolution	resolution
gamma	gamma
omega	omega
python_path	python path to use for Leiden clustering
nn_network_to_use	NN network to use
network_name	NN network name
name	name of clustering
return_gobject	boolean: return giotto object (default = TRUE)
...	additional parameters

Details

Description of iterative clustering.

Value

giotto object appended with new cluster

Examples

```
iterCluster(gobject)
```

iterLeidenCluster	<i>iterLeidenCluster</i>
-------------------	--------------------------

Description

cluster cells iteratively

Usage

```
iterLeidenCluster(gobject, name = "iter_clus", nr_rounds = 5,
  hvg_param = list(reverse_log_scale = T, difference_in_variance = 1,
    expression_values = "normalized"), hvg_min_perc_cells = 5,
  hvg_mean_expr_det = 1, use_all_genes_as_hvg = FALSE,
  min_nr_of_hvg = 5, pca_param = list(expression_values = "normalized",
    scale_unit = T), nn_param = list(dimensions_to_use = 1:20),
  k_neighbors = 20, resolution = 1, n_iterations = 1000,
  python_path = NULL, nn_network_to_use = "sNN",
  network_name = "sNN.pca", return_gobject = TRUE, ...)
```

Arguments

gobject	giotto object
name	name of clustering
nr_rounds	number of iterative rounds
hvg_param	parameters for calculateHVG
hvg_min_perc_cells	threshold for detection in min percentage of cells
hvg_mean_expr_det	threshold for mean expression level in cells with detection
use_all_genes_as_hvg	forces all genes to be HVG and to be used as input for PCA
min_nr_of_hvg	minimum number of HVG, or all genes will be used as input for PCA
pca_param	parameters for runPCA
nn_param	parameters for parameters for runPCA
k_neighbors	k for nn-network
resolution	resolution for Leiden clustering
n_iterations	number of iterations for Leiden clustering

python_path python path to use for Leiden clustering
 nn_network_to_use NN network to use
 network_name NN network name
 return_gobject boolean: return giotto object (default = TRUE)
 ... additional parameters

Details

Description of iterative clustering.

Value

giotto object appended with new cluster

Examples

```
iterLeidenCluster(gobject)
```

iterLouvainCluster	<i>iterLouvainCluster</i>
--------------------	---------------------------

Description

cluster cells iteratively

Usage

```
iterLouvainCluster(gobject, version = c("community", "multinet"),
  nr_rounds = 5, hvg_param = list(reverse_log_scale = T,
  difference_in_variance = 1, expression_values = "normalized"),
  hvg_min_perc_cells = 5, hvg_mean_expr_det = 1,
  use_all_genes_as_hvg = FALSE, min_nr_of_hvg = 5,
  pca_param = list(expression_values = "normalized", scale_unit = T),
  nn_param = list(dimensions_to_use = 1:20), k_neighbors = 20,
  resolution = 1, gamma = 1, omega = 1, python_path = NULL,
  nn_network_to_use = "sNN", network_name = "sNN.pca",
  name = "iter_clus", return_gobject = TRUE, ...)
```

Arguments

gobject giotto object
 version louvain clustering algorithm to use
 nr_rounds number of iterative rounds
 hvg_param parameters for calculateHVG
 hvg_min_perc_cells threshold for detection in min percentage of cells
 hvg_mean_expr_det threshold for mean expression level in cells with detection

use_all_genes_as_hvg	forces all genes to be HVG and to be used as input for PCA
min_nr_of_hvg	minimum number of HVG, or all genes will be used as input for PCA
pca_param	parameters for runPCA
nn_param	parameters for parameters for runPCA
k_neighbors	k for nn-network
resolution	resolution
gamma	gamma
omega	omega
python_path	python path to use for Leiden clustering
nn_network_to_use	NN network to use
network_name	NN network name
name	name of clustering
return_gobject	boolean: return giotto object (default = TRUE)
...	additional parameters

Details

Description of iterative clustering.

Value

giotto object appended with new cluster

Examples

```
iterLouvainCluster(gobject)
```

```
iterLouvainCluster_community
      iterLouvainCluster_community
```

Description

cluster cells iteratively

Usage

```
iterLouvainCluster_community(gobject, nr_rounds = 5,
  hvg_param = list(reverse_log_scale = T, difference_in_variance = 1,
    expression_values = "normalized"), hvg_min_perc_cells = 5,
  hvg_mean_expr_det = 1, use_all_genes_as_hvg = FALSE,
  min_nr_of_hvg = 5, pca_param = list(expression_values = "normalized",
    scale_unit = T), nn_param = list(dimensions_to_use = 1:20),
  k_neighbors = 20, resolution = 1, python_path = NULL,
  nn_network_to_use = "sNN", network_name = "sNN.pca",
  name = "iter_clus", return_gobject = TRUE, ...)
```

Arguments

<code>gobject</code>	giotto object
<code>nr_rounds</code>	number of iterative rounds
<code>hvg_param</code>	parameters for calculateHVG
<code>hvg_min_perc_cells</code>	threshold for detection in min percentage of cells
<code>hvg_mean_expr_det</code>	threshold for mean expression level in cells with detection
<code>use_all_genes_as_hvg</code>	forces all genes to be HVG and to be used as input for PCA
<code>min_nr_of_hvg</code>	minimum number of HVG, or all genes will be used as input for PCA
<code>pca_param</code>	parameters for runPCA
<code>nn_param</code>	parameters for parameters for runPCA
<code>k_neighbors</code>	k for nn-network
<code>resolution</code>	resolution for Leiden clustering
<code>python_path</code>	python path to use for Leiden clustering
<code>nn_network_to_use</code>	NN network to use
<code>network_name</code>	NN network name
<code>name</code>	name of clustering
<code>return_gobject</code>	boolean: return giotto object (default = TRUE)
<code>...</code>	additional parameters

Details

Description of iterative clustering.

Value

giotto object appended with new cluster

Examples

```
iterLouvainCluster_community(gobject)
```

```
iterLouvainCluster_multinet
```

```
iterLouvainCluster_multinet
```

Description

cluster cells iteratively

Usage

```
iterLouvainCluster_multinet(gobject, nr_rounds = 5,
  hvg_param = list(reverse_log_scale = T, difference_in_variance = 1,
    expression_values = "normalized"), hvg_min_perc_cells = 5,
  hvg_mean_expr_det = 1, use_all_genes_as_hvg = FALSE,
  min_nr_of_hvg = 5, pca_param = list(expression_values = "normalized",
    scale_unit = T), nn_param = list(dimensions_to_use = 1:20),
  k_neighbors = 20, gamma = 1, omega = 1,
  nn_network_to_use = "sNN", network_name = "sNN.pca",
  name = "iter_clus", return_gobject = TRUE, ...)
```

Arguments

<code>gobject</code>	giotto object
<code>nr_rounds</code>	number of iterative rounds
<code>hvg_param</code>	parameters for calculateHVG
<code>hvg_min_perc_cells</code>	threshold for detection in min percentage of cells
<code>hvg_mean_expr_det</code>	threshold for mean expression level in cells with detection
<code>use_all_genes_as_hvg</code>	forces all genes to be HVG and to be used as input for PCA
<code>min_nr_of_hvg</code>	minimum number of HVG, or all genes will be used as input for PCA
<code>pca_param</code>	parameters for runPCA
<code>nn_param</code>	parameters for parameters for runPCA
<code>k_neighbors</code>	k for nn-network
<code>gamma</code>	gamma
<code>omega</code>	omega
<code>nn_network_to_use</code>	NN network to use
<code>network_name</code>	NN network name
<code>name</code>	name of clustering
<code>return_gobject</code>	boolean: return giotto object (default = TRUE)
<code>...</code>	additional parameters
<code>python_path</code>	python path to use for Leiden clustering

Details

Description of iterative clustering.

Value

giotto object appended with new cluster

Examples

```
iterLouvainCluster_multinet(gobject)
```

kmeans_binarize	<i>kmeans_binarize</i>
-----------------	------------------------

Description

create binarized scores using kmeans

Usage

```
kmeans_binarize(x, nstart = 3, iter.max = 10)
```

loadHMRf	<i>loadHMRf</i>
----------	-----------------

Description

load previous HMRf

Usage

```
loadHMRf(name_used = "test", output_folder_used, k_used = 10,  
betas_used, python_path_used)
```

Arguments

name_used	name of HMRf that was run
output_folder_used	output folder that was used
k_used	number of HMRf domains that was tested
betas_used	betas that were tested
python_path_used	python path that was used

Details

Description of HMRf parameters ...

Value

reloads a previous ran HMRf from doHMRf

Examples

```
loadHMRf(gobject)
```

make_simulated_network	<i>make_simulated_network</i>
------------------------	-------------------------------

Description

Simulate random network.

Usage

```
make_simulated_network(gobject, spatial_network_name = "spatial_network",
  cluster_column, number_of_simulations = 100)
```

Examples

```
make_simulated_network(gobject)
```

mergeClusters	<i>mergeClusters</i>
---------------	----------------------

Description

Merge selected clusters based on pairwise correlation scores and size of cluster.

Usage

```
mergeClusters(gobject, expression_values = c("normalized", "scaled",
  "custom"), cluster_column, cor = c("pearson", "spearman"),
  new_cluster_name = "merged_cluster", min_cor_score = 0.8,
  max_group_size = 20, force_min_group_size = 10,
  return_gobject = TRUE, verbose = TRUE)
```

Arguments

gobject	giotto object
expression_values	expression values to use
cluster_column	name of column to use for clusters
cor	correlation score to calculate distance
new_cluster_name	new name for merged clusters
min_cor_score	min correlation score to merge pairwise clusters
max_group_size	max cluster size that can be merged
force_min_group_size	size of clusters that will be merged with their most similar neighbor(s)
return_gobject	return giotto object
verbose	be verbose

Details

Merge selected clusters based on pairwise correlation scores and size of cluster. To avoid large clusters to merge the `max_group_size` can be lowered. Small clusters can be forcibly merged with the their most similar pairwise cluster by adjusting the `force_min_group_size` parameter. Clusters smaller than this value will be merged independent on the provided `min_cor_score` value. A giotto object is returned by default, if FALSE then the merging vector will be returned.

Value

Giotto object

Examples

```
mergeClusters(gobject)
```

mygini_fun	<i>mygini_fun</i>
------------	-------------------

Description

calculate gini coefficient

Usage

```
mygini_fun(x, weights = rep(1, length(x)))
```

Value

gini coefficient

nnDT_to_kNN	<i>nnDT_to_kNN</i>
-------------	--------------------

Description

Convert a nearest network data.table to a kNN object

Usage

```
nnDT_to_kNN(nnDT)
```

Arguments

nnDT nearest neighbor network in data.table format

Value

kNN object

node_clusters	<i>node_clusters</i>
---------------	----------------------

Description

Merge selected clusters based on pairwise correlation scores and size of cluster.

Usage

```
node_clusters(hclus_obj, verbose = TRUE)
```

Arguments

hclus_obj	hclus object
verbose	be verbose

Value

list of splitted dendrogram nodes from high to low node height

Examples

```
node_clusters(hclus_obj)
```

normalizeGiotto	<i>normalizeGiotto</i>
-----------------	------------------------

Description

normalize and/or scale expresion values of Giotto object

Usage

```
normalizeGiotto(gobject, norm_methods = c("standard", "osmFISH"),
  library_size_norm = TRUE, scalefactor = 6000, log_norm = TRUE,
  logbase = 2, scale_genes = T, scale_cells = T,
  scale_order = c("first_genes", "first_cells"), verbose = F)
```

Arguments

gobject	giotto object
norm_methods	normalization method to use
library_size_norm	normalize cells by library size
scalefactor	scale factor to use after library size normalization
log_norm	transform values to log-scale
logbase	log base to use to log normalize expression values
scale_genes	z-score genes over all cells
scale_cells	z-score cells over all genes
scale_order	order to scale genes and cells
verbose	be verbose

Details

Currently there are two 'methods' to normalize your raw counts data.

A. The standard method follows the standard protocol which can be adjusted using the provided parameters and follows the following order:

1. Data normalization for total library size and scaling by a custom scale-factor.
2. Log transformation of data.
3. Z-scoring of data by genes and/or cells.

B. The normalization method as provided by the osmFISH paper is also implemented:

1. First normalize genes, for each gene divide the counts by the total gene count and multiply by the total number of genes.
2. Next normalize cells, for each cell divide the normalized gene counts by the total counts per cell and multiply by the total number of cells.

This data will be saved in the Giotto slot for custom expression.

Value

giotto object

Examples

```
normalizeGiotto(gobject)
```

OR_function2

OR_function2

Description

calculate odds-ratio

Usage

```
OR_function2(A, B, C, D)
```

pDataDT

pDataDT

Description

show cell metadata

Usage

```
pDataDT(gobject)
```

Arguments

gobject

giotto object

Value

data.table

Examples

pDataDT(gobject)

plotCPGscores	<i>plotCPGscores</i>
---------------	----------------------

Description

Create heatmap from cell-cell proximity scores

Usage

```
plotCPGscores(CPGscores, selected_interactions = NULL,
  selected_genes = NULL, detail_plot = T, simple_plot = F,
  simple_plot_facet = c("interaction", "genes"),
  facet_scales = "fixed", facet_ncol = length(selected_genes),
  facet_nrow = length(selected_interactions), show_plot = F)
```

Arguments

CPGscores	CPGscores, output from getCellProximityGeneScores()
selected_interactions	interactions to show
selected_genes	genes to show
detail_plot	show detailed info in both interacting cell types
simple_plot	show a simplified plot
simple_plot_facet	facet on interactions or genes with simple plot
facet_scales	ggplot facet scales paramter
facet_ncol	ggplot facet ncol parameter
facet_nrow	ggplot facet nrow parameter
show_plot	show plot

Details

Give more details ...

Value

ggplot barplot

Examples

```
plotCPGscores(CPGscores)
```

plotGTGscores	<i>plotGTGscores</i>
---------------	----------------------

Description

Create heatmap from cell-cell proximity scores

Usage

```
plotGTGscores(GTGscore, selected_interactions = NULL,
  selected_gene_to_gene = NULL, detail_plot = T, simple_plot = F,
  simple_plot_facet = c("interaction", "genes"),
  facet_scales = "fixed", facet_ncol = length(selected_gene_to_gene),
  facet_nrow = length(selected_interactions), colors = c("blue",
  "red"), show_plot = F)
```

Arguments

GTGscore	GTGscore, output from getGeneToGeneScores()
selected_interactions	interactions to show
detail_plot	show detailed info in both interacting cell types
simple_plot	show a simplified plot
simple_plot_facet	facet on interactions or genes with simple plot
facet_scales	ggplot facet scales paramter
facet_ncol	ggplot facet ncol parameter
facet_nrow	ggplot facet nrow parameter
colors	vector with 2 colors to represent respectively all and selected cells
show_plot	show plot
selected_genes	genes to show

Details

Give more details ...

Value

ggplot barplot

Examples

```
plotGTGscores(GTGscore)
```

plotHeatmap

plotHeatmap

Description

creates order for clusters

Usage

```
plotHeatmap(gobject, expression_values = c("normalized", "scaled",
      "custom"), genes, cluster_column = NULL, cluster_order = c("size",
      "correlation", "custom"), cluster_custom_order = NULL,
      cluster_color_code = NULL, cluster_cor_method = "pearson",
      cluster_hclust_method = "ward.D", gene_order = c("custom",
      "correlation"), gene_custom_order = NULL,
      gene_cor_method = "pearson", gene_hclust_method = "complete",
      show_values = c("rescaled", "z-scaled", "original"),
      size_vertical_lines = 1.1, gradient_colors = c("blue", "yellow",
      "red"), gene_label_selection = NULL, axis_text_y_size = NULL,
      legend_nrows = 1, show_plot = TRUE)
```

Arguments

<code>gobject</code>	giotto object
<code>expression_values</code>	expression values to use
<code>genes</code>	genes to use
<code>cluster_column</code>	name of column to use for clusters
<code>cluster_order</code>	method to determine cluster order
<code>cluster_custom_order</code>	custom order for clusters
<code>cluster_color_code</code>	color code for clusters
<code>cluster_cor_method</code>	method for cluster correlation
<code>cluster_hclust_method</code>	method for hierarchical clustering of clusters
<code>gene_order</code>	method to determine gene order
<code>gene_custom_order</code>	custom order for genes
<code>gene_cor_method</code>	method for gene correlation
<code>gene_hclust_method</code>	method for hierarchical clustering of genes
<code>show_values</code>	which values to show on heatmap
<code>size_vertical_lines</code>	sizes for vertical lines

gradient_colors colors for heatmap gradient
 gene_label_selection subset of genes to show on y-axis
 axis_text_y_size size for y-axis text
 legend_nrows number of rows for the cluster legend

Details

Creates heatmap for genes and clusters.

Value

ggplot

Examples

```
plotHeatmap(gobject)
```

`plotly_axis_scale_2D` *plotly_axis_scale_2D*

Description

adjust the axis scale in 3D plotly plot

Usage

```
plotly_axis_scale_2D(cell_locations, sdimx = NULL, sdimy = NULL,
  mode = c("cube", "real", "custom"), custom_ratio = NULL)
```

Arguments

cell_locations spatial_loc in giotto object
 sdimx x axis of cell spatial location
 sdimy y axis of cell spatial location
 mode axis adjustment mode
 custom_ratio set the ratio artificially

Value

edges in spatial grid as data.table()

Examples

```
plotly_axis_scale_2D(gobject)
```

plotly_axis_scale_3D *plotly_axis_scale_3D*

Description

adjust the axis scale in 3D plotly plot

Usage

```
plotly_axis_scale_3D(cell_locations, sdimx = NULL, sdimy = NULL,
  sdimz = NULL, mode = c("cube", "real", "custom"),
  custom_ratio = NULL)
```

Arguments

cell_locations	spatial_loc in giotto object
sdimx	x axis of cell spatial location
sdimy	y axis of cell spatial location
sdimz	z axis of cell spatial location
mode	axis adjustment mode
custom_ratio	set the ratio artificially

Value

edges in spatial grid as data.table()

Examples

```
plotly_axis_scale_3D(gobject)
```

plotly_grid *plotly_grid*

Description

provide grid segment to draw in plot_ly()

Usage

```
plotly_grid(spatial_grid, x_start = "x_start", y_start = "y_start",
  x_end = "x_end", y_end = "y_end")
```

Arguments

spatial_grid	spatial_grid in giotto object
--------------	-------------------------------

Value

edges in spatial grid as data.table()

Examples

```
plotly_grid(gobject)
```

plotly_network	<i>plotly_network</i>
----------------	-----------------------

Description

provide network segment to draw in 3D plot_ly()

Usage

```
plotly_network(network, x = "sdimx_begin", y = "sdimy_begin",
  z = "sdimz_begin", x_end = "sdimx_end", y_end = "sdimy_end",
  z_end = "sdimz_end")
```

Arguments

gobject network in giotto object

Value

edges in network as data.table()

Examples

```
plotly_network(gobject)
```

plotMetaDataHeatmap	<i>plotMetaDataHeatmap</i>
---------------------	----------------------------

Description

creates order for clusters

Usage

```
plotMetaDataHeatmap(gobject, expression_values = c("normalized",
  "scaled", "custom"), metadata_cols = NULL, selected_genes = NULL,
  first_meta_col = NULL, second_meta_col = NULL,
  show_values = c("zscores", "original", "zscores_rescaled"),
  custom_cluster_order = NULL, clus_cor_method = "pearson",
  clus_cluster_method = "complete", custom_gene_order = NULL,
  gene_cor_method = "pearson", gene_cluster_method = "complete",
  midpoint = 0, x_text_size = 8, x_text_angle = 45,
  y_text_size = 8, strip_text_size = 8, show_plot = T)
```

Arguments

<code>gobject</code>	giotto object
<code>expression_values</code>	expression values to use
<code>metadata_cols</code>	annotation columns found in <code>pDataDT(gobject)</code>
<code>selected_genes</code>	subset of genes to use
<code>first_meta_col</code>	if more than 1 metadata column, select the x-axis factor
<code>second_meta_col</code>	if more than 1 metadata column, select the facetting factor
<code>show_values</code>	which values to show on heatmap
<code>custom_cluster_order</code>	custom cluster order (default = NULL)
<code>clus_cor_method</code>	correlation method for clusters
<code>clus_cluster_method</code>	hierarchical cluster method for the clusters
<code>custom_gene_order</code>	custom gene order (default = NULL)
<code>gene_cor_method</code>	correlation method for genes
<code>gene_cluster_method</code>	hierarchical cluster method for the genes
<code>midpoint</code>	midpoint of <code>show_values</code>
<code>x_text_size</code>	size of x-axis text
<code>x_text_angle</code>	angle of x-axis text
<code>y_text_size</code>	size of y-axis text
<code>strip_text_size</code>	size of strip text
<code>show_plot</code>	print plot (default = TRUE)

Details

Creates heatmap for average the average expression of selected genes in the different annotation groups

Value

ggplot or data.table

Examples

```
plotMetaDataHeatmap(gobject)
```

plotPCA	<i>plotPCA</i>
---------	----------------

Description

Short wrapper for PCA visualization

Usage

```
plotPCA(gobject, ...)
```

Arguments

gobject	giotto object
...	other parameters that are part of visDimPlot()

Details

Description of parameters.

Value

ggplot

See Also

[visDimPlot](#)

Examples

```
plotPCA(gobject)
```

plotTSNE	<i>plotTSNE</i>
----------	-----------------

Description

Short wrapper for tSNE visualization

Usage

```
plotTSNE(gobject, ...)
```

Arguments

gobject	giotto object
...	other parameters that are part of visDimPlot()

Details

Description of parameters.

Value

ggplot

See Also

[visDimPlot](#)

Examples

```
plotTSNE(gobject)
```

plotUMAP	<i>plotUMAP</i>
----------	-----------------

Description

Short wrapper for UMAP visualization

Usage

```
plotUMAP(gobject, ...)
```

Arguments

gobject	giotto object
...	other parameters that are part of visDimPlot()

Details

Description of parameters.

Value

ggplot

See Also

[visDimPlot](#)

Examples

```
plotUMAP(gobject)
```

```
plot_network_layer_ggplot  
    plot_network_layer_ggplot
```

Description

Visualize cells in network layer according to dimension reduction coordinates

Usage

```
plot_network_layer_ggplot(ggobject, annotated_network_DT,  
    edge_alpha = NULL, show_legend = T)
```

Arguments

annotated_network_DT	annotated network data.table of selected cells
edge_alpha	alpha of network edges
show_legend	show legend
gobject	giotto object

Details

Description of parameters.

Value

ggplot

Examples

```
plot_network_layer_ggplot(gobject)
```

```
plot_point_layer_ggplot  
    plot_point_layer_ggplot
```

Description

Visualize cells in point layer according to dimension reduction coordinates

Usage

```
plot_point_layer_ggplot(ggobject, annotated_DT_selected,
  annotated_DT_other, cell_color = NULL, color_as_factor = T,
  cell_color_code = NULL, select_cell_groups = NULL,
  select_cells = NULL, show_other_cells = T,
  other_cell_color = "lightgrey", other_point_size = 0.5,
  show_cluster_center = F, show_center_label = T,
  center_point_size = 4, center_point_border_col = "black",
  center_point_border_stroke = 0.1, label_size = 4,
  label_fontface = "bold", edge_alpha = NULL, point_size = 1,
  point_border_col = "black", point_border_stroke = 0.1,
  show_legend = T)
```

Arguments

annotated_DT_selected	annotated data.table of selected cells
annotated_DT_other	annotated data.table of not selected cells
cell_color	color for cells (see details)
color_as_factor	convert color column to factor
cell_color_code	named vector with colors
select_cell_groups	select subset of cells/clusters based on cell_color parameter
select_cells	select subset of cells based on cell IDs
show_other_cells	display not selected cells
other_cell_color	color of not selected cells
other_point_size	size of not selected cells
show_cluster_center	plot center of selected clusters
show_center_label	plot label of selected clusters
center_point_size	size of center points
label_size	size of labels
label_fontface	font of labels
edge_alpha	column to use for alpha of the edges
point_size	size of point (cell)
point_border_col	color of border around points
point_border_stroke	stroke size of border around points
show_legend	show legend
gobject	giotto object

Details

Description of parameters.

Value

ggplot

Examples

```
plot_point_layer_ggplot(gobject)
```

print.giotto	<i>print method for giotto class</i>
--------------	--------------------------------------

Description

print method for giotto class. Prints the chosen number of genes (rows) and cells (columns) from the raw count matrix. Also print the spatial locations for the chosen number of cells.

Usage

```
print.giotto(object, ...)
```

Arguments

nr_genes	number of genes (rows) to print
nr_cells	number of cells (columns) to print

rank_binarize	<i>rank_binarize</i>
---------------	----------------------

Description

create binarized scores using arbitrary rank of top genes

Usage

```
rank_binarize(x, max_rank = 200)
```

runPCA	<i>runPCA</i>
--------	---------------

Description

run PCA

Usage

```
runPCA(gobject, expression_values = c("normalized", "scaled", "custom"),
       reduction = c("cells", "genes"), name = "pca", genes_to_use = NULL,
       return_gobject = TRUE, scale_unit = F, ncp = 200, ...)
```

Arguments

gobject	giotto object
expression_values	expression values to use
reduction	cells or genes
name	arbitrary name for PCA run
genes_to_use	subset of genes to use for PCA
return_gobject	boolean: return giotto object (default = TRUE)
scale_unit	scale features before PCA
ncp	number of principal components to calculate
...	additional parameters for PCA

Details

Description of PCA steps...

Value

giotto object with updated PCA dimension reduction

Examples

```
runPCA(gobject)
```

runtSNE

*runtSNE***Description**

run tSNE

Usage

```
runtSNE(gobject, expression_values = c("normalized", "scaled", "custom"),
        reduction = c("cells", "genes"), dim_reduction_to_use = "pca",
        dim_reduction_name = "pca", dimensions_to_use = 1:10,
        name = "tsne", genes_to_use = NULL, return_gobject = TRUE,
        dims = 2, perplexity = 30, theta = 0.5, do_PCA_first = F,
        set_seed = T, seed_number = 1234, ...)
```

Arguments

<code>gobject</code>	giotto object
<code>expression_values</code>	expression values to use
<code>reduction</code>	cells or genes
<code>dim_reduction_to_use</code>	use another dimension reduction set as input
<code>dim_reduction_name</code>	name of dimension reduction set to use
<code>dimensions_to_use</code>	number of dimensions to use as input
<code>name</code>	arbitrary name for tSNE run
<code>genes_to_use</code>	if <code>dim_reduction_to_use = NULL</code> , which genes to use
<code>return_gobject</code>	boolean: return giotto object (default = TRUE)
<code>dims</code>	tSNE param: number of dimensions to return
<code>perplexity</code>	tSNE param: perplexity
<code>theta</code>	tSNE param: theta
<code>do_PCA_first</code>	tSNE param: do PCA before tSNE (default = FALSE)
<code>set_seed</code>	use of seed
<code>seed_number</code>	seed number to use
<code>...</code>	additional tSNE parameters

Details

Description of tSNE steps and params ...

Value

giotto object with updated tSNE dimension recuction

Examples

```
runtSNE(gobject)
```

runUMAP

*runUMAP***Description**

run UMAP

Usage

```
runUMAP(gobject, expression_values = c("normalized", "scaled", "custom"),
        reduction = c("cells", "genes"), dim_reduction_to_use = "pca",
        dim_reduction_name = "pca", dimensions_to_use = 1:10,
        name = "umap", genes_to_use = NULL, return_gobject = TRUE,
        n_neighbors = 40, n_components = 2, n_epochs = 400,
        min_dist = 0.01, n_threads = 1, spread = 5, set_seed = T,
        seed_number = 1234, ...)
```

Arguments

<code>gobject</code>	giotto object
<code>expression_values</code>	expression values to use
<code>reduction</code>	cells or genes
<code>dim_reduction_to_use</code>	use another dimension reduction set as input
<code>dim_reduction_name</code>	name of dimension reduction set to use
<code>dimensions_to_use</code>	number of dimensions to use as input
<code>name</code>	arbitrary name for UMAP run
<code>genes_to_use</code>	if <code>dim_reduction_to_use = NULL</code> , which genes to use
<code>return_gobject</code>	boolean: return giotto object (default = TRUE)
<code>n_neighbors</code>	UMAP param: number of neighbors
<code>n_components</code>	UMAP param: number of components
<code>n_epochs</code>	UMAP param: number of epochs
<code>min_dist</code>	UMAP param: minimum distance
<code>n_threads</code>	UMAP param: threads to use
<code>spread</code>	UMAP param: spread
<code>set_seed</code>	use of seed
<code>seed_number</code>	seed number to use
<code>...</code>	additional UMAP parameters

Details

Description of UMAP steps...

Value

giotto object with updated UMAP dimension reduction

Examples

```
runUMAP(gobject)
```

selectPatternGenes	<i>selectPatternGenes</i>
--------------------	---------------------------

Description

create a spatial grid

Usage

```
selectPatternGenes(spatPatObj, dimensions = 1:5, top_pos_genes = 10,  
  top_neg_genes = 10, min_pos_cor = 0.5, min_neg_cor = -0.5)
```

Arguments

spatPatObj	Output from detectSpatialPatterns
dimensions	dimensions to identify correlated genes for.
top_pos_genes	Top positively correlated genes.
top_neg_genes	Top negatively correlated genes.
min_pos_cor	Minimum positive correlation score to include a gene.
min_neg_cor	Minimum negative correlation score to include a gene.

Details

Description.

Value

ggplot

Examples

```
selectPatternGenes(gobject)
```

```
select_expression_values
      select_expression_values
```

Description

helper function to select expression values

Usage

```
select_expression_values(gobject, values)
```

Arguments

gobject	giotto object
values	expression values to extract

Value

expression matrix

```
show,giotto-method      show method for giotto class
```

Description

show method for giotto class

Usage

```
## S4 method for signature 'giotto'
show(object)
```

```
showClusterDendrogram  showClusterDendrogram
```

Description

Creates dendrogram based on identified clusters

Usage

```
showClusterDendrogram(gobject, expression_values = c("normalized",
  "scaled", "custom"), cluster_column, cor = c("pearson", "spearman"),
  distance = "ward.D", h = NULL, h_color = "red")
```

Arguments

<code>gobject</code>	giotto object
<code>expression_values</code>	expression values to use
<code>cluster_column</code>	name of column to use for clusters
<code>cor</code>	correlation score to calculate distance
<code>distance</code>	distance method to use for hierarchical clustering
<code>h</code>	height of horizontal lines to plot
<code>h_color</code>	color of horizontal lines

Details

Correlation dendrogram of selected clustering.

Value

ggplot

Examples

```
showClusterDendrogram(gobject)
```

<code>showClusterHeatmap</code>	<i>showClusterHeatmap</i>
---------------------------------	---------------------------

Description

Creates heatmap based on identified clusters

Usage

```
showClusterHeatmap(gobject, expression_values = c("normalized", "scaled",
  "custom"), cluster_column, cor = c("pearson", "spearman"),
  distance = "ward.D")
```

Arguments

<code>gobject</code>	giotto object
<code>expression_values</code>	expression values to use
<code>cluster_column</code>	name of column to use for clusters
<code>cor</code>	correlation score to calculate distance
<code>distance</code>	distance method to use for hierarchical clustering

Details

Correlation heatmap of selected clustering.

Value

ggplot

Examples

```
showClusterHeatmap(gobject)
```

showCPGscores	<i>showCPGscores</i>
---------------	----------------------

Description

visualize Cell Proximity Gene enrichment scores

Usage

```
showCPGscores(CPGscore, method = c("cell_barplot", "cell-cell",  
  "cell_sankey"), min_cells = 5, min_pval = 0.05,  
  min_spat_diff = 0.2, min_log2_fc = 0.5, direction = c("both", "up",  
  "down"), cell_color_code = NULL, show_plot = T, return_DT = F)
```

Arguments

CPGscore	CPGscore, output from getCellProximityGeneScores()
method	visualization method
min_cells	min number of cells threshold
min_pval	p-value threshold
min_spat_diff	spatial difference threshold
min_log2_fc	min log2 fold-change
direction	up or downregulation or both
cell_color_code	color code for cell types
show_plot	print plot
return_DT	return filtered data.table (boolean)

Details

Give more details ...

Value

Gene to gene scores in data.table format

Examples

```
showCPGscores(CPGscore)
```

showGeneExpressionProximityScore

showGeneExpressionProximityScore

Description

Create heatmap from cell-cell proximity scores

Usage

```
showGeneExpressionProximityScore(scores, selected_gene,
  sort_column = "diff_spat")
```

Arguments

scores	CPscore, output from getAverageCellProximityGeneScores()
selected_gene	gene to show
sort_column	column name to use for sorting

Details

Give more details ...

Value

ggplot barplot

Examples

```
showGeneExpressionProximityScore(scores)
```

showGTGscores

showGTGscores

Description

visualize Cell Proximity Gene enrichment scores

Usage

```
showGTGscores(GTGscore, method = c("cell_barplot", "cell-cell",
  "cell_sankey"), min_cells = 5, min_pval = 0.05,
min_spat_diff = 0.2, min_log2_fc = 0.5, direction = c("both", "up",
  "down"), cell_color_code = NULL, show_plot = T,
specific_genes_1 = NULL, specific_genes_2 = NULL,
first_cell_name = "ligand cell", second_cell_name = "receptor cell",
return_DT = F)
```

Arguments

method	visualization method
min_cells	min number of cells threshold
min_pval	p-value threshold
min_spat_diff	spatial difference threshold
min_log2_fc	log2 fold-change threshold
direction	up or downregulation or both
cell_color_code	color code for cell types
show_plot	print plot
specific_genes_1	subset of genes, matched with specific_genes_2
specific_genes_2	subset of genes, matched with specific_genes_1
first_cell_name	name for first cells
second_cell_name	name for second cells
CPGscore	CPGscore, output from getCellProximityGeneScores()

Details

Give more details ...

Value

ggplot

Examples

showGTGscores(CPGscore)

showIntExpressionProximityScore
showIntExpressionProximityScore

Description

Create heatmap from cell-cell proximity scores

Usage

showIntExpressionProximityScore(scores, selected_interaction,
sort_column = "diff_spat", show_enriched_n = 5,
show_depleted_n = 5)

Arguments

- scores scores, output from getAverageCellProximityGeneScores()
- selected_interaction interaction to show
- sort_column column name to use for sorting
- show_enriched_n show top enriched interactions
- show_depleted_n show top depleted interactions

Details

Give more details ...

Value

ggplot barplot

Examples

```
showIntExpressionProximityScore(scores)
```

showPattern	<i>showPattern</i>
-------------	--------------------

Description

create a spatial grid

Usage

```
showPattern(spatPatObj, dimension = 1, trim = c(0.02, 0.98),
background_color = "white", grid_border_color = "grey",
show_legend = T, plot_dim = 2, point_size = 1,
axis_scale = c("cube", "real", "custom"), custom_ratio = NULL,
x_ticks = NULL, y_ticks = NULL, z_ticks = NULL, show_plot = F)
```

Arguments

- spatPatObj Output from detectSpatialPatterns
- dimension dimension to plot
- trim Trim ends of the PC values.
- background_color background color for plot
- grid_border_color color for grid
- show_legend show legend of ggplot
- show_plot Show the plot.

Details

Description.

Value

ggplot

Examples

```
showPattern(gobject)
```

showPatternGenes	<i>showPatternGenes</i>
------------------	-------------------------

Description

create a spatial grid

Usage

```
showPatternGenes(spatPatObj, dimension = 1, top_pos_genes = 5,  
top_neg_genes = 5, point_size = 1, show_plot = F)
```

Arguments

spatPatObj	Output from detectSpatialPatterns
dimension	dimension to plot genes for.
top_pos_genes	Top positively correlated genes.
top_neg_genes	Top negatively correlated genes.
point_size	size of points
show_plot	Show the plot.

Details

Description.

Value

ggplot

Examples

```
showPatternGenes(gobject)
```

showProcessingSteps	<i>showProcessingSteps</i>
---------------------	----------------------------

Description

shows the sequential processing steps that were performed

Usage

```
showProcessingSteps(gobject)
```

Arguments

gobject	giotto object
---------	---------------

Value

list of processing steps and names

Examples

```
showProcessingSteps(gobject)
```

showTopGeneToGene	<i>showTopGeneToGene</i>
-------------------	--------------------------

Description

Show enriched/depleted gene-gene enrichments

Usage

```
showTopGeneToGene(GTGscore, top_interactions = 10,
  direction = c("increased", "decreased"), complement_data = T,
  subset_cell_ints = NULL, subset_genes = NULL)
```

Arguments

GTGscore	GTGscore, output from getGeneToGeneScores()
top_interactions	number of top gene-gene enrichments to show
direction	show top increased or decreased gene-gene enrichments
complement_data	include non-enriched gene-gene scores from other cell-cell interactions
subset_cell_ints	subset cell-cell interactions to show
subset_genes	subset genes to show

Details

[Give more details ...](#)

Value

ggplot barplot

Examples

```
showTopGeneToGene(scores)
```

signPCA	<i>signPCA</i>
---------	----------------

Description

identify significant prinicipal components (PCs)

Usage

```
signPCA(gobject, method = c("screeplot", "jackstraw"),
  expression_values = c("normalized", "scaled", "custom"),
  reduction = c("cells", "genes"), genes_to_use = NULL,
  scale_unit = T, ncp = 50, scree_labels = T, scree_ylim = c(0,
  10), jack_iter = 10, jack_threshold = 0.01, jack_verbose = T,
  show_plot = T, ...)
```

Arguments

<code>gobject</code>	giotto object
<code>method</code>	method to use to identify significant PCs
<code>expression_values</code>	expression values to use
<code>reduction</code>	cells or genes
<code>genes_to_use</code>	subset of genes to use for PCA
<code>scale_unit</code>	scale features before PCA
<code>ncp</code>	number of principal components to calculate
<code>scree_labels</code>	show labels on scree plot
<code>scree_ylim</code>	y-axis limits on scree plot
<code>jack_iter</code>	number of iterations for jackstraw
<code>jack_threshold</code>	p-value threshold to call a PC significant
<code>jack_verbose</code>	show progress of jackstraw method
<code>show_plot</code>	show plots
<code>...</code>	additional parameters for PCA

Details

Description of PCA steps...

Value

ggplot object for scree method and maxtrix of p-values for jackstraw

Examples

```
signPCA(gobject)
```

<i>Spatial_AEH</i>	<i>Spatial_AEH</i>
--------------------	--------------------

Description

calculate automatic expression histology with spatialDE method

Usage

```
Spatial_AEH(gobject = NULL, results = NULL, pattern_num = 5,
  l = 1.05, show_AEH = T, sdimx = NULL, sdimy = NULL,
  point_size = 3, point_alpha = 1, low_color = "blue",
  mid_color = "white", high_color = "red", midpoint = 0,
  python_path = NULL)
```

Arguments

<code>gobject</code>	Giotto object
<code>results</code>	output from spatial_DE
<code>pattern_num</code>	the number of gene expression patterns
<code>show_AEH</code>	show AEH plot
<code>python_path</code>	specify specific path to python if required

Details

Description.

Value

a list or a dataframe of SVs

Examples

```
Spatial_DE(gobject)
```

Spatial_DE	<i>Spatial_DE</i>
------------	-------------------

Description

calculate spatial variable genes with spatialDE method

Usage

```
Spatial_DE(gobject = NULL, show_plot = T, size = c(4, 2, 1),
  color = c("blue", "green", "red"), sig_alpha = 0.5,
  unsig_alpha = 0.5, python_path = NULL)
```

Arguments

gobject	Giotto object
show_plot	show FSV plot
python_path	specify specific path to python if required

Details

Description.

Value

a list or a dataframe of SVs

Examples

```
Spatial_DE(gobject)
```

specificCellCellcommunicationScores
<i>specificCellCellcommunicationScores</i>

Description

Specific Cell-Cell communication scores based on spatial expression of interacting cells

Usage

```
specificCellCellcommunicationScores(gobject,
  spatial_network_name = "spatial_network",
  cluster_column = "cell_types", random_iter = 100,
  cell_type_1 = "astrocyte", cell_type_2 = "endothelial", gene_set_1,
  gene_set_2, log2FC_addendum = 0.1, min_observations = 2,
  verbose = T)
```

Arguments

<code>gobject</code>	giotto object to use
<code>spatial_network_name</code>	spatial network to use for identifying interacting cells
<code>cluster_column</code>	cluster column with cell type information
<code>random_iter</code>	number of iterations
<code>cell_type_1</code>	first cell type
<code>cell_type_2</code>	second cell type
<code>gene_set_1</code>	first specific gene set from gene pairs
<code>gene_set_2</code>	second specific gene set from gene pairs
<code>log2FC_addendum</code>	addendum to add when calculating log2FC
<code>min_observations</code>	minimum number of interactions needed to be considered
<code>verbose</code>	verbose

Details

Details will follow.

Value

Cell-Cell communication scores for gene pairs based on spatial interaction

Examples

```
specificCellCellcommunicationScores(gobject)
```

```
split_dendrogram_in_two
      split_dendrogram_in_two
```

Description

Merge selected clusters based on pairwise correlation scores and size of cluster.

Usage

```
split_dendrogram_in_two(dend)
```

Arguments

<code>dend</code>	dendrogram object
-------------------	-------------------

Value

list of two dendrograms and height of node

Examples

```
split_dendrogram_in_two(dend)
```

stitchFieldCoordinates

stitchFieldCoordinates

Description

Helper function to stitch field coordinates together to form one complete picture

Usage

```
stitchFieldCoordinates(location_file, offset_file, cumulate_offset_x = F,
  cumulate_offset_y = F, field_col = "Field of View",
  X_coord_col = "X", Y_coord_col = "Y", reverse_final_x = F,
  reverse_final_y = T)
```

Arguments

location_file	location dataframe with X and Y coordinates
offset_file	dataframe that describes to offset for each field (see details)
cumulate_offset_x	(boolean) Do the x-axis offset values need to be cumulated?
cumulate_offset_y	(boolean) Do the y-axis offset values need to be cumulated?
field_col	column that indicates the field within the location_file
X_coord_col	column that indicates the x coordinates
Y_coord_col	column that indicates the x coordinates
reverse_final_x	(boolean) Do the final x coordinates need to be reversed?
reverse_final_y	(boolean) Do the final y coordinates need to be reversed?

Details

Describe how stitching works.

Value

Updated location dataframe with new X ['X_final'] and Y ['Y_final'] coordinates

Examples

```
stitchFieldCoordinates(gobject)
```

subClusterCells	<i>subClusterCells</i>
-----------------	------------------------

Description

subcluster cells

Usage

```
subClusterCells(gobject, name = "sub_clus",
  cluster_method = c("leiden", "louvain_community", "louvain_multinet"),
  cluster_column = NULL, selected_clusters = NULL,
  hvg_param = list(reverse_log_scale = T, difference_in_variance = 1,
    expression_values = "normalized"), hvg_min_perc_cells = 5,
  hvg_mean_expr_det = 1, use_all_genes_as_hvg = FALSE,
  min_nr_of_hvg = 5, pca_param = list(expression_values = "normalized",
    scale_unit = T), nn_param = list(dimensions_to_use = 1:20),
  k_neighbors = 10, resolution = 1, gamma = 1, omega = 1,
  python_path = NULL, nn_network_to_use = "sNN",
  network_name = "sNN.pca", return_gobject = TRUE, verbose = T, ...)
```

Arguments

gobject	giotto object
name	name for new clustering result
cluster_method	clustering method to use
cluster_column	cluster column to subcluster
selected_clusters	only do subclustering on these clusters
hvg_param	parameters for calculateHVG
hvg_min_perc_cells	threshold for detection in min percentage of cells
hvg_mean_expr_det	threshold for mean expression level in cells with detection
use_all_genes_as_hvg	forces all genes to be HVG and to be used as input for PCA
min_nr_of_hvg	minimum number of HVG, or all genes will be used as input for PCA
pca_param	parameters for runPCA
nn_param	parameters for parameters for createNearestNetwork
k_neighbors	number of k for createNearestNetwork
resolution	resolution
gamma	gamma
omega	omega
python_path	specify specific path to python if required
nn_network_to_use	type of NN network to use (kNN vs sNN)

network_name name of NN network to use
return_gobject boolean: return giotto object (default = TRUE)
verbose verbose
... additional parameters

Details

Description of Louvain clustering method.

Value

giotto object appended with new cluster

Examples

subClusterCells(gobject)

subsetGiotto	<i>subsetGiotto</i>
--------------	---------------------

Description

subsets Giotto object including previous calculations

Usage

subsetGiotto(gobject, cell_ids = NULL, gene_ids = NULL)

Arguments

gobject giotto object
cell_ids cell IDs to keep
gene_ids gene IDs to keep

Value

giotto object

Examples

subsetGiotto(gobject)

viewHMRFresults	<i>viewHMRFresults</i>
-----------------	------------------------

Description

View results from doHMRF.

Usage

```
viewHMRFresults(gobject, HMRFoutput, k = NULL, betas_to_view = NULL,
  third_dim = NULL, ...)
```

Arguments

gobject	giotto object
HMRFoutput	HMRF output from doHMRF
k	number of HMRF domains
betas_to_view	results from different betas that you want to view
...	paramters to visPlot()

Details

Description ...

Value

spatial plots with HMRF domains

See Also

[visPlot](#)

Examples

```
viewHMRFresults(gobject)
```

violinPlot	<i>violinPlot</i>
------------	-------------------

Description

Creates heatmap based on identified clusters

Usage

```
violinPlot(gobject, expression_values = c("normalized", "scaled",
  "custom"), genes, cluster_column, cluster_custom_order = NULL,
  color_violin = c("genes", "cluster"), cluster_color_code = NULL,
  strip_text = 7, axis_text_x_size = 10, axis_text_y_size = 6)
```

Arguments

gobject	giotto object
expression_values	expression values to use
genes	genes to plot
cluster_column	name of column to use for clusters
color_violin	color violinplots according to genes or clusters

Details

Correlation heatmap of clusters vs genes.

Value

ggplot

Examples

violinPlot(gobject)

visDimGenePlot	<i>visDimGenePlot</i>
----------------	-----------------------

Description

Visualize cells and gene expression according to dimension reduction coordinates

Usage

```
visDimGenePlot(gobject, expression_values = c("normalized", "scaled",
"custom"), genes = NULL, dim_reduction_to_use = "umap",
dim_reduction_name = "umap", dim1_to_use = 1, dim2_to_use = 2,
dim3_to_use = NULL, show_NN_network = F, nn_network_to_use = "sNN",
network_name = "sNN.pca", network_color = "lightgray",
edge_alpha = NULL, scale_alpha_with_expression = TRUE,
point_size = 1, genes_high_color = NULL, genes_mid_color = "white",
genes_low_color = "blue", point_border_col = "black",
point_border_stroke = 0.1, midpoint = 0, cow_n_col = 2,
cow_rel_h = 1, cow_rel_w = 1, cow_align = "h", show_legend = T,
plot_method = c("ggplot", "plotly"), show_plots = F)
```

Arguments

gobject	giotto object
expression_values	gene expression values to use
genes	genes to show
dim_reduction_to_use	dimension reduction to use

dim_reduction_name	dimension reduction name
dim1_to_use	dimension to use on x-axis
dim2_to_use	dimension to use on y-axis
dim3_to_use	dimension to use on z-axis
show_NN_network	show underlying NN network
nn_network_to_use	type of NN network to use (kNN vs sNN)
network_name	name of NN network to use, if show_NN_network = TRUE
edge_alpha	column to use for alpha of the edges
scale_alpha_with_expression	scale expression with ggplot alpha parameter
point_size	size of point (cell)
point_border_col	color of border around points
point_border_stroke	stroke size of border around points
midpoint	size of point (cell)
cow_n_col	cowplot param: how many columns
cow_rel_h	cowplot param: relative height
cow_rel_w	cowplot param: relative width
cow_align	cowplot param: how to align
show_legend	show legend
show_plots	show plots

Details

Description of parameters.

Value

ggplot

Examples

```
visDimGenePlot(gobject)
```

```
visDimGenePlot_2D_ggplot
      visDimGenePlot_2D_ggplot
```

Description

Visualize cells and gene expression according to dimension reduction coordinates

Usage

```
visDimGenePlot_2D_ggplot(gobject, expression_values = c("normalized",
  "scaled", "custom"), genes = NULL, dim_reduction_to_use = "umap",
  dim_reduction_name = "umap", dim1_to_use = 1, dim2_to_use = 2,
  show_NN_network = F, nn_network_to_use = "sNN",
  network_name = "sNN.pca", network_color = "lightgray",
  edge_alpha = NULL, scale_alpha_with_expression = TRUE,
  point_size = 1, genes_high_color = "red",
  genes_mid_color = "white", genes_low_color = "blue",
  point_border_col = "black", point_border_stroke = 0.1,
  midpoint = 0, cow_n_col = 2, cow_rel_h = 1, cow_rel_w = 1,
  cow_align = "h", show_legend = T, show_plots = F)
```

Arguments

<code>gobject</code>	giotto object
<code>expression_values</code>	gene expression values to use
<code>genes</code>	genes to show
<code>dim_reduction_to_use</code>	dimension reduction to use
<code>dim_reduction_name</code>	dimension reduction name
<code>dim1_to_use</code>	dimension to use on x-axis
<code>dim2_to_use</code>	dimension to use on y-axis
<code>show_NN_network</code>	show underlying NN network
<code>nn_network_to_use</code>	type of NN network to use (kNN vs sNN)
<code>network_name</code>	name of NN network to use, if <code>show_NN_network = TRUE</code>
<code>edge_alpha</code>	column to use for alpha of the edges
<code>scale_alpha_with_expression</code>	scale expression with ggplot alpha parameter
<code>point_size</code>	size of point (cell)
<code>point_border_col</code>	color of border around points
<code>point_border_stroke</code>	stroke size of border around points

midpoint	size of point (cell)
cow_n_col	cowplot param: how many columns
cow_rel_h	cowplot param: relative height
cow_rel_w	cowplot param: relative width
cow_align	cowplot param: how to align
show_legend	show legend
show_plots	show plots

Details

Description of parameters.

Value

ggplot

Examples

```
visDimGenePlot_2D_ggplot(gobject)
```

```
visDimGenePlot_3D_plotly
```

```
visDimGenePlot_3D_plotly
```

Description

Visualize cells and gene expression according to dimension reduction coordinates

Usage

```
visDimGenePlot_3D_plotly(gobject, expression_values = c("normalized",
  "scaled", "custom"), genes = NULL, dim_reduction_to_use = "umap",
  dim_reduction_name = "umap", dim1_to_use = 1, dim2_to_use = 2,
  dim3_to_use = 3, show_NN_network = F, nn_network_to_use = "sNN",
  network_name = "sNN.pca", network_color = "lightgray",
  edge_alpha = NULL, point_size = 1, genes_high_color = NULL,
  genes_mid_color = "white", genes_low_color = "blue",
  show_legend = T, show_plots = F)
```

Arguments

gobject	giotto object
expression_values	gene expression values to use
genes	genes to show
dim_reduction_to_use	dimension reduction to use
dim_reduction_name	dimension reduction name

dim1_to_use	dimension to use on x-axis
dim2_to_use	dimension to use on y-axis
dim3_to_use	dimension to use on z-axis
show_NN_network	show underlying NN network
nn_network_to_use	type of NN network to use (kNN vs sNN)
network_name	name of NN network to use, if show_NN_network = TRUE
edge_alpha	column to use for alpha of the edges
point_size	size of point (cell)
show_legend	show legend
show_plots	show plots

Details

Description of parameters.

Value

ggplot

Examples

visDimGenePlot_3D_plotly(gobject)

visDimPlot	visDimPlot
------------	------------

Description

Visualize cells according to dimension reduction coordinates

Usage

```
visDimPlot(gobject, dim_reduction_to_use = "umap",
  dim_reduction_name = "umap", dim1_to_use = 1, dim2_to_use = 2,
  dim3_to_use = NULL, show_NN_network = F, nn_network_to_use = "sNN",
  network_name = "sNN.pca", cell_color = NULL, color_as_factor = T,
  cell_color_code = NULL, select_cell_groups = NULL,
  select_cells = NULL, show_other_cells = T,
  other_cell_color = "lightgrey", other_point_size = 0.5,
  show_cluster_center = F, show_center_label = T,
  center_point_size = 4, center_point_border_col = "black",
  center_point_border_stroke = 0.1, label_size = 4,
  label_fontface = "bold", edge_alpha = NULL, point_size = 3,
  point_border_col = "black", point_border_stroke = 0.1,
  plot_method = c("ggplot", "plotly"), show_legend = T)
```

Arguments

<code>gobject</code>	giotto object
<code>dim_reduction_to_use</code>	dimension reduction to use
<code>dim_reduction_name</code>	dimension reduction name
<code>dim1_to_use</code>	dimension to use on x-axis
<code>dim2_to_use</code>	dimension to use on y-axis
<code>dim3_to_use</code>	dimension to use on z-axis
<code>show_NN_network</code>	show underlying NN network
<code>nn_network_to_use</code>	type of NN network to use (kNN vs sNN)
<code>network_name</code>	name of NN network to use, if <code>show_NN_network = TRUE</code>
<code>cell_color</code>	color for cells (see details)
<code>color_as_factor</code>	convert color column to factor
<code>cell_color_code</code>	named vector with colors
<code>show_cluster_center</code>	plot center of selected clusters
<code>show_center_label</code>	plot label of selected clusters
<code>center_point_size</code>	size of center points
<code>label_size</code>	size of labels
<code>label_fontface</code>	font of labels
<code>edge_alpha</code>	column to use for alpha of the edges
<code>point_size</code>	size of point (cell)
<code>point_border_col</code>	color of border around points
<code>point_border_stroke</code>	stroke size of border around points
<code>show_legend</code>	show legend

Details

Description of parameters.

Value

ggplot or plotly

Examples

```
visDimPlot(gobject)
```

visDimPlot_2D_ggplot *visDimPlot_2D_ggplot*

Description

Visualize cells according to dimension reduction coordinates

Usage

```
visDimPlot_2D_ggplot(gobject, dim_reduction_to_use = "umap",
  dim_reduction_name = "umap", dim1_to_use = 1, dim2_to_use = 2,
  show_NN_network = F, nn_network_to_use = "sNN",
  network_name = "sNN.pca", cell_color = NULL, color_as_factor = T,
  cell_color_code = NULL, select_cell_groups = NULL,
  select_cells = NULL, show_other_cells = T,
  other_cell_color = "lightgrey", other_point_size = 0.5,
  show_cluster_center = F, show_center_label = T,
  center_point_size = 4, center_point_border_col = "black",
  center_point_border_stroke = 0.1, label_size = 4,
  label_fontface = "bold", edge_alpha = NULL, point_size = 1,
  point_border_col = "black", point_border_stroke = 0.1,
  show_legend = T)
```

Arguments

gobject	giotto object
dim_reduction_to_use	dimension reduction to use
dim_reduction_name	dimension reduction name
dim1_to_use	dimension to use on x-axis
dim2_to_use	dimension to use on y-axis
show_NN_network	show underlying NN network
nn_network_to_use	type of NN network to use (kNN vs sNN)
network_name	name of NN network to use, if show_NN_network = TRUE
cell_color	color for cells (see details)
color_as_factor	convert color column to factor
cell_color_code	named vector with colors
select_cell_groups	select subset of cells/clusters based on cell_color parameter
select_cells	select subset of cells based on cell IDs
show_other_cells	display not selected cells

other_cell_color	color of not selected cells
other_point_size	size of not selected cells
show_cluster_center	plot center of selected clusters
show_center_label	plot label of selected clusters
center_point_size	size of center points
label_size	size of labels
label_fontface	font of labels
edge_alpha	column to use for alpha of the edges
point_size	size of point (cell)
point_border_col	color of border around points
point_border_stroke	stroke size of border around points
show_legend	show legend

Details

Description of parameters.

Value

ggplot

Examples

```
visDimPlot_2D_ggplot(gobject)
```

visDimPlot_2D_plotly	<i>visDimPlot_2D_plotly</i>
----------------------	-----------------------------

Description

Visualize cells according to dimension reduction coordinates

Usage

```
visDimPlot_2D_plotly(gobject, dim_reduction_to_use = "umap",
  dim_reduction_name = "umap", dim1_to_use = 1, dim2_to_use = 2,
  show_NN_network = F, nn_network_to_use = "sNN",
  network_name = "sNN.pca", cell_color = NULL, color_as_factor = T,
  cell_color_code = NULL, show_cluster_center = F,
  show_center_label = T, center_point_size = 4, label_size = 4,
  edge_alpha = NULL, point_size = 5, show_legend = T)
```

Arguments

<code>gobject</code>	giotto object
<code>dim_reduction_to_use</code>	dimension reduction to use
<code>dim_reduction_name</code>	dimension reduction name
<code>dim1_to_use</code>	dimension to use on x-axis
<code>dim2_to_use</code>	dimension to use on y-axis
<code>show_NN_network</code>	show underlying NN network
<code>nn_network_to_use</code>	type of NN network to use (kNN vs sNN)
<code>network_name</code>	name of NN network to use, if <code>show_NN_network = TRUE</code>
<code>cell_color</code>	color for cells (see details)
<code>color_as_factor</code>	convert color column to factor
<code>cell_color_code</code>	named vector with colors
<code>show_cluster_center</code>	plot center of selected clusters
<code>show_center_label</code>	plot label of selected clusters
<code>center_point_size</code>	size of center points
<code>label_size</code>	size of labels
<code>edge_alpha</code>	column to use for alpha of the edges
<code>point_size</code>	size of point (cell)
<code>show_legend</code>	show legend

Details

Description of parameters.

Value

plotly

Examples

```
visDimPlot_2D_plotly(gobject)
```

visDimPlot_3D_plotly *visDimPlot_3D_plotly*

Description

Visualize cells according to dimension reduction coordinates

Usage

```
visDimPlot_3D_plotly(gobject, dim_reduction_to_use = "umap",
  dim_reduction_name = "umap", dim1_to_use = 1, dim2_to_use = 2,
  dim3_to_use = 3, show_NN_network = F, nn_network_to_use = "sNN",
  network_name = "sNN.pca", cell_color = NULL, color_as_factor = T,
  cell_color_code = NULL, show_cluster_center = F,
  show_center_label = T, center_point_size = 4, label_size = 4,
  edge_alpha = NULL, point_size = 1, show_legend = T)
```

Arguments

gobject	giotto object
dim_reduction_to_use	dimension reduction to use
dim_reduction_name	dimension reduction name
dim1_to_use	dimension to use on x-axis
dim2_to_use	dimension to use on y-axis
dim3_to_use	dimension to use on z-axis
show_NN_network	show underlying NN network
nn_network_to_use	type of NN network to use (kNN vs sNN)
network_name	name of NN network to use, if show_NN_network = TRUE
cell_color	color for cells (see details)
color_as_factor	convert color column to factor
cell_color_code	named vector with colors
show_cluster_center	plot center of selected clusters
show_center_label	plot label of selected clusters
center_point_size	size of center points
label_size	size of labels
edge_alpha	column to use for alpha of the edges
point_size	size of point (cell)
show_legend	show legend

Details

Description of parameters.

Value

plotly

Examples

```
visDimPlot_3D_plotly(gobject)
```

visForceLayoutPlot	<i>visForceLayoutPlot</i>
--------------------	---------------------------

Description

Visualize cells according to forced layout algorithm coordinates

Usage

```
visForceLayoutPlot(gobject, nn_network_to_use = "sNN",
  network_name = "sNN.pca", layout_name = "layout", dim1_to_use = 1,
  dim2_to_use = 2, show_NN_network = T, cell_color = NULL,
  color_as_factor = F, cell_color_code = NULL, edge_alpha = NULL,
  point_size = 1, point_border_col = "black",
  point_border_stroke = 0.1, show_legend = T)
```

Arguments

gobject	giotto object
nn_network_to_use	type of NN network to use (kNN vs sNN)
network_name	NN network to use
layout_name	name of layout to use
dim1_to_use	dimension to use on x-axis
dim2_to_use	dimension to use on y-axis
show_NN_network	show underlying NN network
cell_color	color for cells (see details)
color_as_factor	convert color column to factor
cell_color_code	named vector with colors
edge_alpha	column to use for alpha of the edges
point_size	size of point (cell)
point_border_col	color of border around points
point_border_stroke	stroke size of border around points
show_legend	show legend

Details

Description of parameters.

Value

ggplot

Examples

```
visForceLayoutPlot(gobject)
```

visGenePlot	<i>visGenePlot</i>
-------------	--------------------

Description

Visualize cells and gene expression according to spatial coordinates

Usage

```
visGenePlot(gobject, expression_values = c("normalized", "scaled",
      "custom"), genes, genes_high_color = NULL, genes_mid_color = "white",
      genes_low_color = "blue", show_network = F, network_color = NULL,
      spatial_network_name = "spatial_network", edge_alpha = NULL,
      show_grid = F, grid_color = NULL,
      spatial_grid_name = "spatial_grid", midpoint = 0,
      scale_alpha_with_expression = TRUE, point_size = 1,
      point_border_col = "black", point_border_stroke = 0.1,
      show_legend = T, cow_n_col = 2, cow_rel_h = 1, cow_rel_w = 1,
      cow_align = "h", axis_scale = c("cube", "real", "custom"),
      custom_ratio = NULL, x_ticks = NULL, y_ticks = NULL,
      z_ticks = NULL, plot_method = c("ggplot", "plotly"),
      show_plots = F)
```

Arguments

<code>gobject</code>	giotto object
<code>expression_values</code>	gene expression values to use
<code>genes</code>	genes to show
<code>genes_high_color</code>	color represents high gene expression
<code>genes_mid_color</code>	color represents middle gene expression
<code>genes_low_color</code>	color represents low gene expression
<code>show_network</code>	show underlying spatial network
<code>network_color</code>	color of spatial network
<code>spatial_network_name</code>	name of spatial network to use

show_grid	show spatial grid
grid_color	color of spatial grid
spatial_grid_name	name of spatial grid to use
midpoint	expression midpoint
scale_alpha_with_expression	scale expression with ggplot alpha parameter
point_size	size of point (cell)
point_border_col	color of border around points
point_border_stroke	stroke size of border around points
show_legend	show legend
cow_n_col	cowplot param: how many columns
cow_rel_h	cowplot param: relative height
cow_rel_w	cowplot param: relative width
cow_align	cowplot param: how to align
axis_scale	three mode to adjust axis scale
x_ticks	number of ticks on x axis
y_ticks	number of ticks on y axis
z_ticks	number of ticks on z axis
plot_method	two methods of plot
show_plots	show plots

Details

Description of parameters.

Value

ggplot or plotly

Examples

```
visGenePlot(gobject)
```

```
visGenePlot_2D_ggplot  visGenePlot_2D_ggplot
```

Description

Visualize cells and gene expression according to spatial coordinates

Usage

```
visGenePlot_2D_ggplot(gobject, expression_values = c("normalized",
  "scaled", "custom"), genes, genes_high_color = "darkred",
  genes_mid_color = "white", genes_low_color = "darkblue",
  show_network = F, network_color = NULL,
  spatial_network_name = "spatial_network", edge_alpha = NULL,
  show_grid = F, grid_color = NULL,
  spatial_grid_name = "spatial_grid", midpoint = 0,
  scale_alpha_with_expression = TRUE, point_size = 1,
  point_border_col = "black", point_border_stroke = 0.1,
  show_legend = T, cow_n_col = 2, cow_rel_h = 1, cow_rel_w = 1,
  cow_align = "h", show_plots = F)
```

Arguments

<code>gobject</code>	giotto object
<code>expression_values</code>	gene expression values to use
<code>genes</code>	genes to show
<code>genes_high_color</code>	color represents high gene expression
<code>genes_mid_color</code>	color represents middle gene expression
<code>genes_low_color</code>	color represents low gene expression
<code>show_network</code>	show underlying spatial network
<code>network_color</code>	color of spatial network
<code>spatial_network_name</code>	name of spatial network to use
<code>show_grid</code>	show spatial grid
<code>grid_color</code>	color of spatial grid
<code>spatial_grid_name</code>	name of spatial grid to use
<code>midpoint</code>	expression midpoint
<code>scale_alpha_with_expression</code>	scale expression with ggplot alpha parameter
<code>point_size</code>	size of point (cell)
<code>point_border_col</code>	color of border around points
<code>point_border_stroke</code>	stroke size of border around points
<code>show_legend</code>	show legend
<code>cow_n_col</code>	cowplot param: how many columns
<code>cow_rel_h</code>	cowplot param: relative height
<code>cow_rel_w</code>	cowplot param: relative width
<code>cow_align</code>	cowplot param: how to align
<code>show_plots</code>	show plots

Details

Description of parameters.

Value

ggplot

Examples

visGenePlot_2D_ggplot(gobject)

visGenePlot_3D_plotly *visGenePlot_3D_plotly*

Description

Visualize cells and gene expression according to spatial coordinates

Usage

```
visGenePlot_3D_plotly(gobject, expression_values = c("normalized",
  "scaled", "custom"), genes, show_network = F, network_color = NULL,
  spatial_network_name = "spatial_network", edge_alpha = NULL,
  show_grid = F, genes_high_color = NULL, genes_mid_color = "white",
  genes_low_color = "blue", spatial_grid_name = "spatial_grid",
  point_size = 1, show_legend = T, axis_scale = c("cube", "real",
  "custom"), custom_ratio = NULL, x_ticks = NULL, y_ticks = NULL,
  z_ticks = NULL, show_plots = F)
```

Arguments

gobject	giotto object
expression_values	gene expression values to use
genes	genes to show
show_network	show underlying spatial network
network_color	color of spatial network
spatial_network_name	name of spatial network to use
show_grid	show spatial grid
genes_high_color	color represents high gene expression
genes_mid_color	color represents middle gene expression
genes_low_color	color represents low gene expression
spatial_grid_name	name of spatial grid to use

point_size	size of point (cell)
show_legend	show legend
axis_scale	three mode to adjust axis scale
x_ticks	number of ticks on x axis
y_ticks	number of ticks on y axis
z_ticks	number of ticks on z axis
show_plots	show plots
grid_color	color of spatial grid
cow_n_col	cowplot param: how many columns
cow_rel_h	cowplot param: relative height
cow_rel_w	cowplot param: relative width
cow_align	cowplot param: how to align

Details

Description of parameters.

Value

plotly

Examples

```
visGenePlot_3D_plotly(gobject)
```

visPlot	<i>visPlot</i>
---------	----------------

Description

Visualize cells according to spatial coordinates

Usage

```
visPlot(gobject, sdimx = NULL, sdimy = NULL, sdimz = NULL,
  point_size = 3, point_border_col = "black",
  point_border_stroke = 0.1, cell_color = NULL,
  cell_color_code = NULL, color_as_factor = T,
  select_cell_groups = NULL, select_cells = NULL,
  show_other_cells = T, other_cell_color = "lightgrey",
  show_network = F, network_color = NULL, network_alpha = 1,
  other_cells_alpha = 0.1, spatial_network_name = "spatial_network",
  show_grid = F, grid_color = NULL, grid_alpha = 1,
  spatial_grid_name = "spatial_grid", coord_fix_ratio = 0.6,
  title = "", show_legend = T, axis_scale = c("cube", "real",
  "custom"), custom_ratio = NULL, x_ticks = NULL, y_ticks = NULL,
  z_ticks = NULL, plot_method = c("ggplot", "plotly"), show_plot = F)
```

Arguments

<code>gobject</code>	giotto object
<code>sdimx</code>	x-axis dimension name (default = 'sdimx')
<code>sdimy</code>	y-axis dimension name (default = 'sdimy')
<code>sdimz</code>	z-axis dimension name (default = 'sdimz')
<code>point_size</code>	size of point (cell)
<code>point_border_col</code>	color of border around points
<code>point_border_stroke</code>	stroke size of border around points
<code>cell_color</code>	color for cells (see details)
<code>cell_color_code</code>	named vector with colors
<code>color_as_factor</code>	convert color column to factor
<code>select_cell_groups</code>	select subset of cells/clusters based on <code>cell_color</code> parameter
<code>select_cells</code>	select subset of cells based on cell IDs
<code>show_other_cells</code>	display not selected cells
<code>other_cell_color</code>	color of not selected cells
<code>show_network</code>	show underlying spatial network
<code>network_color</code>	color of spatial network
<code>spatial_network_name</code>	name of spatial network to use
<code>show_grid</code>	show spatial grid
<code>grid_color</code>	color of spatial grid
<code>spatial_grid_name</code>	name of spatial grid to use
<code>coord_fix_ratio</code>	fix ratio between x and y-axis
<code>title</code>	title of plot
<code>show_legend</code>	show legend
<code>show_plot</code>	show plot

Details

Description of parameters.

Value

ggplot

Examples

```
visPlot(gobject)
```

visPlot_2D_ggplot	<i>visPlot_2D_ggplot</i>
-------------------	--------------------------

Description

Visualize cells according to spatial coordinates

Usage

```
visPlot_2D_ggplot(gobject, sdinx = NULL, sdimy = NULL,
  point_size = 3, point_border_col = "black",
  point_border_stroke = 0.1, cell_color = NULL,
  cell_color_code = NULL, color_as_factor = T,
  select_cell_groups = NULL, select_cells = NULL,
  show_other_cells = T, other_cell_color = "lightgrey",
  show_network = F, network_color = NULL, network_alpha = 1,
  other_cells_alpha = 0.1, spatial_network_name = "spatial_network",
  show_grid = F, grid_color = NULL,
  spatial_grid_name = "spatial_grid", coord_fix_ratio = 0.6,
  title = "", show_legend = T, axis_scale = c("cube", "real",
  "custom"), custom_ratio = NULL, x_ticks = NULL, y_ticks = NULL,
  z_ticks = NULL, show_plot = F)
```

Arguments

<code>gobject</code>	giotto object
<code>sdinx</code>	x-axis dimension name (default = 'sdinx')
<code>sdimy</code>	y-axis dimension name (default = 'sdimy')
<code>point_size</code>	size of point (cell)
<code>point_border_col</code>	color of border around points
<code>point_border_stroke</code>	stroke size of border around points
<code>cell_color</code>	color for cells (see details)
<code>cell_color_code</code>	named vector with colors
<code>color_as_factor</code>	convert color column to factor
<code>select_cell_groups</code>	select subset of cells/clusters based on cell_color parameter
<code>select_cells</code>	select subset of cells based on cell IDs
<code>show_other_cells</code>	display not selected cells
<code>other_cell_color</code>	color of not selected cells
<code>show_network</code>	show underlying spatial network
<code>network_color</code>	color of spatial network

spatial_network_name	name of spatial network to use
show_grid	show spatial grid
grid_color	color of spatial grid
spatial_grid_name	name of spatial grid to use
coord_fix_ratio	fix ratio between x and y-axis
title	title of plot
show_legend	show legend
show_plot	show plot

Details

Description of parameters.

Value

ggplot

Examples

visPlot_2D_ggplot(gobject)

visPlot_2D_plotly	<i>visPlot_2D_plotly</i>
-------------------	--------------------------

Description

Visualize cells according to spatial coordinates

Usage

```
visPlot_2D_plotly(gobject, sdinx = NULL, sdimy = NULL,
  point_size = 3, cell_color = NULL, cell_color_code = NULL,
  color_as_factor = T, select_cell_groups = NULL,
  select_cells = NULL, show_other_cells = T, show_network = F,
  network_color = "lightgray", network_alpha = 1,
  other_cells_alpha = 0.5, spatial_network_name = "spatial_network",
  show_grid = F, grid_color = NULL, grid_alpha = 1,
  spatial_grid_name = "spatial_grid", show_legend = T,
  axis_scale = c("cube", "real", "custom"), custom_ratio = NULL,
  x_ticks = NULL, y_ticks = NULL, show_plot = F)
```

Arguments

<code>gobject</code>	giotto object
<code>sdimx</code>	x-axis dimension name (default = 'sdimx')
<code>sdimy</code>	y-axis dimension name (default = 'sdimy')
<code>point_size</code>	size of point (cell)
<code>cell_color</code>	color for cells (see details)
<code>cell_color_code</code>	named vector with colors
<code>color_as_factor</code>	convert color column to factor
<code>select_cell_groups</code>	select a subset of the groups from <code>cell_color</code>
<code>show_network</code>	show underlying spatial network
<code>network_color</code>	color of spatial network
<code>spatial_network_name</code>	name of spatial network to use
<code>show_grid</code>	show spatial grid
<code>grid_color</code>	color of spatial grid
<code>grid_alpha</code>	alpha of spatial grid
<code>spatial_grid_name</code>	name of spatial grid to use
<code>show_legend</code>	show legend
<code>show_plot</code>	show plot

Details

Description of parameters.

Value

plotly

Examples

```
visPlot_2D_plotly(gobject)
```

<code>visPlot_3D_plotly</code>	<i>visPlot_3D_plotly</i>
--------------------------------	--------------------------

Description

Visualize cells according to spatial coordinates

Usage

```
visPlot_3D_plotly(gobject, sdinx = NULL, sdimy = NULL, sdimz = NULL,
  point_size = 3, point_border_col = "black",
  point_border_stroke = 0.1, cell_color = NULL,
  cell_color_code = NULL, color_as_factor = T,
  select_cell_groups = NULL, select_cells = NULL,
  show_other_cells = T, show_network = F, network_color = NULL,
  network_alpha = 1, other_cells_alpha = 0.1,
  spatial_network_name = "spatial_network", show_grid = F,
  grid_color = NULL, spatial_grid_name = "spatial_grid",
  coord_fix_ratio = 0.6, title = "", show_legend = T,
  axis_scale = c("cube", "real", "custom"), custom_ratio = NULL,
  x_ticks = NULL, y_ticks = NULL, z_ticks = NULL, show_plot = F)
```

Arguments

<code>gobject</code>	giotto object
<code>sdinx</code>	x-axis dimension name (default = 'sdinx')
<code>sdimy</code>	y-axis dimension name (default = 'sdimy')
<code>sdinz</code>	z-axis dimension name (default = 'sdinz')
<code>point_size</code>	size of point (cell)
<code>point_border_col</code>	color of border around points
<code>point_border_stroke</code>	stroke size of border around points
<code>cell_color</code>	color for cells (see details)
<code>cell_color_code</code>	named vector with colors
<code>color_as_factor</code>	convert color column to factor
<code>select_cell_groups</code>	select a subset of the groups from <code>cell_color</code>
<code>show_network</code>	show underlying spatial network
<code>network_color</code>	color of spatial network
<code>spatial_network_name</code>	name of spatial network to use
<code>show_grid</code>	show spatial grid
<code>grid_color</code>	color of spatial grid
<code>spatial_grid_name</code>	name of spatial grid to use
<code>coord_fix_ratio</code>	fix ratio between x and y-axis
<code>title</code>	title of plot
<code>show_legend</code>	show legend
<code>show_plot</code>	show plot

Details

Description of parameters.

Value

ggplot

Examples

```
visPlot_3D_plotly(gobject)
```

visSpatDimGenePlot	<i>visSpatDimGenePlot</i>
--------------------	---------------------------

Description

integration of visSpatDimGenePlot_2D(ggplot) and visSpatDimGenePlot_3D(plotly)

Usage

```
visSpatDimGenePlot(gobject, plot_method = c("ggplot", "plotly"),
  expression_values = c("normalized", "scaled", "custom"),
  plot_alignment = c("horizontal", "vertical"),
  dim_reduction_to_use = "umap", dim_reduction_name = "umap",
  dim1_to_use = 1, dim2_to_use = 2, dim3_to_use = NULL,
  sdimx = NULL, sdimy = NULL, sdimz = NULL, genes,
  dim_point_border_col = "black", dim_point_border_stroke = 0.1,
  show_NN_network = F, nn_network_to_use = "sNN",
  network_name = "sNN.pca", edge_alpha_dim = NULL,
  scale_alpha_with_expression = TRUE, label_size = 16,
  genes_low_color = "blue", genes_mid_color = "white",
  genes_high_color = "red", dim_point_size = 3,
  nn_network_alpha = 0.5, show_spatial_network = F,
  spatial_network_name = "spatial_network",
  network_color = "lightgray", spatial_network_alpha = 0.5,
  show_spatial_grid = F, spatial_grid_name = "spatial_grid",
  spatial_grid_color = NULL, spatial_grid_alpha = 0.5,
  spatial_point_size = 3, spatial_point_border_col = "black",
  spatial_point_border_stroke = 0.1, legend_text_size = 12,
  axis_scale = c("cube", "real", "custom"), custom_ratio = NULL,
  x_ticks = NULL, y_ticks = NULL, z_ticks = NULL, midpoint = 0,
  point_size = 1, cow_n_col = 2, cow_rel_h = 1, cow_rel_w = 1,
  cow_align = "h", show_legend = T, show_plots = F)
```

Arguments

gobject giotto object

expression_values gene expression values to use

plot_alignment direction to align plot

dim_reduction_to_use	dimension reduction to use
dim_reduction_name	dimension reduction name
dim1_to_use	dimension to use on x-axis
dim2_to_use	dimension to use on y-axis
dim3_to_use	dimension to use on z-axis
sdimx	x-axis dimension name (default = 'sdimx')
sdimy	y-axis dimension name (default = 'sdimy')
sdimz	z-axis dimension name (default = 'sdimz')
genes	genes to show
dim_point_border_col	color of border around points
dim_point_border_stroke	stroke size of border around points
show_NN_network	show underlying NN network
nn_network_to_use	type of NN network to use (kNN vs sNN)
network_name	name of NN network to use, if show_NN_network = TRUE
edge_alpha_dim	dim reduction plot: column to use for alpha of the edges
scale_alpha_with_expression	scale expression with ggplot alpha parameter
label_size	size for the label
genes_low_color	color to represent low expression of gene
genes_high_color	color to represent high expression of gene
dim_point_size	dim reduction plot: point size
spatial_network_name	name of spatial network to use
spatial_grid_name	name of spatial grid to use
spatial_point_size	spatial plot: point size
spatial_point_border_col	color of border around points
spatial_point_border_stroke	stroke size of border around points
legend_text_size	the size of the text in legend
axis_scale	three modes to adjust axis scale ratio
custom_ratio	set the axis scale ratio on custom
x_ticks	number of ticks on x axis
y_ticks	number of ticks on y axis

z_ticks	number of ticks on z axis
midpoint	size of point (cell)
point_size	size of point (cell)
cow_n_col	cowplot param: how many columns
cow_rel_h	cowplot param: relative height
cow_rel_w	cowplot param: relative width
cow_align	cowplot param: how to align
show_legend	show legend
show_plot	show plot

Details

Description of parameters.

Value

ggplot or plotly

Examples

```
visSpatDimGenePlot(gobject)
```

visSpatDimGenePlot_2D *visSpatDimGenePlot_2D*

Description

Visualize cells according to spatial AND dimension reduction coordinates in ggplot mode

Usage

```
visSpatDimGenePlot_2D(gobject, expression_values = c("normalized",
  "scaled", "custom"), plot_alignment = c("horizontal", "vertical"),
  genes, dim_reduction_to_use = "umap", dim_reduction_name = "umap",
  dim1_to_use = 1, dim2_to_use = 2, point_size = 1,
  dim_point_border_col = "black", dim_point_border_stroke = 0.1,
  show_NN_network = F, show_spatial_network = F,
  show_spatial_grid = F, nn_network_to_use = "sNN",
  network_name = "sNN.pca", edge_alpha_dim = NULL,
  scale_alpha_with_expression = TRUE,
  spatial_network_name = "spatial_network",
  spatial_grid_name = "spatial_grid", spatial_point_size = 1,
  spatial_point_border_col = "black",
  spatial_point_border_stroke = 0.1, midpoint = 0,
  genes_high_color = "red", genes_mid_color = "white",
  genes_low_color = "blue", cow_n_col = 2, cow_rel_h = 1,
  cow_rel_w = 1, cow_align = "h", axis_scale = c("cube", "real",
  "custom"), custom_ratio = NULL, x_ticks = NULL, y_ticks = NULL,
  show_legend = T, show_plots = F)
```

Arguments

<code>gobject</code>	giotto object
<code>expression_values</code>	gene expression values to use
<code>plot_alignment</code>	direction to align plot
<code>genes</code>	genes to show
<code>dim_reduction_to_use</code>	dimension reduction to use
<code>dim_reduction_name</code>	dimension reduction name
<code>dim1_to_use</code>	dimension to use on x-axis
<code>dim2_to_use</code>	dimension to use on y-axis
<code>point_size</code>	size of point (cell)
<code>dim_point_border_col</code>	color of border around points
<code>dim_point_border_stroke</code>	stroke size of border around points
<code>show_NN_network</code>	show underlying NN network
<code>nn_network_to_use</code>	type of NN network to use (kNN vs sNN)
<code>network_name</code>	name of NN network to use, if <code>show_NN_network = TRUE</code>
<code>edge_alpha_dim</code>	dim reduction plot: column to use for alpha of the edges
<code>scale_alpha_with_expression</code>	scale expression with ggplot alpha parameter
<code>spatial_network_name</code>	name of spatial network to use
<code>spatial_grid_name</code>	name of spatial grid to use
<code>spatial_point_size</code>	spatial plot: point size
<code>spatial_point_border_col</code>	color of border around points
<code>spatial_point_border_stroke</code>	stroke size of border around points
<code>midpoint</code>	size of point (cell)
<code>cow_n_col</code>	cowplot param: how many columns
<code>cow_rel_h</code>	cowplot param: relative height
<code>cow_rel_w</code>	cowplot param: relative width
<code>cow_align</code>	cowplot param: how to align
<code>show_legend</code>	show legend
<code>dim_point_size</code>	dim reduction plot: point size
<code>show_plot</code>	show plot

Details

Description of parameters.

Value

ggplot

Examples

```
visSpatDimGenePlot_2D(gobject)
```

```
visSpatDimGenePlot_3D visSpatDimGenePlot_3D
```

Description

Visualize cells according to spatial AND dimension reduction coordinates in plotly mode

Usage

```
visSpatDimGenePlot_3D(gobject, expression_values = c("normalized",
  "scaled", "custom"), plot_alignment = c("horizontal", "vertical"),
  dim_reduction_to_use = "umap", dim_reduction_name = "umap",
  dim1_to_use = 1, dim2_to_use = 2, dim3_to_use = NULL,
  sdimx = NULL, sdimy = NULL, sdimz = NULL, genes,
  show_NN_network = F, nn_network_to_use = "sNN",
  network_name = "sNN.pca", label_size = 16,
  genes_low_color = "blue", genes_mid_color = "white",
  genes_high_color = "red", dim_point_size = 3,
  nn_network_alpha = 0.5, show_spatial_network = F,
  spatial_network_name = "spatial_network",
  network_color = "lightgray", spatial_network_alpha = 0.5,
  show_spatial_grid = F, spatial_grid_name = "spatial_grid",
  spatial_grid_color = NULL, spatial_grid_alpha = 0.5,
  spatial_point_size = 3, legend_text_size = 12,
  axis_scale = c("cube", "real", "custom"), custom_ratio = NULL,
  x_ticks = NULL, y_ticks = NULL, z_ticks = NULL)
```

Arguments

<code>gobject</code>	giotto object
<code>plot_alignment</code>	direction to align plot
<code>dim_reduction_to_use</code>	dimension reduction to use
<code>dim_reduction_name</code>	dimension reduction name
<code>dim1_to_use</code>	dimension to use on x-axis
<code>dim2_to_use</code>	dimension to use on y-axis
<code>dim3_to_use</code>	dimension to use on z-axis

show_NN_network	show underlying NN network
nn_network_to_use	type of NN network to use (kNN vs sNN)
network_name	name of NN network to use, if show_NN_network = TRUE
genes_low_color	color represent high gene expression (see details)
genes_high_color	color represent high gene expression (see details)
nn_network_alpha	column to use for alpha of the edges
show_spatial_network	show spatial network
spatial_network_name	name of spatial network to use
network_color	color of spatial/nn network
spatial_network_alpha	alpha of spatial network
show_spatial_grid	show spatial grid
spatial_grid_name	name of spatial grid to use
spatial_grid_color	color of spatial grid
spatial_grid_alpha	alpha of spatial grid
legend_text_size	text size of legend
show_legend	show legend
show_plot	show plot

Details

Description of parameters.

Value

plotly

Examples

```
visSpatDimPlot_3D(gobject)
```

visSpatDimPlot

*visSpatDimPlot***Description**

integration of visSpatDimPlot_2D and visSpatDimPlot_3D

Usage

```
visSpatDimPlot(gobject, plot_method = c("ggplot", "plotly"),
  plot_alignment = NULL, dim_reduction_to_use = "umap",
  dim_reduction_name = "umap", dim1_to_use = 1, dim2_to_use = 2,
  dim3_to_use = NULL, sdimx = NULL, sdimy = NULL, sdimz = NULL,
  show_NN_network = F, nn_network_to_use = "sNN",
  network_name = "sNN.pca", show_cluster_center = F,
  show_center_label = T, center_point_size = 4, label_size = NULL,
  label_fontface = "bold", cell_color = NULL, color_as_factor = T,
  cell_color_code = NULL, select_cell_groups = NULL,
  select_cells = NULL, show_other_cells = T,
  other_cell_color = "lightgrey", dim_point_size = 3,
  dim_point_border_col = "black", dim_point_border_stroke = 0.1,
  nn_network_alpha = NULL, show_spatial_network = F,
  spatial_network_name = "spatial_network",
  network_color = "lightgray", spatial_network_alpha = 0.5,
  show_spatial_grid = F, spatial_grid_name = "spatial_grid",
  spatial_grid_color = NULL, spatial_grid_alpha = 0.5,
  spatial_point_size = 3, legend_text_size = 12,
  spatial_point_border_col = "black",
  spatial_point_border_stroke = 0.1, show_legend = T,
  axis_scale = c("cube", "real", "custom"), custom_ratio = NULL,
  x_ticks = NULL, y_ticks = NULL, z_ticks = NULL, show_plot = F)
```

Arguments

<code>gobject</code>	giotto object
<code>plot_alignment</code>	direction to align plot
<code>dim_reduction_to_use</code>	dimension reduction to use
<code>dim_reduction_name</code>	dimension reduction name
<code>dim1_to_use</code>	dimension to use on x-axis
<code>dim2_to_use</code>	dimension to use on y-axis
<code>dim3_to_use</code>	dimension to use on z-axis
<code>show_NN_network</code>	show underlying NN network
<code>nn_network_to_use</code>	type of NN network to use (kNN vs sNN)
<code>network_name</code>	name of NN network to use, if <code>show_NN_network = TRUE</code>
<code>cell_color</code>	color for cells (see details)

color_as_factor	convert color column to factor
cell_color_code	named vector with colors
select_cell_groups	select subset of cells/clusters based on cell_color parameter
select_cells	select subset of cells based on cell IDs
show_other_cells	display not selected cells
other_cell_color	color of not selected cells
nn_network_alpha	column to use for alpha of the edges
show_spatial_network	show spatial network
spatial_network_name	name of spatial network to use
spatial_network_alpha	alpha of spatial network
show_spatial_grid	show spatial grid
spatial_grid_name	name of spatial grid to use
spatial_grid_color	color of spatial grid
spatial_grid_alpha	alpha of spatial grid
legend_text_size	text size of legend
show_legend	show legend
show_plot	show plot
plot_mode	choose the mode to draw plot : ggplot or plotly
spatial_network_color	color of spatial network

Details

Description of parameters.

Value

ggplot or plotly

Examples

```
visSpatDimPlot(gobject)
```

visSpatDimPlot_2D	<i>visSpatDimPlot_2D</i>
-------------------	--------------------------

Description

Visualize cells according to spatial AND dimension reduction coordinates in ggplot2 mode

Usage

```
visSpatDimPlot_2D(gobject, plot_alignment = c("vertical", "horizontal"),
  dim_reduction_to_use = "umap", dim_reduction_name = "umap",
  dim1_to_use = 1, dim2_to_use = 2, sdimx = NULL, sdimy = NULL,
  show_NN_network = F, nn_network_to_use = "sNN",
  network_name = "sNN.pca", show_cluster_center = F,
  show_center_label = T, center_point_size = 4, label_size = 4,
  label_fontface = "bold", cell_color = NULL, color_as_factor = T,
  cell_color_code = NULL, select_cell_groups = NULL,
  select_cells = NULL, show_other_cells = T,
  other_cell_color = "lightgrey", dim_plot_mode = NULL,
  dim_point_size = 1, dim_point_border_col = "black",
  dim_point_border_stroke = 0.1, nn_network_alpha = 0.05,
  show_spatial_network = F, spatial_network_name = "spatial_network",
  spatial_network_color = NULL, show_spatial_grid = F,
  spatial_grid_name = "spatial_grid", spatial_grid_color = NULL,
  spatial_point_size = 1, spatial_point_border_col = "black",
  spatial_point_border_stroke = 0.1, show_legend = T, show_plot = F,
  plot_method = "ggplot")
```

Arguments

<code>gobject</code>	giotto object
<code>plot_alignment</code>	direction to align plot
<code>dim_reduction_to_use</code>	dimension reduction to use
<code>dim_reduction_name</code>	dimension reduction name
<code>dim1_to_use</code>	dimension to use on x-axis
<code>dim2_to_use</code>	dimension to use on y-axis
<code>show_NN_network</code>	show underlying NN network
<code>nn_network_to_use</code>	type of NN network to use (kNN vs sNN)
<code>network_name</code>	name of NN network to use, if <code>show_NN_network = TRUE</code>
<code>cell_color</code>	color for cells (see details)
<code>color_as_factor</code>	convert color column to factor
<code>cell_color_code</code>	named vector with colors

select_cell_groups	select subset of cells/clusters based on cell_color parameter
select_cells	select subset of cells based on cell IDs
show_other_cells	display not selected cells
other_cell_color	color of not selected cells
nn_network_alpha	column to use for alpha of the edges
show_spatial_network	show spatial network
spatial_network_name	name of spatial network to use
spatial_network_color	color of spatial network
show_spatial_grid	show spatial grid
spatial_grid_name	name of spatial grid to use
spatial_grid_color	color of spatial grid
show_legend	show legend
show_plot	show plot

Details

Description of parameters.

Value

ggplot

Examples

visSpatDimPlot_2D(gobject)

visSpatDimPlot_3D	<i>visSpatDimPlot_3D</i>
-------------------	--------------------------

Description

Visualize cells according to spatial AND dimension reduction coordinates in plotly mode

Usage

```
visSpatDimPlot_3D(gobject, plot_alignment = c("horizontal", "vertical"),
  dim_reduction_to_use = "umap", dim_reduction_name = "umap",
  dim1_to_use = 1, dim2_to_use = 2, dim3_to_use = NULL,
  sdimx = NULL, sdimy = NULL, sdimz = NULL, show_NN_network = F,
  nn_network_to_use = "sNN", network_name = "sNN.pca",
  show_cluster_center = F, show_center_label = T,
  center_point_size = 4, label_size = 16, cell_color = NULL,
  color_as_factor = T, cell_color_code = NULL, dim_point_size = 3,
  nn_network_alpha = 0.5, show_spatial_network = F,
  spatial_network_name = "spatial_network",
  network_color = "lightgray", spatial_network_alpha = 0.5,
  show_spatial_grid = F, spatial_grid_name = "spatial_grid",
  spatial_grid_color = NULL, spatial_grid_alpha = 0.5,
  spatial_point_size = 3, axis_scale = c("cube", "real", "custom"),
  custom_ratio = NULL, x_ticks = NULL, y_ticks = NULL,
  z_ticks = NULL, legend_text_size = 12)
```

Arguments

<code>gobject</code>	giotto object
<code>plot_alignment</code>	direction to align plot
<code>dim_reduction_to_use</code>	dimension reduction to use
<code>dim_reduction_name</code>	dimension reduction name
<code>dim1_to_use</code>	dimension to use on x-axis
<code>dim2_to_use</code>	dimension to use on y-axis
<code>dim3_to_use</code>	dimension to use on z-axis
<code>show_NN_network</code>	show underlying NN network
<code>nn_network_to_use</code>	type of NN network to use (kNN vs sNN)
<code>network_name</code>	name of NN network to use, if <code>show_NN_network = TRUE</code>
<code>cell_color</code>	color for cells (see details)
<code>color_as_factor</code>	convert color column to factor
<code>cell_color_code</code>	named vector with colors
<code>nn_network_alpha</code>	column to use for alpha of the edges
<code>show_spatial_network</code>	show spatial network
<code>spatial_network_name</code>	name of spatial network to use
<code>spatial_network_alpha</code>	alpha of spatial network
<code>show_spatial_grid</code>	show spatial grid

spatial_grid_name	name of spatial grid to use
spatial_grid_color	color of spatial grid
spatial_grid_alpha	alpha of spatial grid
legend_text_size	text size of legend
spatial_network_color	color of spatial network
show_legend	show legend
show_plot	show plot

Details

Description of parameters.

Value

plotly

Examples

```
visSpatDimPlot_3D(gobject)
```

```
write_giotto_viewer_annotation
      write_giotto_viewer_annotation
```

Description

write out annotation data from a giotto object for the Viewer

Usage

```
write_giotto_viewer_annotation(annotation, annot_name = "test",
  output_directory = getwd())
```

Arguments

annotation	annotation from the data.table from giotto object
annot_name	name of the annotation
output_directory	directory where to save the files

Value

write a .txt and .annot file for the selection annotation

```
write_giotto_viewer_dim_reduction
      write_giotto_viewer_dim_reduction
```

Description

write out dimensional reduction data from a giotto object for the Viewer

Usage

```
write_giotto_viewer_dim_reduction(dim_reduction_cell, dim_red = NULL,
  dim_red_name = NULL, dim_red_rounding = NULL,
  dim_red_rescale = c(-20, 20), output_directory = getwd())
```

Arguments

dim_reduction_cell	dimension reduction slot from giotto object
dim_red	high level name of dimension reduction
dim_red_name	specific name of dimension reduction to use
dim_red_rounding	numerical indicating how to round the coordinates
dim_red_rescale	numericals to rescale the coordinates
output_directory	directory where to save the files

Value

write a .txt and .annot file for the selection annotation

Index

- *Topic **giotto**,
 - giotto-class, [78](#)
 - print.giotto, [101](#)
 - show,giotto-method, [106](#)
- *Topic **giotto**
 - createGiottoObject, [30](#)
- *Topic **object**
 - giotto-class, [78](#)
 - print.giotto, [101](#)
 - show,giotto-method, [106](#)

- addCellMetadata, [5](#)
- addCellStatistics, [6](#)
- addGeneMetadata, [6](#)
- addGeneStatistics, [7](#)
- addHMRF, [8](#)
- addNetworkLayout, [8](#)
- addStatistics, [9](#)
- adjustGiottoMatrix, [10](#)
- allCellCellcommunicationsScores, [11](#)
- annotate_spatlocs_with_spatgrid_2D, [13](#)
- annotate_spatlocs_with_spatgrid_3D, [13](#)
- annotateGiotto, [12](#)
- annotateSpatialNetwork, [12](#)
- average_gene_gene_expression_in_groups,
[14](#)

- binGetSpatialGenes, [15](#)

- calculate_spatial_genes_python, [19](#)
- calculateHVG, [16](#)
- calculateMetaTable, [17](#)
- calculateSpatialGenes, [17](#)
- cellProximityBarplot, [20](#)
- cellProximityEnrichment, [20](#)
- cellProximityHeatmap, [21](#)
- cellProximityNetwork, [22](#)
- cellProximityVisPlot, [23](#)
- cellProximityVisPlot_2D_ggplot, [24](#)
- cellProximityVisPlot_2D_plotly, [26](#)
- cellProximityVisPlot_3D_plotly, [27](#)
- clusterCells, [28](#)
- create_average_detection_DT, [37](#)
- create_average_DT, [37](#)

- create_cell_type_random_cell_IDs, [38](#)
- create_cluster_matrix, [38](#)
- create_dimObject, [39](#)
- createGiottoObject, [30](#)
- createHeatmap_DT, [31](#)
- createNearestNetwork, [32](#)
- createSpatialGrid, [33](#)
- createSpatialGrid_2D, [34](#)
- createSpatialGrid_3D, [35](#)
- createSpatialNetwork, [36](#)

- decide_cluster_order, [39](#)
- detectSpatialPatterns, [40](#)
- direction_test (direction_test_CPG), [41](#)
- direction_test_CPG, [41](#)
- doHclust, [41](#)
- doHMRF, [42](#)
- doKmeans, [43](#)
- doLeidenCluster, [44](#)
- doLeidenSubCluster, [45](#)
- doLouvainCluster, [47](#)
- doLouvainCluster_community, [48](#)
- doLouvainCluster_multinet, [49](#)
- doLouvainSubCluster, [50](#)
- doLouvainSubCluster_community, [51](#)
- doLouvainSubCluster_multinet, [52](#)
- doRandomWalkCluster, [54](#)
- doSNNCluster, [55](#)
- dt_to_matrix, [56](#)

- exportGiottoViewer, [56](#)
- exprOnlyCellCellcommunicationScores,
[57](#)
- extended_gini_fun, [58](#)
- extractNearestNetwork, [58](#)

- fDataDT, [59](#)
- filterCombinations, [59](#)
- filterDistributions, [60](#)
- filterGiotto, [61](#)
- find_grid_2D, [69](#)
- find_grid_3D, [70](#)
- find_grid_x, [70](#)
- find_grid_y, [70](#)

[find_grid_z](#), 70
[findGiniMarkers](#), 62
[findGiniMarkers_one_vs_all](#), 63
[findMarkers](#), 64
[findMarkers_one_vs_all](#), 65
[findMastMarkers](#), 66
[findMastMarkers_one_vs_all](#), 67
[findScranMarkers](#), 68
[findScranMarkers_one_vs_all](#), 68
[fish_function](#), 71
[fish_function2](#), 71
[FSV_show](#), 71

[GenePattern_show](#), 72
[get_cell_to_cell_sorted_name_conversion](#), 77
[get_interaction_gene_enrichment](#), 77
[get_specific_interaction_gene_enrichment](#), 78
[getCellProximityGeneScores](#), 73
[getClusterSimilarity](#), 74
[getDendrogramSplits](#), 74
[getDistinctColors](#), 75
[getGeneToGeneSelection](#), 76
[giotto \(giotto-class\)](#), 78
[giotto-class](#), 78

[iterCluster](#), 79
[iterLeidenCluster](#), 80
[iterLouvainCluster](#), 81
[iterLouvainCluster_community](#), 82
[iterLouvainCluster_multinet](#), 83

[kmeans_binarize](#), 85

[loadHMRF](#), 85

[make_simulated_network](#), 86
[mergeClusters](#), 86
[mygini_fun](#), 87

[nnDT_to_kNN](#), 87
[node_clusters](#), 88
[normalizeGiotto](#), 88

[OR_function2](#), 89

[pDataDT](#), 89
[plot_network_layer_ggplot](#), 99
[plot_point_layer_ggplot](#), 99
[plotCPGscores](#), 90
[plotGTGscores](#), 91
[plotHeatmap](#), 92
[plotly_axis_scale_2D](#), 93
[plotly_axis_scale_3D](#), 94
[plotly_grid](#), 94
[plotly_network](#), 95
[plotMetaDataHeatmap](#), 95
[plotPCA](#), 97
[plotTSNE](#), 97
[plotUMAP](#), 98
[print.giotto](#), 101

[rank_binarize](#), 101
[runPCA](#), 102
[runtSNE](#), 103
[runUMAP](#), 104

[select_expression_values](#), 106
[selectPatternGenes](#), 105
[show,giotto-method](#), 106
[showClusterDendrogram](#), 106
[showClusterHeatmap](#), 107
[showCPGscores](#), 108
[showGeneExpressionProximityScore](#), 109
[showGTGscores](#), 109
[showIntExpressionProximityScore](#), 110
[showPattern](#), 111
[showPatternGenes](#), 112
[showProcessingSteps](#), 113
[showTopGeneToGene](#), 113
[signPCA](#), 114
[Spatial_AEH](#), 115
[Spatial_DE](#), 116
[specificCellCellcommunicationScores](#), 116
[split_dendrogram_in_two](#), 117
[stitchFieldCoordinates](#), 118
[subClusterCells](#), 119
[subsetGiotto](#), 120

[viewHMRFresults](#), 121
[violinPlot](#), 121
[visDimGenePlot](#), 122
[visDimGenePlot_2D_ggplot](#), 124
[visDimGenePlot_3D_plotly](#), 125
[visDimPlot](#), 97, 98, 126
[visDimPlot_2D_ggplot](#), 128
[visDimPlot_2D_plotly](#), 129
[visDimPlot_3D_plotly](#), 131
[visForceLayoutPlot](#), 132
[visGenePlot](#), 133
[visGenePlot_2D_ggplot](#), 134
[visGenePlot_3D_plotly](#), 136
[visPlot](#), 121, 137
[visPlot_2D_ggplot](#), 139
[visPlot_2D_plotly](#), 140

`visPlot_3D_plotly`, [141](#)
`visSpatDimGenePlot`, [143](#)
`visSpatDimGenePlot_2D`, [145](#)
`visSpatDimGenePlot_3D`, [147](#)
`visSpatDimPlot`, [149](#)
`visSpatDimPlot_2D`, [151](#)
`visSpatDimPlot_3D`, [152](#)

`write_giotto_viewer_annotation`, [154](#)
`write_giotto_viewer_dim_reduction`, [155](#)