

Package ‘Giotto’

April 3, 2020

Title Spatial single-cell transcriptomics toolbox.

Version 0.3.0

Description Toolbox to process, analyze and visualize spatial single-cell expression data.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

URL <https://rubd.github.io/Giotto/>, <https://github.com/RubD/Giotto>

RoxygenNote 7.1.0

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),
FactoMineR (>= 1.34),
factoextra (>= 1.0.5),
cowplot (>= 0.9.4),
grDevices,
RColorBrewer (>= 1.1-2),
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 (>= 1.14),
magrittr,
limma,
ggdendro,
smfishHmrf,
matrixStats (>= 0.55.0),
devtools,
reshape2,

ggraph,
Rcpp,
rlang ($\geq 0.4.3$),
fitdistrplus

Suggests knitr,
rmarkdown,
MAST,
scrn ($\geq 1.10.1$),
png,
tiff,
biomaRt,
trendsceek,
multinet ($\geq 3.0.2$),
RTriangle ($\geq 1.6-0.10$)

biocViews

VignetteBuilder knitr

LinkingTo Rcpp,
RcppArmadillo

Remotes lambdamoses/smfishhmr-f

R topics documented:

adapt_aspect_ratio	8
addCellIntMetadata	8
addCellMetadata	9
addCellStatistics	10
addGeneMetadata	11
addGeneStatistics	12
addHMRF	13
addNetworkLayout	13
addStatistics	14
adjustGiottoMatrix	15
aes_string2	16
all_plots_save_function	16
annotateGiotto	17
annotateSpatialNetwork	18
annotate_spatlocs_with_spatgrid_2D	19
annotate_spatlocs_with_spatgrid_3D	19
average_gene_gene_expression_in_groups	20
binSpect	20
calculateHVG	22
calculateMetaTable	24
calculateMetaTableCells	24
calculate_distance_and_weight	25
cellProximityBarplot	25
cellProximityEnrichment	27
cellProximityHeatmap	28
cellProximityNetwork	29
cellProximitySpatPlot	30
cellProximitySpatPlot2D	32

cellProximitySpatPlot3D	34
cellProximityVisPlot	36
cellProximityVisPlot_2D_ggplot	38
cellProximityVisPlot_2D_plotly	40
cellProximityVisPlot_3D_plotly	41
changeGiottoInstructions	43
clusterCells	44
clusterSpatialCorGenes	46
combCCcom	47
combineCellProximityGenes	48
combineCellProximityGenes_per_interaction	49
combineCPG	49
combineMetadata	50
convertEnsemblToGeneSymbol	51
convert_to_full_spatial_network	51
convert_to_reduced_spatial_network	52
createCrossSection	52
createGiottoInstructions	53
createGiottoObject	54
createHeatmap_DT	56
createMetagenes	57
createNearestNetwork	58
createSpatialDelaunayNetwork	60
createSpatialEnrich	61
createSpatialGrid	62
createSpatialGrid_2D	63
createSpatialGrid_3D	64
createSpatialKNNnetwork	65
createSpatialNetwork	66
create_2d_mesh_grid_line_obj	67
create_average_detection_DT	68
create_average_DT	68
create_cell_type_random_cell_IDs	69
create_cluster_matrix	70
create_crossSection_object	70
create_delaunayNetwork2D	71
create_delaunayNetwork3D	71
create_delaunayNetwork_deldir	72
create_delaunayNetwork_geometry	72
create_delaunayNetwork_geometry_3D	73
create_delaunayNetwork_RTriangle	73
create_dimObject	74
create_KNNnetwork_dbscan	74
create_mesh_grid_lines	75
create_spatialNetworkObject	75
crossSectionGenePlot	76
crossSectionGenePlot3D	78
crossSectionPlot	80
crossSectionPlot3D	83
decide_cluster_order	85
detectSpatialCorGenes	86
detectSpatialPatterns	87

dimCellPlot	88
dimCellPlot2D	91
dimGenePlot	94
dimGenePlot2D	96
dimGenePlot3D	98
dimPlot	100
dimPlot2D	103
dimPlot2D_single	106
dimPlot3D	109
doHclust	111
doHMRF	112
doKmeans	113
doLeidenCluster	115
doLeidenSubCluster	116
doLouvainCluster	118
doLouvainCluster_community	119
doLouvainCluster_multinet	120
doLouvainSubCluster	121
doLouvainSubCluster_community	123
doLouvainSubCluster_multinet	125
doRandomWalkCluster	126
doSNNCluster	128
do_cell_proximity_test	129
do_limtest	129
do_multi_permuttest_random	130
do_page_permutation	130
do_permuttest_original	131
do_permuttest_random	131
do_rank_permutation	132
do_spatial_grid_averaging	132
do_spatial_knn_smoothing	133
do_ttest	134
DT_removeNA	134
dt_to_matrix	135
estimateCellCellDistance	135
exportGiottoViewer	135
exprCellCellcom	137
extended_gini_fun	138
extend_vector	138
extractNearestNetwork	138
fDataDT	139
filterCellProximityGenes	139
filterCombinations	140
filterCPG	141
filterDistributions	142
filterGiotto	144
filter_network	145
findCellProximityGenes	145
findCellProximityGenes_per_interaction	147
findCPG	147
findGiniMarkers	149
findGiniMarkers_one_vs_all	150

findMarkers	152
findMarkers_one_vs_all	153
findMastMarkers	155
findMastMarkers_one_vs_all	156
findScranMarkers	157
findScranMarkers_one_vs_all	158
find_grid_2D	159
find_grid_3D	159
find_grid_x	159
find_grid_y	159
find_grid_z	160
find_x_y_ranges	160
general_save_function	160
get10Xmatrix	161
getClusterSimilarity	162
getDendrogramSplits	163
getDistinctColors	164
get_cross_section_coordinates	164
get_distance	164
get_sectionThickness	165
ggplot_save_function	165
giotto-class	166
heatmSpatialCorGenes	167
hyperGeometricEnrich	168
insertCrossSectionGenePlot3D	169
insertCrossSectionSpatPlot3D	171
kmeans_binarize	173
loadHMRP	174
makeSignMatrixPAGE	174
makeSignMatrixRank	175
make_simulated_network	176
mergeClusters	176
mygini_fun	177
my_arowMeans	178
my_growMeans	178
my_rowMeans	178
nnDT_to_kNN	179
node_clusters	179
normalizeGiotto	180
normalizeGiottoOld	181
PAGEEnrich	182
pDataDT	183
plotCCcomDotplot	184
plotCCcomHeatmap	185
plotCellProximityGenes	186
plotCombineCCcom	188
plotCombineCellCellCommunication	189
plotCombineCellProximityGenes	190
plotCombineCPG	191
plotCPG	193
plotHeatmap	194
plotICG	196

plotInteractionChangedGenes	197
plotly_axis_scale_2D	198
plotly_axis_scale_3D	198
plotly_grid	199
plotly_network	200
plotMetaDataCellsHeatmap	200
plotMetaDataHeatmap	202
plotPCA	204
plotPCA_2D	206
plotPCA_3D	208
plotRankSpatvsExpr	210
plotRecovery	211
plotRecovery_sub	212
plotStatDelaunayNetwork	212
plotTSNE	213
plotTSNE_2D	215
plotTSNE_3D	218
plotUMAP	219
plotUMAP_2D	221
plotUMAP_3D	223
plot_network_layer_ggplot	225
plot_point_layer_ggplot	226
plot_point_layer_ggplot_noFILL	228
plot_spat_point_layer_ggplot	229
plot_spat_point_layer_ggplot_noFILL	231
print.giotto	233
projection_fun	234
rankEnrich	234
rankSpatialCorGroups	235
rank_binarize	236
readGiottoInstructions	236
read_crossSection	237
removeCellAnnotation	237
removeGeneAnnotation	238
replaceGiottoInstructions	238
reshape_to_data_point	239
reshape_to_mesh_grid_obj	239
runPCA	239
runSNE	240
runUMAP	242
selectPatternGenes	243
select_expression_values	244
select_spatialNetwork	244
set_giotto_python_path	245
show,giotto-method	245
showClusterDendrogram	245
showClusterHeatmap	246
showGiottoInstructions	248
showPattern	248
showPattern2D	249
showPattern3D	250
showPatternGenes	251

showProcessingSteps	253
showSpatialCorGenes	253
signPCA	254
silhouetteRank	256
sort_combine_two_DT_columns	257
spatCellCellcom	257
spatCellPlot	258
spatCellPlot2D	261
spatDimCellPlot	264
spatDimCellPlot2D	268
spatDimGenePlot	272
spatDimGenePlot2D	275
spatDimGenePlot3D	278
spatDimPlot	280
spatDimPlot2D	284
spatDimPlot3D	288
spatGenePlot	291
spatGenePlot2D	293
spatGenePlot3D	296
spatialAEH	297
spatialDE	298
Spatial_AEH	300
Spatial_DE	301
spatNetwDistributions	301
spatNetwDistributionsDistance	303
spatNetwDistributionsKneighbors	304
spatPlot	305
spatPlot2D	308
spatPlot2D_single	311
spatPlot3D	314
spat_fish_func	316
spat_OR_func	316
specificCellCellcommunicationScores	317
split_dendrogram_in_two	318
stitchFieldCoordinates	318
stitchTileCoordinates	320
subClusterCells	320
subsetGiotto	322
subsetGiottoLocs	323
transform_2d_mesh_to_3d_mesh	324
trendSceek	324
viewHMRFresults	325
viewHMRFresults2D	326
viewHMRFresults3D	327
violinPlot	328
visDimGenePlot	329
visDimGenePlot_2D_ggplot	331
visDimGenePlot_3D_plotly	332
visDimPlot	334
visDimPlot_2D_ggplot	336
visDimPlot_2D_plotly	338
visDimPlot_3D_plotly	340

visForceLayoutPlot 341

visGenePlot 343

visGenePlot_2D_ggplot 345

visGenePlot_3D_plotly 346

visPlot 348

visPlot_2D_ggplot 350

visPlot_2D_plotly 353

visPlot_3D_plotly 354

visSpatDimGenePlot 356

visSpatDimGenePlot_2D 359

visSpatDimGenePlot_3D 361

visSpatDimPlot 363

visSpatDimPlot_2D 365

visSpatDimPlot_3D 368

writeHMRResults 370

write_giotto_viewer_annotation 370

write_giotto_viewer_dim_reduction 371

write_giotto_viewer_numeric_annotation 372

Index 373

adapt_aspect_ratio	<i>adapt_aspect_ratio</i>
--------------------	---------------------------

Description

adapt the aspect ratio after inserting cross section mesh grid lines

Usage

```
adapt_aspect_ratio(  
  current_ratio,  
  cell_locations,  
  sdimx = NULL,  
  sdimy = NULL,  
  sdimz = NULL,  
  mesh_obj = NULL  
)
```

addCellIntMetadata	<i>addCellIntMetadata</i>
--------------------	---------------------------

Description

Creates an additional metadata column with information about interacting and non-interacting cell types of the selected cell-cell interaction.

Usage

```
addCellIntMetadata(
  gobject,
  spatial_network = "spatial_network",
  cluster_column,
  cell_interaction,
  name = "select_int",
  return_gobject = TRUE
)
```

Arguments

gobject	giotto object
spatial_network	name of spatial network to use
cluster_column	column of cell types
cell_interaction	cell-cell interaction to use
name	name for the new metadata column
return_gobject	return an updated giotto object

Details

This function will create an additional metadata column which selects interacting cell types for a specific cell-cell interaction. For example, if you want to color interacting astrocytes and oligodendrocytes it will create a new metadata column with the values "select_astrocytes", "select_oligodendrocytes", "other_astrocytes", "other_oligodendrocytes" and "other". Where "other" is all other cell types found within the selected cell type column.

Value

Giotto object

Examples

```
addCellIntMetadata(gobject)
```

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

Description

adds cell metadata to the giotto object

Usage

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

Arguments

<code>gobject</code>	giotto object
<code>new_metadata</code>	new cell metadata to use (data.table, data.frame, ...)
<code>by_column</code>	merge metadata based on cell_ID column in pDataDT (default = FALSE)
<code>column_cell_ID</code>	column name of new metadata to use if by_column = TRUE

Details

You can add additional cell metadata in two manners: 1. Provide a data.table or data.frame with cell annotations in the same order as the cell_ID column in pDataDT(gobject) 2. Provide a data.table or data.frame with cell annotations and specify which column contains the cell IDs, these cell IDs need to match with the cell_ID column in pDataDT(gobject)

Value

giotto object

Examples

```
addCellMetadata(gobject)
```

<code>addCellStatistics</code>	<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

<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

This function will add the following statistics to cell metadata:

- `nr_genes`: Denotes in how many genes are detected per cell
- `perc_genes`: Denotes what percentage of genes is detected per cell
- `total_expr`: Shows the total sum of gene expression per cell

Value

giotto object if `return_gobject = TRUE`

Examples

```
addCellStatistics(gobject)
```

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

Description

adds gene metadata to the giotto object

Usage

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

Arguments

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

Details

You can add additional gene metadata in two manners: 1. Provide a `data.table` or `data.frame` with gene annotations in the same order as the `gene_ID` column in `fDataDT(gobject)` 2. Provide a `data.table` or `data.frame` with gene annotations and specify which column contains the gene IDs, these gene IDs need to match with the `gene_ID` column in `fDataDT(gobject)`

Value

giotto object

Examples

```
addGeneMetadata(gobject)
```

addGeneStatistics	<i>addGeneStatistics</i>
-------------------	--------------------------

Description

adds gene statistics to the giotto object

Usage

```
addGeneStatistics(  
  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

This function will add the following statistics to gene metadata:

- `nr_cells`: Denotes in how many cells the gene is detected
- `per_cells`: Denotes in what percentage of cells the gene is detected
- `total_expr`: Shows the total sum of gene expression in all cells
- `mean_expr`: Average gene expression in all cells
- `mean_expr_det`: Average gene expression in cells with detectable levels of the gene

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 selected nearest neighbor network

Usage

```
addNetworkLayout(
  gobject,
  nn_network_to_use = "sNN",
  network_name = "sNN.pca",
  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

This function creates layout coordinates based on the provided kNN or sNN. Currently only the force-directed graph layout "drl", see [layout_with_drl](#), is implemented. This provides an alternative to tSNE or UMAP based visualizations.

Value

giotto object with updated layout for selected NN network

Examples

```
addNetworkLayout(gobject)
```

`addStatistics`

addStatistics

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

See [addGeneStatistics](#) and [addCellStatistics](#)

Value

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

Examples

```
addStatistics(gobject)
```

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

Description

normalize and/or scale expression 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 (max = 2)
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

This function implements the [limma::removeBatchEffect](#) function to remove known batch effects and to adjust expression values according to provided covariates.

Value

giotto object

Examples

```
adjustGiottoMatrix(gobject)
```

aes_string2	<i>aes_string2</i>
-------------	--------------------

Description

makes sure aes_string can also be used with names that start with numeric values

Usage

```
aes_string2(...)
```

all_plots_save_function	<i>all_plots_save_function</i>
-------------------------	--------------------------------

Description

Function to automatically save plots to directory of interest

Usage

```
all_plots_save_function(
  gobject,
  plot_object,
  save_dir = NULL,
  save_folder = NULL,
  save_name = NULL,
  default_save_name = "giotto_plot",
  save_format = NULL,
  show_saved_plot = F,
  ncol = 1,
  nrow = 1,
  scale = 1,
  base_width = NULL,
  base_height = NULL,
  base_aspect_ratio = NULL,
  units = NULL,
  dpi = NULL,
  limitsize = TRUE,
  ...
)
```

Arguments

gobject	giotto object
plot_object	object to plot
save_dir	directory to save to
save_folder	folder in save_dir to save to

save_name	name of plot
save_format	format (e.g. png, tiff, pdf, ...)
show_saved_plot	load & display the saved plot
ncol	number of columns
nrow	number of rows
scale	scale
base_width	width
base_height	height
base_aspect_ratio	aspect ratio
units	units
dpi	Plot resolution
limitsize	When TRUE (the default), ggsave will not save images larger than 50x50 inches, to prevent the common error of specifying dimensions in pixels.
...	additional parameters to ggplot_save_function or general_save_function

See Also

[general_save_function](#)

Examples

```
all_plots_save_function(gobject)
```

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

Description

Converts cluster results into provided annotation.

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

You need to specify which (cluster) column you want to annotate and you need to provide an annotation vector like this:

- 1. identify the cell type of each cluster
- 2. create a vector of these cell types, e.g. `cell_types = c('T-cell', 'B-cell', 'Stromal')`
- 3. provide original cluster names to previous vector, e.g. `names(cell_types) = c(2, 1, 3)`

Value

giotto object

Examples

```
annotateGiotto(gobject)
```

```
annotateSpatialNetwork
```

```
annotateSpatialNetwork
```

Description

Annotate spatial network with cell metadata information.

Usage

```
annotateSpatialNetwork(
  gobject,
  spatial_network_name = "Delaunay_network",
  cluster_column,
  create_full_network = F
)
```

Arguments

```
gobject          giotto object
spatial_network_name
                  name of spatial network to use
cluster_column  name of column to use for clusters
create_full_network
                  convert from reduced to full network representation
```

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 soon.

Value

data.table with average expression scores for each cluster

Examples

```
average_gene_gene_expression_in_groups(gobject)
```

binSpect

binSpect

Description

Previously: binGetSpatialGenes. BinSpect (Binary Spatial Extraction of genes) is a fast computational method that identifies genes with a spatially coherent expression pattern.

Usage

```
binSpect(
  gobject,
  bin_method = c("kmeans", "rank"),
  expression_values = c("normalized", "scaled", "custom"),
  subset_genes = NULL,
  spatial_network_name = "Delaunay_network",
  nstart = 3,
  iter_max = 10,
  percentage_rank = 30,
  do_fisher_test = TRUE,
  calc_hub = FALSE,
  hub_min_int = 3,
  get_av_expr = TRUE,
  get_high_expr = TRUE,
  do_parallel = TRUE,
  cores = NA,
  verbose = T
)
```

Arguments

<code>gobject</code>	giotto object
<code>bin_method</code>	method to binarize gene expression
<code>expression_values</code>	expression values to use
<code>subset_genes</code>	only select a subset of genes to test
<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>percentage_rank</code>	percentage of top cells for binarization
<code>do_fisher_test</code>	perform fisher test
<code>calc_hub</code>	calculate the number of hub cells
<code>hub_min_int</code>	minimum number of cell-cell interactions for a hub cell
<code>get_av_expr</code>	calculate the average expression per gene of the high expressing cells
<code>get_high_expr</code>	calculate the number of high expressing cells per gene
<code>do_parallel</code>	run calculations in parallel with mclapply
<code>cores</code>	number of cores to use if <code>do_parallel = TRUE</code>
<code>verbose</code>	be verbose

Details

We provide two ways to identify spatial genes based on gene expression binarization. Both methods are identical except for how binarization is performed.

- 1. binarize: Each gene is binarized (0 or 1) in each cell with **kmeans** (k = 2) or based on **rank** percentile

- 2. network: All cells are connected through a spatial network based on the physical coordinates
- 3. contingency table: A contingency table is calculated based on all edges of neighboring cells and the binarized expression (0-0, 0-1, 1-0 or 1-1)
- 4. For each gene an odds-ratio (OR) and fisher.test (optional) is calculated

Other statistics are provided (optional):

- Number of cells with high expression (binary = 1)
- Average expression of each gene within high expressing cells
- Number of hub cells, these are high expressing cells that have a user defined number of high expressing neighbors

By selecting a subset of likely spatial genes (e.g. soft thresholding highly variable genes) or using multiple cores can accelerate the speed.

Value

data.table with results (see details)

Examples

```
binSpect(gobject)
```

calculateHVG

calculateHVG

Description

compute highly variable genes

Usage

```
calculateHVG(
  gobject,
  expression_values = c("normalized", "scaled", "custom"),
  method = c("cov_groups", "cov_loess"),
  reverse_log_scale = FALSE,
  logbase = 2,
  expression_threshold = 0,
  nr_expression_groups = 20,
  zscore_threshold = 1.5,
  HVGname = "hvg",
  difference_in_cov = 0.1,
  show_plot = NA,
  return_plot = NA,
  save_plot = NA,
  save_param = list(),
  default_save_name = "HVGplot",
  return_gobject = TRUE
)
```

Arguments

<code>gobject</code>	giotto object
<code>expression_values</code>	expression values to use
<code>method</code>	method to calculate highly variable genes
<code>reverse_log_scale</code>	reverse log-scale of expression values (default = FALSE)
<code>logbase</code>	if <code>reverse_log_scale</code> is TRUE, which log base was used?
<code>expression_threshold</code>	expression threshold to consider a gene detected
<code>nr_expression_groups</code>	number of expression groups for <code>cov_groups</code>
<code>zscore_threshold</code>	zscore to select hvg for <code>cov_groups</code>
<code>HVGname</code>	name for highly variable genes in cell metadata
<code>difference_in_cov</code>	minimum difference in coefficient of variance required
<code>show_plot</code>	show plot
<code>return_plot</code>	return ggplot object
<code>save_plot</code>	directly save the plot [boolean]
<code>save_param</code>	list of saving parameters from all_plots_save_function
<code>default_save_name</code>	default save name for saving, don't change, change <code>save_name</code> in <code>save_param</code>
<code>return_gobject</code>	boolean: return giotto object (default = TRUE)

Details

Currently we provide 2 ways to calculate highly variable genes: **1. high coeff of variance (COV) within groups:**

First genes are binned (`nr_expression_groups`) into average expression groups and the COV for each gene is converted into a z-score within each bin. Genes with a z-score higher than the threshold (`zscore_threshold`) are considered highly variable.

2. high COV based on loess regression prediction:

A predicted COV is calculated for each gene using loess regression ($COV \sim \log(\text{mean expression})$). Genes that show a higher than predicted COV (`difference_in_cov`) are considered highly variable.

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)
```

calculateMetaTableCells	<i>calculateMetaTableCells</i>
-------------------------	--------------------------------

Description

calculates the average metadata values for one or more (combined) annotation columns.

Usage

```
calculateMetaTableCells(
  gobject,
  value_cols = NULL,
  metadata_cols = NULL,
  spat_enr_names = NULL
)
```


Arguments

<code>gobject</code>	giotto object
<code>value_cols</code>	metadata or enrichment value columns to use
<code>metadata_cols</code>	annotation columns found in <code>pDataDT(gobject)</code>
<code>spat_enr_names</code>	which spatial enrichment results to include

Value

data.table with average metadata values per (combined) annotation

Examples

```
calculateMetaTableCells(gobject)
```

```
calculate_distance_and_weight
      calculate_distance_and_weight
```

Description

`calculate_distance_and_weight`

Usage

```
calculate_distance_and_weight(
  networkDT,
  sdimx = "sdimx",
  sdimy = "sdimy",
  sdimz = "sdimz",
  d2_or_d3 = c(2, 3)
)
```

```
cellProximityBarplot  cellProximityBarplot
```

Description

Create barplot from cell-cell proximity scores

Usage

```
cellProximityBarplot(
  gobject,
  CPscore,
  min_orig_ints = 5,
  min_sim_ints = 5,
  p_val = 0.05,
  show_plot = NA,
  return_plot = NA,
  save_plot = NA,
  save_param = list(),
  default_save_name = "cellProximityBarplot"
)
```

Arguments

<code>gobject</code>	giotto object
<code>CPscore</code>	CPscore, output from <code>cellProximityEnrichment()</code>
<code>min_orig_ints</code>	filter on minimum original cell-cell interactions
<code>min_sim_ints</code>	filter on minimum simulated cell-cell interactions
<code>p_val</code>	p-value
<code>show_plot</code>	show plot
<code>return_plot</code>	return ggplot object
<code>save_plot</code>	directly save the plot [boolean]
<code>save_param</code>	list of saving parameters from all_plots_save_function
<code>default_save_name</code>	default save name for saving, don't change, change <code>save_name</code> in <code>save_param</code>

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 = "Delaunay_network",  
  cluster_column,  
  number_of_simulations = 1000,  
  adjust_method = c("none", "fdr", "bonferroni", "BH", "holm", "hochberg", "hommel",  
    "BY")  
)
```

Arguments

<code>gobject</code>	giotto object
<code>spatial_network_name</code>	name of spatial network to use
<code>cluster_column</code>	name of column to use for clusters
<code>number_of_simulations</code>	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 reshuffling 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(
  gobject,
  CPscore,
  scale = T,
  order_cell_types = T,
  color_breaks = NULL,
  color_names = NULL,
  show_plot = NA,
  return_plot = NA,
  save_plot = NA,
  save_param = list(),
  default_save_name = "cellProximityHeatmap"
)
```

Arguments

gobject	giotto object
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
show_plot	show plot
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param

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(
  gobject,
  CPscore,
  remove_self_edges = FALSE,
  self_loop_strength = 0.1,
  color_depletion = "lightgreen",
  color_enrichment = "red",
  rescale_edge_weights = TRUE,
  edge_weight_range_depletion = c(0.1, 1),
  edge_weight_range_enrichment = c(1, 5),
  layout = c("Fruchterman", "DrL", "Kamada-Kawai"),
  only_show_enrichment_edges = F,
  edge_width_range = c(0.1, 2),
  node_size = 4,
  node_text_size = 6,
  show_plot = NA,
  return_plot = NA,
  save_plot = NA,
  save_param = list(),
  default_save_name = "cellProximityNetwork"
)
```

Arguments

gobject	giotto object
CPscore	CPscore, output from cellProximityEnrichment()
remove_self_edges	remove enrichment/depletion edges with itself
self_loop_strength	size of self-loops
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
only_show_enrichment_edges	show only the enriched pairwise scores
edge_width_range	range of edge width
node_size	size of nodes
node_text_size	size of node labels
show_plot	show plot
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param

Details

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

Value

igraph plot

Examples

```
cellProximityNetwork(CPscore)
```

cellProximitySpatPlot *cellProximitySpatPlot*

Description

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

Usage

```
cellProximitySpatPlot(gobject, ...)
```

Arguments

gobject	giotto object
interaction_name	cell-cell interaction name
cluster_column	cluster column with cell clusters
sdimx	x-axis dimension name (default = 'sdimx')
sdimy	y-axis dimension name (default = 'sdimy')

cell_color	color for cells (see details)
cell_color_code	named vector with colors
color_as_factor	convert color column to factor
show_other_cells	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
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
show_plot	show plots
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param

Details

Description of parameters.

Value

ggplot

See Also

[cellProximitySpatPlot2D](#) and [cellProximitySpatPlot3D](#) for 3D

Examples

```
cellProximitySpatPlot(gobject)
```

```
cellProximitySpatPlot2D
```

```
cellProximitySpatPlot2D
```

Description

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

Usage

```
cellProximitySpatPlot2D(
  gobject,
  interaction_name = NULL,
  cluster_column = NULL,
  sdimx = "sdimx",
  sdimy = "sdimy",
  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 = "Delaunay_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,
  show_plot = NA,
  return_plot = NA,
  save_plot = NA,
  save_param = list(),
  default_save_name = "cellProximitySpatPlot2D"
)
```

Arguments

gobject	giotto object
interaction_name	cell-cell interaction name

cluster_column	cluster column with cell clusters
sdimx	x-axis dimension name (default = 'sdimx')
sdimy	y-axis dimension name (default = 'sdimy')
cell_color	color for cells (see details)
cell_color_code	named vector with colors
color_as_factor	convert color column to factor
show_other_cells	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
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
show_plot	show plots
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param

Details

Description of parameters.

Value

ggplot

Examples

```
cellProximitySpatPlot2D(gobject)
```

```
cellProximitySpatPlot3D
```

```
cellProximitySpatPlot2D
```

Description

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

Usage

```
cellProximitySpatPlot3D(
  gobject,
  interaction_name = NULL,
  cluster_column = NULL,
  sdimx = "sdimx",
  sdimy = "sdimy",
  sdimz = "sdimz",
  cell_color = NULL,
  cell_color_code = NULL,
  color_as_factor = T,
  show_other_cells = T,
  show_network = T,
  show_other_network = F,
  network_color = NULL,
  spatial_network_name = "Delaunay_network",
  show_grid = F,
  grid_color = NULL,
  spatial_grid_name = "spatial_grid",
  show_legend = T,
  point_size_select = 4,
  point_size_other = 2,
  point_alpha_other = 0.5,
  axis_scale = c("cube", "real", "custom"),
  custom_ratio = NULL,
  x_ticks = NULL,
  y_ticks = NULL,
  z_ticks = NULL,
  show_plot = NA,
  return_plot = NA,
  save_plot = NA,
  save_param = list(),
  default_save_name = "cellProximitySpatPlot3D",
  ...
)
```

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
<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>point_size_other</code>	size of other points
<code>show_plot</code>	show plots
<code>return_plot</code>	return plotly object
<code>save_plot</code>	directly save the plot [boolean]
<code>save_param</code>	list of saving parameters from all_plots_save_function
<code>default_save_name</code>	default save name for saving, don't change, change save_name in save_param

Details

Description of parameters.

Value

plotly

Examples

```
cellProximitySpatPlot3D(gobject)
```

 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 = "Delaunay_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
<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 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 = "Delaunay_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

color_as_factor	convert color column to factor
show_other_cells	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
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

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 = "Delaunay_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

color_as_factor	convert color column to factor
show_other_cells	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_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 = "Delaunay_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
<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_3D_plotly(gobject)
```

changeGiottoInstructions
changeGiottoInstructions

Description

Function to change one or more instructions from giotto object

Usage

```
changeGiottoInstructions(  
  gobject,  
  params = NULL,  
  new_values = NULL,  
  return_gobject = TRUE  
)
```

Arguments

gobject	giotto object
params	parameter(s) to change
new_values	new value(s) for parameter(s)
return_gobject	(boolean) return giotto object

Value

named vector with giotto instructions

Examples

```
changeGiottoInstructions()
```

clusterCells

*clusterCells***Description**

cluster cells using a variety of different methods

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

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

Details

Wrapper for the different clustering methods.

Value

giotto object with new clusters appended to cell metadata

See Also

[doLeidenCluster](#), [doLouvainCluster_community](#), [doLouvainCluster_multinet](#), [doLouvainCluster](#), [doRandomWalkCluster](#), [doSNNCluster](#), [doKmeans](#), [doHclust](#)

Examples

```
clusterCells(gobject)
```

```
clusterSpatialCorGenes
```

```
clusterSpatialCorGenes
```

Description

Cluster based on spatially correlated genes

Usage

```
clusterSpatialCorGenes(
  spatCorObject,
  name = "spat_clus",
  hclust_method = "ward.D",
  k = 10,
  return_obj = TRUE
)
```

Arguments

spatCorObject	spatial correlation object
name	name for spatial clustering results
hclust_method	method for hierarchical clustering
k	number of clusters to extract
return_obj	return spatial correlation object (spatCorObject)

Value

spatCorObject or cluster results

Examples

```
clusterSpatialCorGenes(gobject)
```

combCCcom

combCCcom

Description

Combine spatial and expression based cell-cell communication data.tables

Usage

```
combCCcom(
  spatialCC,
  exprCC,
  min_lig_nr = 3,
  min_rec_nr = 3,
  min_padj_value = 1,
  min_log2fc = 0,
  min_av_diff = 0
)
```

Arguments

spatialCC	spatial cell-cell communication scores
exprCC	expression cell-cell communication scores
min_lig_nr	minimum number of ligand cells
min_rec_nr	minimum number of receptor cells
min_padj_value	minimum adjusted p-value
min_log2fc	minimum log2 fold-change
min_av_diff	minimum average expression difference

Value

combined data.table with spatial and expression communication data

Examples

```
combCCcom(gobject)
```

```
combineCellProximityGenes
      combineCellProximityGenes
```

Description

Combine CPG scores in a pairwise manner.

Usage

```
combineCellProximityGenes(
  cpgObject,
  selected_ints = NULL,
  selected_genes = NULL,
  specific_genes_1 = NULL,
  specific_genes_2 = NULL,
  min_cells = 5,
  min_int_cells = 3,
  min_fdr = 0.05,
  min_spat_diff = 0,
  min_log2_fc = 0.5,
  do_parallel = TRUE,
  cores = NA,
  verbose = T
)
```

Arguments

cpgObject	cell proximity gene score object
selected_ints	subset of selected cell-cell interactions (optional)
selected_genes	subset of selected genes (optional)
specific_genes_1	specific geneset combo (need to position match specific_genes_2)
specific_genes_2	specific geneset combo (need to position match specific_genes_1)
min_cells	minimum number of target cell type
min_int_cells	minimum number of interacting cell type
min_fdr	minimum adjusted p-value
min_spat_diff	minimum absolute spatial expression difference
min_log2_fc	minimum absolute log2 fold-change
do_parallel	run calculations in parallel with mclapply
cores	number of cores to use if do_parallel = TRUE
verbose	verbose

Value

cpgObject that contains the filtered differential gene scores

Examples

```
combineCellProximityGenes(gobject)
```

```
combineCellProximityGenes_per_interaction
      combineCellProximityGenes_per_interaction
```

Description

Combine CPG scores per interaction

Usage

```
combineCellProximityGenes_per_interaction(
  cpgObject,
  sel_int,
  selected_genes = NULL,
  specific_genes_1 = NULL,
  specific_genes_2 = NULL,
  min_cells = 5,
  min_int_cells = 3,
  min_fdr = 0.05,
  min_spat_diff = 0,
  min_log2_fc = 0.5
)
```

Examples

```
combineCellProximityGenes_per_interaction()
```

```
combineCPG      combineCPG
```

Description

Combine CPG scores in a pairwise manner.

Usage

```
combineCPG(
  cpgObject,
  selected_ints = NULL,
  selected_genes = NULL,
  specific_genes_1 = NULL,
  specific_genes_2 = NULL,
  min_cells = 5,
  min_int_cells = 3,
  min_fdr = 0.05,
  min_spat_diff = 0,
```

```

    min_log2_fc = 0.5,
    do_parallel = TRUE,
    cores = NA,
    verbose = T
  )

```

Arguments

cpgObject	cell proximity gene score object
selected_ints	subset of selected cell-cell interactions (optional)
selected_genes	subset of selected genes (optional)
specific_genes_1	specific geneset combo (need to position match specific_genes_2)
specific_genes_2	specific geneset combo (need to position match specific_genes_1)
min_cells	minimum number of target cell type
min_int_cells	minimum number of interacting cell type
min_fdr	minimum adjusted p-value
min_spat_diff	minimum absolute spatial expression difference
min_log2_fc	minimum absolute log2 fold-change
do_parallel	run calculations in parallel with mclapply
cores	number of cores to use if do_parallel = TRUE
verbose	verbose

Value

cpgObject that contains the filtered differential gene scores

Examples

```
combineCPG(gobject)
```

combineMetadata	<i>combineMetadata</i>
-----------------	------------------------

Description

This function combines the cell metadata with spatial locations and enrichment results from createSpatialEnrich

Usage

```
combineMetadata(gobject, spat_enr_names = NULL)
```

Arguments

gobject	Giotto object
spat_enr_names	names of spatial enrichment results to include

Value

Extended cell metadata in data.table format.

Examples

```
combineMetadata(gobject)
```

```
convertEnsemblToGeneSymbol
```

```
convertEnsemblToGeneSymbol
```

Description

This function convert ensembl gene IDs from a matrix to official gene symbols

Usage

```
convertEnsemblToGeneSymbol(matrix, species = c("mouse", "human"))
```

Arguments

matrix	an expression matrix with ensembl gene IDs as rownames
species	species to use for gene symbol conversion

Details

This function requires that the biomaRt library is installed

Value

expression matrix with gene symbols as rownames

Examples

```
convertEnsemblToGeneSymbol(matrix)
```

```
convert_to_full_spatial_network
```

```
convert_to_full_spatial_network
```

Description

convert to a full spatial network

Usage

```
convert_to_full_spatial_network(reduced_spatial_network_DT)
```

```
convert_to_reduced_spatial_network
      convert_to_reduced_spatial_network
```

Description

convert to a reduced spatial network

Usage

```
convert_to_reduced_spatial_network(full_spatial_network_DT)
```

```
createCrossSection      createCrossSection
```

Description

Create a virtual 2D cross section.

Usage

```
createCrossSection(
  gobject,
  name = "cross_section",
  spatial_network_name = "Delaunay_network",
  thickness_unit = c("cell", "natural"),
  slice_thickness = 2,
  cell_distance_estimate_method = "mean",
  extend_ratio = 0.2,
  method = c("equation", "3 points", "point and norm vector",
    "point and two plane vectors"),
  equation = NULL,
  point1 = NULL,
  point2 = NULL,
  point3 = NULL,
  normVector = NULL,
  planeVector1 = NULL,
  planeVector2 = NULL,
  mesh_grid_n = 20,
  return_gobject = TRUE
)
```

Arguments

<code>gobject</code>	giotto object
<code>name</code>	name of cress section object. (default = cross_sectino)
<code>spatial_network_name</code>	name of spatial network object. (default = Delaunay_network)

thickness_unit	unit of the virtual section thickness. If "cell", average size of the observed cells is used as length unit. If "natural", the unit of cell location coordinates is used.(default = cell)
cell_distance_estimate_method	method to estimate average distance between neighboring cells. (default = mean)
extend_ratio	deciding the span of the cross section meshgrid, as a ratio of extension compared to the borders of the virtual tissue section. (default = 0.2)
method	method to define the cross section plane. If equation, the plane is defined by a four element numerical vector (equation) in the form of c(A,B,C,D), corresponding to a plane with equation $Ax+By+Cz=D$. If 3 points, the plane is defined by the coordinates of 3 points, as given by point1, point2, and point3. If point and norm vector, the plane is defined by the coordinates of one point (point1) in the plane and the coordinates of one norm vector (normVector) to the plane. If point and two plane vector, the plane is defined by the coordinates of one point (point1) in the plane and the coordinates of two vectors (planeVector1, planeVector2) in the plane. (default = equation)
equation	equation required by method "equation".equations needs to be a numerical vector of length 4, in the form of c(A,B,C,D), which defines plane $Ax+By+Cz=D$.
point1	coordinates of the first point required by method "3 points", "point and norm vector", and "point and two plane vectors".
point2	coordinates of the second point required by method "3 points"
point3	coordinates of the third point required by method "3 points"
normVector	coordinates of the norm vector required by method "point and norm vector"
planeVector1	coordinates of the first plane vector required by method "point and two plane vectors"
planeVector2	coordinates of the second plane vector required by method "point and two plane vectors"
mesh_grid_n	number of meshgrid lines to generate along both directions for the cross section plane.
return_gobject	boolean: return giotto object (default = TRUE)

Details

Creates a virtual 2D cross section object for a given spatial network object. The users need to provide the definition of the cross section plane (see method).

Value

giotto object with updated spatial network slot

```
createGiottoInstructions
      createGiottoInstructions
```

Description

Function to set global instructions for giotto functions

Usage

```
createGiottoInstructions(
  python_path = NULL,
  show_plot = NULL,
  return_plot = NULL,
  save_plot = NULL,
  save_dir = NULL,
  plot_format = NULL,
  dpi = NULL,
  units = NULL,
  height = NULL,
  width = NULL
)
```

Arguments

python_path	path to python binary to use
show_plot	print plot to console, default = TRUE
return_plot	return plot as object, default = TRUE
save_plot	automatically save plot, default = FALSE
save_dir	path to directory where to save plots
dpi	resolution for raster images
height	height of plots
width	width of plots

Value

named vector with giotto instructions

Examples

```
createGiottoInstructions()
```

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,
    spatial_enrichment = NULL,
    spatial_enrichment_name = NULL,
    dimension_reduction = NULL,
    nn_network = NULL,
    offset_file = NULL,
    instructions = NULL
)

```

Arguments

raw_exprs	matrix with raw expression counts [required]
spatial_locs	data.table or data.frame with coordinates for cell centroids
norm_expr	normalized expression values
norm_scaled_expr	scaled expression values
custom_expr	custom expression values
cell_metadata	cell annotation metadata
gene_metadata	gene annotation 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)
spatial_enrichment	list of spatial enrichment score(s) for each spatial region
spatial_enrichment_name	list of spatial enrichment 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)
instructions	list of instructions or output result from createGiottoInstructions

Details

[Requirements] To create a giotto object you need to provide at least a matrix with genes as row names and cells as column names. To include spatial information about cells (or regions) you need to provide a data.table or data.frame with coordinates for all spatial dimensions. This can be 2D (x and y) or 3D (x, y, x). The row order for the cell coordinates should be the same as the column order for the provided expression data.

[Instructions] Additionally an instruction file, generated manually or with [createGiottoInstructions](#) can be provided to instructions, if not a default instruction file will be created for the Giotto object.

[Multiple fields] In case a dataset consists of multiple fields, like seqFISH+ for example, an offset file can be provided to stitch the different fields together. [stitchFieldCoordinates](#) can be used to generate such an offset file.

[Processed data] Processed count data, such as normalized data, can be provided using one of the different expression slots (norm_expr, norm_scaled_expr, custom_expr).

[Metadata] Cell and gene metadata can be provided using the cell and gene metadata slots. This data can also be added afterwards using the [addGeneMetadata](#) or [addCellMetadata](#) functions.

[Other information] Additional information can be provided through the appropriate slots:

- spatial networks
- spatial girds
- spatial enrichments
- dimensions reductions
- nearest neighbours networks

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("correlation", "custom"),
  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)
```

createMetagenes	<i>createMetagenes</i>
-----------------	------------------------

Description

This function creates an average metagene for gene clusters.

Usage

```
createMetagenes(
  gobject,
  expression_values = c("normalized", "scaled", "custom"),
  gene_clusters,
  name = "metagene",
  return_gobject = TRUE
)
```

Arguments

gobject Giotto object
 expression_values expression values to use
 gene_clusters numerical vector with genes as names
 name name of the metagene results
 return_gobject return giotto object

Details

An example for the 'gene_clusters' could be like this: `cluster_vector = c(1, 1, 2, 2); names(cluster_vector) = c('geneA', 'geneB', 'geneC', 'geneD')`

Value

giotto object

Examples

```
createMetagenes(gobject)
```

```
createNearestNetwork    createNearestNetwork
```

Description

create a nearest neighbour (NN) network

Usage

```

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

Arguments

<code>gobject</code>	giotto object
<code>type</code>	sNN or kNN
<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>expression_values</code>	expression values 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 for kNN and sNN functions from dbscan

Details

This function creates a k-nearest neighbour (kNN) or shared nearest neighbour (sNN) network based on the provided dimension reduction space. To run it directly on the gene expression matrix set `dim_reduction_to_use = NULL`.

See also [kNN](#) and [sNN](#) for more information about how the networks are created.

Output for kNN:

- from: `cell_ID` for source cell
- to: `cell_ID` for target cell
- distance: distance between cells
- weight: $\text{weight} = 1/(1 + \text{distance})$

Output for sNN:

- from: `cell_ID` for source cell
- to: `cell_ID` for target cell
- distance: distance between cells
- weight: $1/(1 + \text{distance})$
- shared: number of shared neighbours
- rank: ranking of pairwise cell neighbours

For sNN networks two additional parameters can be set:

- `minimum_shared`: minimum number of shared neighbours needed
- `top_shared`: keep this number of the top shared neighbours, irrespective of `minimum_shared` setting

Value

giotto object with updated NN network

Examples

```
createNearestNetwork(gobject)
```

```
createSpatialDelaunayNetwork
```

createSpatialDelaunayNetwork

Description

Create a spatial Delaunay network based on cell centroid physical distances.

Usage

```
createSpatialDelaunayNetwork(
  gobject,
  method = c("delaunayn_geometry", "RTriangle", "deldir"),
  dimensions = "all",
  name = "delaunay_network",
  maximum_distance = "auto",
  minimum_k = 0,
  options = "Pp",
  Y = TRUE,
  j = TRUE,
  S = 0,
  verbose = T,
  return_gobject = TRUE,
  ...
)
```

Arguments

<code>gobject</code>	giotto object
<code>dimensions</code>	which spatial dimensions to use (default = all)
<code>name</code>	name for spatial network (default = 'delaunay_network')
<code>maximum_distance</code>	distance cutoff for Delaunay neighbors to consider. If "auto", "upper whisker" value of the distance vector between neighbors is used; see the boxplotgraphics documentation for more details.(default = "auto")
<code>minimum_k</code>	minimum number of neighbours if maximum_distance != NULL
<code>options</code>	(geometry) String containing extra control options for the underlying Qhull command; see the Qhull documentation (../doc/qhull/html/qdelaun.html) for the available options. (default = 'Pp', do not report precision problems)
<code>Y</code>	(RTriangle) If TRUE prohibits the insertion of Steiner points on the mesh boundary.

j	(RTriangle) If TRUE jettisons vertices that are not part of the final triangulation from the output.
S	(RTriangle) Specifies the maximum number of added Steiner points.
verbose	verbose
return_gobject	boolean: return giotto object (default = TRUE)
...	Other parameters of the triangulate function

Details

Creates a spatial Delaunay network as explained in [delaunayn](#) (default), [deldir](#), or [triangulate](#).

Value

giotto object with updated spatial network slot

Examples

```
createSpatialDelaunayNetwork(gobject)
```

createSpatialEnrich	<i>createSpatialEnrich</i>
---------------------	----------------------------

Description

Function to calculate gene signature enrichment scores per spatial position using a hypergeometric test.

Usage

```
createSpatialEnrich(
  gobject,
  enrich_method = c("PAGE", "rank", "hypergeometric"),
  sign_matrix,
  expression_values = c("normalized", "scaled", "custom"),
  reverse_log_scale = TRUE,
  logbase = 2,
  p_value = TRUE,
  n_genes = 100,
  n_times = 1000,
  top_percentage = 5,
  output_enrichment = c("original", "zscore"),
  name = "PAGE",
  return_gobject = TRUE
)
```

Arguments

<code>gobject</code>	Giotto object
<code>enrich_method</code>	method for gene signature enrichment calculation
<code>sign_matrix</code>	Matrix of signature genes for each cell type / process
<code>expression_values</code>	expression values to use
<code>reverse_log_scale</code>	reverse expression values from log scale
<code>logbase</code>	log base to use if <code>reverse_log_scale = TRUE</code>
<code>p_value</code>	calculate p-value (default = FALSE)
<code>n_times</code>	(page/rank) number of permutation iterations to calculate p-value
<code>top_percentage</code>	(hyper) percentage of cells that will be considered to have gene expression with matrix binarization
<code>output_enrichment</code>	how to return enrichment output
<code>name</code>	to give to spatial enrichment results, default = PAGE
<code>return_gobject</code>	return giotto object

Details

For details see the individual functions:

- PAGE: [PAGEEnrich](#)
- PAGE: [rankEnrich](#)
- PAGE: [hyperGeometricEnrich](#)

Value

Giotto object or enrichment results if `return_gobject = FALSE`

Examples

```
createSpatialEnrich(gobject)
```

<code>createSpatialGrid</code>	<i>createSpatialGrid</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. The dimension units are based on the provided spatial location units.

Value

giotto object with updated spatial grid slot

Examples

```
createSpatialGrid(gobject)
```

```
createSpatialGrid_2D    createSpatialGrid_2D
```

Description

create a spatial grid for 2D spatial data.

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. The dimension units are based on the provided spatial location units.

Value

giotto object with updated spatial grid slot

Examples

```
createSpatialGrid_2D(gobject)
```

```
createSpatialGrid_3D    createSpatialGrid_3D
```

Description

Create a spatial grid for 3D spatial data.

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. The dimension units are based on the provided spatial location units.

Value

giotto object with updated spatial grid slot

Examples

```
createSpatialGrid_3D(gobject)
```

```
createSpatialKNNnetwork
      createSpatialKNNnetwork
```

Description

Create a spatial knn network.

Usage

```
createSpatialKNNnetwork(
  gobject,
  method = "dbscan",
  dimensions = "all",
  name = "knn_network",
  k = 4,
  maximum_distance = NULL,
  minimum_k = 0,
  verbose = F,
  return_gobject = TRUE,
  ...
)
```

Arguments

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

Value

giotto object with updated spatial network slot

dimensions: default = 'all' which takes all possible dimensions. Alternatively you can provide a character vector that specifies the spatial dimensions to use, e.g. c("sdimx", "sdimy") or a numerical vector, e.g. 2:3

maximum_distance: to create a network based on maximum distance only, you also need to set k to a very high value, e.g. k = 100

Examples

```
createSpatialKNNnetwork(gobject)
```

```
createSpatialNetwork    createSpatialNetwork
```

Description

Create a spatial network based on cell centroid physical distances.

Usage

```
createSpatialNetwork(
  gobject,
  name = NULL,
  dimensions = "all",
  method = c("Delaunay", "kNN"),
  delaunay_method = c("delaunayn_geometry", "RTriangle", "deldir"),
  maximum_distance_delaunay = "auto",
  options = "Pp",
  Y = TRUE,
  j = TRUE,
  S = 0,
  minimum_k = 0,
  knn_method = "dbscan",
  k = 4,
  maximum_distance_knn = NULL,
  verbose = F,
  return_gobject = TRUE,
  ...
)
```

Arguments

gobject	giotto object
name	name for spatial network (default = 'spatial_network')
dimensions	which spatial dimensions to use (default = all)
method	which method to use to create a spatial network. (default = Delaunay)
delaunay_method	Delaunay method to use

maximum_distance_delaunay	distance cutoff for nearest neighbors to consider for Delaunay network
options	(geometry) String containing extra control options for the underlying Qhull command; see the Qhull documentation (../doc/qhull/html/qdelaun.html) for the available options. (default = 'Pp', do not report precision problems)
Y	(RTriangle) If TRUE prohibits the insertion of Steiner points on the mesh boundary.
j	(RTriangle) If TRUE jettisons vertices that are not part of the final triangulation from the output.
S	(RTriangle) Specifies the maximum number of added Steiner points.
minimum_k	minimum nearest neighbours if maximum_distance != NULL
knn_method	method to create kNN network
k	number of nearest neighbors based on physical distance
maximum_distance_knn	distance cutoff for nearest neighbors to consider for kNN network
verbose	verbose
return_gobject	boolean: return giotto object (default = TRUE)

Details

Creates a spatial network connecting single-cells based on their physical distance to each other. For Delaunay method, neighbors will be decided by delaunay triangulation and a maximum distance criteria. For kNN method, number of neighbors can be determined by k, or maximum distance from each cell with or without setting a minimum k for each cell.

dimensions: default = 'all' which takes all possible dimensions. Alternatively you can provide a character vector that specifies the spatial dimensions to use, e.g. c("sdmx", "sdimy") or a numerical vector, e.g. 2:3

Value

giotto object with updated spatial network slot

Examples

```
createSpatialNetwork(gobject)
```

```
create_2d_mesh_grid_line_obj
      create_2d_mesh_grid_line_obj
```

Description

create 2d mesh grid line object

Usage

```
create_2d_mesh_grid_line_obj(x_min, x_max, y_min, y_max, mesh_grid_n)
```

```
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_crossSection_object`
create_crossSection_object

Description

create a crossSection object

Usage

```
create_crossSection_object(  
  name = NULL,  
  method = NULL,  
  thickness_unit = NULL,  
  slice_thickness = NULL,  
  plane_equation = NULL,  
  mesh_grid_n = NULL,  
  mesh_obj = NULL,  
  cell_subset = NULL,  
  cell_subset_spatial_locations = NULL,  
  cell_subset_projection_locations = NULL,  
  cell_subset_projection_PCA = NULL,  
  cell_subset_projection_coords = NULL  
)
```

```
create_delaunayNetwork2D  
    create_delaunayNetwork2D
```

Description

Create a spatial Delaunay network.

Usage

```
create_delaunayNetwork2D(  
  gobject,  
  method = c("delaunayn_geometry", "RTriangle", "deldir"),  
  sdimx = "sdimx",  
  sdimy = "sdimy",  
  name = "delaunay_network",  
  maximum_distance = "auto",  
  minimum_k = 0,  
  options = "Pp",  
  Y = TRUE,  
  j = TRUE,  
  S = 0,  
  verbose = T,  
  return_gobject = TRUE,  
  ...  
)
```

Examples

```
create_delaunayNetwork2D(gobject)
```

```
create_delaunayNetwork3D  
    create_delaunayNetwork3D
```

Description

Create a spatial Delaunay network.

Usage

```
create_delaunayNetwork3D(  
  gobject,  
  method = "delaunayn_geometry",  
  sdimx = "sdimx",  
  sdimy = "sdimy",  
  sdimz = "sdimz",  
  name = "delaunay_network_3D",  
  maximum_distance = "auto",
```

```

    minimum_k = 0,
    options = "Pp",
    return_gobject = TRUE,
    ...
)

```

Examples

```
create_delaunayNetwork3D(gobject)
```

```
create_delaunayNetwork_deldir
      create_delaunayNetwork_deldir
```

Description

Create a spatial Delaunay network.

Usage

```

create_delaunayNetwork_deldir(
  spatial_locations,
  sdimx = "sdimx",
  sdimy = "sdimy",
  ...
)

```

Examples

```
create_delaunayNetwork_deldir(gobject)
```

```
create_delaunayNetwork_geometry
      create_delaunayNetwork_geometry
```

Description

Create a spatial Delaunay network.

Usage

```

create_delaunayNetwork_geometry(
  spatial_locations,
  sdimx = "sdimx",
  sdimy = "sdimy",
  options = "Pp",
  ...
)

```

Examples

```
create_delaunayNetwork_geometry(gobject)
```

```
create_delaunayNetwork_geometry_3D  
    create_delaunayNetwork_geometry_3D
```

Description

Create a spatial Delaunay network.

Usage

```
create_delaunayNetwork_geometry_3D(  
  spatial_locations,  
  sdimx = "sdimx",  
  sdimy = "sdimy",  
  sdimz = "sdimz",  
  options = options,  
  ...  
)
```

Examples

```
create_delaunayNetwork_geometry_3D(gobject)
```

```
create_delaunayNetwork_RTriangle  
    create_delaunayNetwork_RTriangle
```

Description

Create a spatial Delaunay network.

Usage

```
create_delaunayNetwork_RTriangle(  
  spatial_locations,  
  sdimx = "sdimx",  
  sdimy = "sdimy",  
  Y = TRUE,  
  j = TRUE,  
  S = 0,  
  ...  
)
```

Examples

```
create_delaunayNetwork_RTriangle(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,
  my_rownames = 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

create_KNNnetwork_dbscan	<i>create_KNNnetwork_dbscan</i>
--------------------------	---------------------------------

Description

Create a spatial knn network.

Usage

```
create_KNNnetwork_dbscan(
  spatial_locations,
  sdimx = "sdimx",
  sdimy = "sdimy",
  sdimz = "sdimz",
  k = 4,
  ...
)
```

Examples

```
create_KNNnetwork_dbscan(gobject)
```

create_mesh_grid_lines
<i>create_mesh_grid_lines</i>

Description

create mesh grid lines for cross section

Usage

```
create_mesh_grid_lines(  
    cell_subset_projection_locations,  
    extend_ratio,  
    mesh_grid_n  
)
```

create_spatialNetworkObject
<i>create_spatialNetworkObject</i>

Description

creates a spatial network object to store the created spatial network and additional information

Usage

```
create_spatialNetworkObject(  
    name = NULL,  
    method = NULL,  
    parameters = NULL,  
    outputObj = NULL,  
    networkDT = NULL,  
    cellShapeObj = NULL,  
    networkDT_before_filter = NULL,  
    crossSectionObjects = NULL,  
    misc = NULL  
)
```

crossSectionGenePlot *crossSectionGenePlot*

Description

Visualize cells and gene expression in a virtual cross section according to spatial coordinates

Usage

```
crossSectionGenePlot(
  gobject = NULL,
  crossSection_obj = NULL,
  name = NULL,
  spatial_network_name = "Delaunay_network",
  expression_values = c("normalized", "scaled", "custom"),
  genes,
  genes_high_color = "red",
  genes_mid_color = "white",
  genes_low_color = "darkblue",
  show_network = F,
  network_color = NULL,
  edge_alpha = NULL,
  show_grid = F,
  grid_color = NULL,
  spatial_grid_name = "spatial_grid",
  midpoint = 0,
  scale_alpha_with_expression = FALSE,
  point_shape = c("border", "no_border"),
  point_size = 1,
  point_border_col = "black",
  point_border_stroke = 0.1,
  show_legend = T,
  legend_text = 8,
  background_color = "white",
  axis_text = 8,
  axis_title = 8,
  cow_n_col = 2,
  cow_rel_h = 1,
  cow_rel_w = 1,
  cow_align = "h",
  show_plot = NA,
  return_plot = NA,
  save_plot = NA,
  save_param = list(),
  default_save_name = "crossSectionGenePlot"
)
```

Arguments

<code>gobject</code>	giotto object
<code>name</code>	name of virtual cross section to use

spatial_network_name	name of spatial network to use
expression_values	gene expression values to use
genes	genes to show
genes_high_color	color represents high gene expression
genes_mid_color	color represents middle gene expression
genes_low_color	color represents low gene expression
show_network	show underlying spatial network
network_color	color of spatial network
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_shape	point with border or not (border or no_border)
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
legend_text	size of legend text
background_color	color of plot background
axis_text	size of axis text
axis_title	size of axis title
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_plot	show plots
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param
...	parameters for cowplot::save_plot()

Details

Description of parameters.

Value

ggplot

See Also

[spatGenePlot3D](#) and [spatGenePlot2D](#)

Examples

```
crossSectionGenePlot(gobject)
```

```
crossSectionGenePlot3D
```

```
crossSectionGenePlot3D
```

Description

Visualize cells and gene expression in a virtual cross section according to spatial coordinates

Usage

```
crossSectionGenePlot3D(
  gobject,
  crossSection_obj = NULL,
  name = NULL,
  spatial_network_name = "Delaunay_network",
  expression_values = c("normalized", "scaled", "custom"),
  genes,
  show_network = F,
  network_color = NULL,
  edge_alpha = NULL,
  show_grid = F,
  cluster_column = NULL,
  select_cell_groups = NULL,
  select_cells = NULL,
  show_other_cells = T,
  other_cell_color = alpha("lightgrey", 0),
  other_point_size = 1,
  genes_high_color = "red",
  genes_mid_color = "white",
  genes_low_color = "darkblue",
  spatial_grid_name = "spatial_grid",
  point_size = 2,
  show_legend = T,
  axis_scale = c("cube", "real", "custom"),
  custom_ratio = NULL,
```

```

    x_ticks = NULL,
    y_ticks = NULL,
    z_ticks = NULL,
    show_plot = NA,
    return_plot = NA,
    save_plot = NA,
    save_param = list(),
    default_save_name = "crossSectionGenePlot3D"
)

```

Arguments

<code>gobject</code>	giotto object
<code>name</code>	name of virtual cross section to use
<code>spatial_network_name</code>	name of spatial network to use
<code>expression_values</code>	gene expression values to use
<code>genes</code>	genes to show
<code>show_network</code>	show underlying spatial network
<code>network_color</code>	color of spatial network
<code>show_grid</code>	show spatial grid
<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>spatial_grid_name</code>	name of spatial grid to use
<code>point_size</code>	size of point (cell)
<code>show_legend</code>	show legend
<code>show_plot</code>	show plots
<code>return_plot</code>	return ggplot object
<code>save_plot</code>	directly save the plot [boolean]
<code>save_param</code>	list of saving parameters from all_plots_save_function
<code>default_save_name</code>	default save name for saving, don't change, change <code>save_name</code> in <code>save_param</code>
<code>grid_color</code>	color of spatial grid
<code>midpoint</code>	expression midpoint
<code>scale_alpha_with_expression</code>	scale expression with ggplot alpha parameter
<code>...</code>	parameters for <code>cowplot::save_plot()</code>

Details

Description of parameters.

Value

ggplot

Examples

```
crossSectionGenePlot3D(gobject)
```

crossSectionPlot

crossSectionPlot

Description

Visualize cells in a virtual cross section according to spatial coordinates

Usage

```
crossSectionPlot(
  gobject,
  crossSection_obj = NULL,
  name = NULL,
  spatial_network_name = "Delaunay_network",
  group_by = NULL,
  group_by_subset = NULL,
  sdimx = "sdimx",
  sdimy = "sdimy",
  spat_enr_names = NULL,
  cell_color = NULL,
  color_as_factor = T,
  cell_color_code = NULL,
  cell_color_gradient = c("blue", "white", "red"),
  gradient_midpoint = NULL,
  gradient_limits = NULL,
  select_cell_groups = NULL,
  select_cells = NULL,
  point_shape = c("border", "no_border"),
  point_size = 3,
  point_border_col = "black",
  point_border_stroke = 0.1,
  show_cluster_center = F,
  show_center_label = F,
  center_point_size = 4,
  center_point_border_col = "black",
  center_point_border_stroke = 0.1,
  label_size = 4,
  label_fontface = "bold",
  show_network = F,
  network_color = NULL,
  network_alpha = 1,
  show_grid = F,
  spatial_grid_name = "spatial_grid",
```



```

    grid_color = NULL,
    show_other_cells = T,
    other_cell_color = "lightgrey",
    other_point_size = 1,
    other_cells_alpha = 0.1,
    coord_fix_ratio = NULL,
    title = NULL,
    show_legend = T,
    legend_text = 8,
    legend_symbol_size = 1,
    background_color = "white",
    axis_text = 8,
    axis_title = 8,
    cow_n_col = 2,
    cow_rel_h = 1,
    cow_rel_w = 1,
    cow_align = "h",
    show_plot = NA,
    return_plot = NA,
    save_plot = NA,
    save_param = list(),
    default_save_name = "crossSectionPlot"
  )

```

Arguments

<code>gobject</code>	giotto object
<code>name</code>	name of virtual cross section to use
<code>spatial_network_name</code>	name of spatial network to use
<code>group_by_subset</code>	subset the <code>group_by</code> factor column
<code>sdimx</code>	x-axis dimension name (default = 'sdimx')
<code>sdimy</code>	y-axis dimension name (default = 'sdimy')
<code>spat_enr_names</code>	names of spatial enrichment results to include
<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>cell_color_gradient</code>	vector with 3 colors for numeric data
<code>gradient_midpoint</code>	midpoint for color gradient
<code>gradient_limits</code>	vector with lower and upper limits
<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>point_shape</code>	point with border or not (border or no_border)

point_size	size of point (cell)
point_border_col	color of border around points
point_border_stroke	stroke size of border around points
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
show_network	show underlying spatial network
network_color	color of spatial network
network_alpha	alpha of spatial network
show_grid	show spatial grid
spatial_grid_name	name of spatial grid to use
grid_color	color of spatial grid
show_other_cells	display not selected cells
other_cell_color	color of not selected cells
other_point_size	point size of not selected cells
other_cells_alpha	alpha of not selected cells
coord_fix_ratio	fix ratio between x and y-axis
title	title of plot
show_legend	show legend
legend_text	size of legend text
legend_symbol_size	size of legend symbols
background_color	color of plot background
axis_text	size of axis text
axis_title	size of axis title
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_plot	show plot

return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param
groub_by	create multiple plots based on cell annotation column

Details

Description of parameters.

Value

ggplot

See Also

[crossSectionPlot](#)

crossSectionPlot3D	<i>crossSectionPlot3D</i>
--------------------	---------------------------

Description

Visualize cells in a virtual cross section according to spatial coordinates

Usage

```
crossSectionPlot3D(
  gobject,
  crossSection_obj = NULL,
  name = NULL,
  spatial_network_name = "Delaunay_network",
  sdimx = "sdimx",
  sdimy = "sdimy",
  sdimz = "sdimz",
  point_size = 3,
  cell_color = NULL,
  cell_color_code = NULL,
  select_cell_groups = NULL,
  show_other_cells = T,
  other_cell_color = alpha("lightgrey", 0),
  other_point_size = 0.5,
  show_network = F,
  network_color = NULL,
  network_alpha = 1,
  other_cell_alpha = 0.5,
  show_grid = F,
  grid_color = NULL,
  spatial_grid_name = "spatial_grid",
  title = "",
```

```

show_legend = T,
axis_scale = c("cube", "real", "custom"),
custom_ratio = NULL,
x_ticks = NULL,
y_ticks = NULL,
z_ticks = NULL,
show_plot = NA,
return_plot = NA,
save_plot = NA,
save_param = list(),
default_save_name = "crossSection3D"
)

```

Arguments

<code>gobject</code>	giotto object
<code>name</code>	name of virtual cross section to use
<code>spatial_network_name</code>	name of spatial network to use
<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 = '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>select_cell_groups</code>	select subset of cells/clusters based on <code>cell_color</code> parameter
<code>show_other_cells</code>	display not selected cells
<code>other_cell_color</code>	color of not selected cells
<code>other_point_size</code>	point size of not selected cells
<code>network_color</code>	color of spatial network
<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>title</code>	title of plot
<code>show_legend</code>	show legend
<code>axis_scale</code>	the way to scale the axis
<code>custom_ratio</code>	customize the scale of the plot
<code>x_ticks</code>	set the number of ticks on the x-axis
<code>y_ticks</code>	set the number of ticks on the y-axis
<code>z_ticks</code>	set the number of ticks on the z-axis
<code>show_plot</code>	show plot

return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param

Details

Description of parameters.

Value

ggplot

Examples

```
crossSectionPlot3D(gobject)
```

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)
```

```
detectSpatialCorGenes  detectSpatialCorGenes
```

Description

Detect genes that are spatially correlated

Usage

```
detectSpatialCorGenes(
  gobject,
  method = c("grid", "network"),
  expression_values = c("normalized", "scaled", "custom"),
  subset_genes = NULL,
  spatial_network_name = "Delaunay_network",
  network_smoothing = NULL,
  spatial_grid_name = "spatial_grid",
  min_cells_per_grid = 4,
  cor_method = c("pearson", "kendall", "spearman")
)
```

Arguments

<code>gobject</code>	giotto object
<code>method</code>	method to use for spatial averaging
<code>expression_values</code>	gene expression values to use
<code>subset_genes</code>	subset of genes to use
<code>spatial_network_name</code>	name of spatial network to use
<code>network_smoothing</code>	smoothing factor between 0 and 1 (default: automatic)
<code>spatial_grid_name</code>	name of spatial grid to use
<code>min_cells_per_grid</code>	minimum number of cells to consider a grid
<code>b</code>	smoothing factor between 0 and 1 (default: automatic)

Details

For method = network, it expects a fully connected spatial network. You can make sure to create a fully connected network by setting minimal_k > 0 in the [createSpatialNetwork](#) function.

- 1. grid-averaging: average gene expression values within a predefined spatial grid
- 2. network-averaging: smoothens the gene expression matrix by averaging the expression within one cell by using the neighbours within the predefined spatial network. b is a smoothening factor that defaults to $1 - 1/k$, where k is the median number of k-neighbors in the selected spatial network. Setting b = 0 means no smoothing and b = 1 means no contribution from its own expression.

The spatCorObject can be further explored with showSpatialCorGenes()

Value

returns a spatial correlation object: "spatCorObject"

See Also

[showSpatialCorGenes](#)

Examples

```
detectSpatialCorGenes(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

Steps to identify spatial patterns:

- 1. average gene expression for cells within a grid, see `createSpatialGrid`
- 2. perform PCA on the average grid expression profiles
- 3. convert variance of principal components (PCs) to z-scores and select PCs based on a z-score threshold

Value

spatial pattern object 'spatPatObj'

Examples

```
detectSpatialPatterns(gobject)
```

<code>dimCellPlot</code>	<i>dimCellPlot</i>
--------------------------	--------------------

Description

Visualize cells according to dimension reduction coordinates

Usage

```
dimCellPlot(  
  gobject,  
  dim_reduction_to_use = "umap",  
  dim_reduction_name = "umap",  
  dim1_to_use = 1,  
  dim2_to_use = 2,  
  spat_enr_names = NULL,  
  cell_annotation_values = NULL,  
  show_NN_network = F,  
  nn_network_to_use = "sNN",  
)
```



```

network_name = "sNN.pca",
cell_color_gradient = c("blue", "white", "red"),
gradient_midpoint = NULL,
gradient_limits = 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_shape = c("border", "no_border"),
point_size = 1,
point_border_col = "black",
point_border_stroke = 0.1,
show_legend = T,
legend_text = 8,
legend_symbol_size = 1,
background_color = "white",
axis_text = 8,
axis_title = 8,
cow_n_col = 2,
cow_rel_h = 1,
cow_rel_w = 1,
cow_align = "h",
show_plot = NA,
return_plot = NA,
save_plot = NA,
save_param = list(),
default_save_name = "dimCellPlot"
)

```

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>spat_enr_names</code>	names of spatial enrichment results to include
<code>cell_annotation_values</code>	numeric cell annotation columns
<code>show_NN_network</code>	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_gradient	vector with 3 colors for numeric data
gradient_midpoint	midpoint for color gradient
gradient_limits	vector with lower and upper limits
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_shape	point with border or not (border or no_border)
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
legend_text	size of legend text
legend_symbol_size	size of legend symbols
background_color	color of plot background
axis_text	size of axis text
axis_title	size of axis title
show_plot	show plot
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function

default_save_name	default save name for saving, don't change, change save_name in save_param
cell_color	color for cells (see details)
color_as_factor	convert color column to factor
cell_color_code	named vector with colors
title	title for plot, defaults to cell_color parameter

Details

Description of parameters. For 3D plots see [dimCellPlot2D](#)

Value

ggplot

Examples

dimCellPlot(gobject)

dimCellPlot2D	<i>dimCellPlot2D</i>
---------------	----------------------

Description

Visualize cells according to dimension reduction coordinates

Usage

```
dimCellPlot2D(  
  gobject,  
  dim_reduction_to_use = "umap",  
  dim_reduction_name = "umap",  
  dim1_to_use = 1,  
  dim2_to_use = 2,  
  spat_enr_names = NULL,  
  cell_annotation_values = NULL,  
  show_NN_network = F,  
  nn_network_to_use = "sNN",  
  network_name = "sNN.pca",  
  cell_color_gradient = c("blue", "white", "red"),  
  gradient_midpoint = NULL,  
  gradient_limits = 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_shape = c("border", "no_border"),
point_size = 1,
point_border_col = "black",
point_border_stroke = 0.1,
show_legend = T,
legend_text = 8,
legend_symbol_size = 1,
background_color = "white",
axis_text = 8,
axis_title = 8,
cow_n_col = 2,
cow_rel_h = 1,
cow_rel_w = 1,
cow_align = "h",
show_plot = NA,
return_plot = NA,
save_plot = NA,
save_param = list(),
default_save_name = "dimCellPlot2D"
)

```

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>spat_enr_names</code>	names of spatial enrichment results to include
<code>cell_annotation_values</code>	numeric cell annotation columns
<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_gradient</code>	vector with 3 colors for numeric data
<code>gradient_midpoint</code>	midpoint for color gradient
<code>gradient_limits</code>	vector with lower and upper limits

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_shape	point with border or not (border or no_border)
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
legend_text	size of legend text
legend_symbol_size	size of legend symbols
background_color	color of plot background
axis_text	size of axis text
axis_title	size of axis title
show_plot	show plot
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param
cell_color	color for cells (see details)
color_as_factor	convert color column to factor
cell_color_code	named vector with colors
title	title for plot, defaults to cell_color parameter

Details

Description of parameters. For 3D plots see [dimPlot3D](#)

Value

ggplot

Examples

```
dimCellPlot2D(gobject)
```

dimGenePlot	<i>dimGenePlot</i>
-------------	--------------------

Description

Visualize cells and gene expression according to dimension reduction coordinates

Usage

```
dimGenePlot(
  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 = FALSE,
  point_shape = c("border", "no_border"),
  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,
  show_legend = T,
  legend_text = 8,
  background_color = "white",
  axis_text = 8,
  axis_title = 8,
  cow_n_col = 2,
  cow_rel_h = 1,
  cow_rel_w = 1,
  cow_align = "h",
```

```

    show_plot = NA,
    return_plot = NA,
    save_plot = NA,
    save_param = list(),
    default_save_name = "dimGenePlot"
)

```

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
<code>midpoint</code>	size of point (cell)
<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_plot</code>	show plots
<code>return_plot</code>	return ggplot object
<code>save_plot</code>	directly save the plot [boolean]
<code>save_param</code>	list of saving parameters from all_plots_save_function
<code>default_save_name</code>	default save name for saving, don't change, change <code>save_name</code> in <code>save_param</code>
<code>...</code>	parameters for <code>cowplot::save_plot()</code>

Details

Description of parameters.

Value

ggplot

See Also

[dimGenePlot3D](#)

Examples

```
dimGenePlot(gobject)
```

dimGenePlot2D

dimGenePlot2D

Description

Visualize cells and gene expression according to dimension reduction coordinates

Usage

```
dimGenePlot2D(
  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 = FALSE,
  point_shape = c("border", "no_border"),
  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,
  show_legend = T,
  legend_text = 8,
  background_color = "white",
  axis_text = 8,
  axis_title = 8,
```



```

cow_n_col = 2,
cow_rel_h = 1,
cow_rel_w = 1,
cow_align = "h",
show_plot = NA,
return_plot = NA,
save_plot = NA,
save_param = list(),
default_save_name = "dimGenePlot2D"
)

```

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_shape</code>	point with border or not (border or no_border)
<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>midpoint</code>	size of point (cell)
<code>show_legend</code>	show legend
<code>legend_text</code>	size of legend text
<code>background_color</code>	color of plot background
<code>axis_text</code>	size of axis text
<code>axis_title</code>	size of axis title
<code>cow_n_col</code>	cowplot param: how many columns
<code>cow_rel_h</code>	cowplot param: relative height

cow_rel_w	cowplot param: relative width
cow_align	cowplot param: how to align
show_plot	show plots
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param
...	parameters for cowplot::save_plot()

Details

Description of parameters.

Value

ggplot

See Also

[dimGenePlot3D](#)

Examples

dimGenePlot2D(gobject)

dimGenePlot3D	<i>dimGenePlot3D</i>
---------------	----------------------

Description

Visualize cells and gene expression according to dimension reduction coordinates

Usage

```
dimGenePlot3D(  
  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",  
  cluster_column = NULL,  
  select_cell_groups = NULL,
```

```

    select_cells = NULL,
    show_other_cells = T,
    other_cell_color = "lightgrey",
    other_point_size = 1,
    edge_alpha = NULL,
    point_size = 2,
    genes_high_color = NULL,
    genes_mid_color = "white",
    genes_low_color = "blue",
    show_legend = T,
    show_plot = NA,
    return_plot = NA,
    save_plot = NA,
    save_param = list(),
    default_save_name = "dimGenePlot3D"
)

```

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>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>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
<code>show_plot</code>	show plots
<code>return_plot</code>	return ggplot object
<code>save_plot</code>	directly save the plot [boolean]
<code>save_param</code>	list of saving parameters from all_plots_save_function
<code>default_save_name</code>	default save name for saving, don't change, change <code>save_name</code> in <code>save_param</code>
<code>...</code>	parameters for <code>cowplot::save_plot()</code>

Details

Description of parameters.

Value

ggplot

Examples

dimGenePlot3D(gobject)

dimPlot	<i>dimPlot</i>
---------	----------------

Description

Visualize cells according to dimension reduction coordinates

Usage

```
dimPlot(  
  gobject,  
  group_by = NULL,  
  group_by_subset = NULL,  
  dim_reduction_to_use = "umap",  
  dim_reduction_name = "umap",  
  dim1_to_use = 1,  
  dim2_to_use = 2,  
  spat_enr_names = 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,  
  cell_color_gradient = c("blue", "white", "red"),  
  gradient_midpoint = NULL,  
  gradient_limits = 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_shape = c("border", "no_border"),  
  point_size = 1,  
  point_border_col = "black",  
  point_border_stroke = 0.1,
```

```

    show_legend = T,
    legend_text = 8,
    legend_symbol_size = 1,
    background_color = "white",
    axis_text = 8,
    axis_title = 8,
    title = NULL,
    cow_n_col = 2,
    cow_rel_h = 1,
    cow_rel_w = 1,
    cow_align = "h",
    show_plot = NA,
    return_plot = NA,
    save_plot = NA,
    save_param = list(),
    default_save_name = "dimPlot"
)

```

Arguments

<code>gobject</code>	giotto object
<code>group_by_subset</code>	subset the <code>group_by</code> factor column
<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>spat_enr_names</code>	names of spatial enrichment results to include
<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>cell_color_gradient</code>	vector with 3 colors for numeric data
<code>gradient_midpoint</code>	midpoint for color gradient
<code>gradient_limits</code>	vector with lower and upper limits
<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

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_shape	point with border or not (border or no_border)
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
legend_text	size of legend text
legend_symbol_size	size of legend symbols
background_color	color of plot background
axis_text	size of axis text
axis_title	size of axis title
title	title for plot, defaults to cell_color parameter
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_plot	show plot
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param
groub_by	create multiple plots based on cell annotation column

Details

Description of parameters, see [dimPlot2D](#). For 3D plots see [dimPlot3D](#)

Value

ggplot

Examples

```
dimPlot(gobject)
```

dimPlot2D	<i>dimPlot2D</i>
-----------	------------------

Description

Visualize cells according to dimension reduction coordinates

Usage

```
dimPlot2D(
  gobject,
  group_by = NULL,
  group_by_subset = NULL,
  dim_reduction_to_use = "umap",
  dim_reduction_name = "umap",
  dim1_to_use = 1,
  dim2_to_use = 2,
  spat_enr_names = 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,
  cell_color_gradient = c("blue", "white", "red"),
  gradient_midpoint = NULL,
  gradient_limits = 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_shape = c("border", "no_border"),
  point_size = 1,
  point_border_col = "black",
  point_border_stroke = 0.1,
```

```

    title = NULL,
    show_legend = T,
    legend_text = 8,
    legend_symbol_size = 1,
    background_color = "white",
    axis_text = 8,
    axis_title = 8,
    cow_n_col = 2,
    cow_rel_h = 1,
    cow_rel_w = 1,
    cow_align = "h",
    show_plot = NA,
    return_plot = NA,
    save_plot = NA,
    save_param = list(),
    default_save_name = "dimPlot2D"
  )

```

Arguments

<code>gobject</code>	giotto object
<code>group_by_subset</code>	subset the <code>group_by</code> factor column
<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>spat_enr_names</code>	names of spatial enrichment results to include
<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</code> = TRUE
<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>cell_color_gradient</code>	vector with 3 colors for numeric data
<code>gradient_midpoint</code>	midpoint for color gradient
<code>gradient_limits</code>	vector with lower and upper limits
<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

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_shape	point with border or not (border or no_border)
point_size	size of point (cell)
point_border_col	color of border around points
point_border_stroke	stroke size of border around points
title	title for plot, defaults to cell_color parameter
show_legend	show legend
legend_text	size of legend text
legend_symbol_size	size of legend symbols
background_color	color of plot background
axis_text	size of axis text
axis_title	size of axis title
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_plot	show plot
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param
groub_by	create multiple plots based on cell annotation column

Details

Description of parameters. For 3D plots see [dimPlot3D](#)

Value

ggplot

Examples

```
dimPlot2D(gobject)
```

dimPlot2D_single

dimPlot2D_single

Description

Visualize cells according to dimension reduction coordinates

Usage

```
dimPlot2D_single(
  gobject,
  dim_reduction_to_use = "umap",
  dim_reduction_name = "umap",
  dim1_to_use = 1,
  dim2_to_use = 2,
  spat_enr_names = 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,
  cell_color_gradient = c("blue", "white", "red"),
  gradient_midpoint = NULL,
  gradient_limits = 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_shape = c("border", "no_border"),
  point_size = 1,
  point_border_col = "black",
  point_border_stroke = 0.1,
  title = NULL,
  show_legend = T,
```

```

    legend_text = 8,
    legend_symbol_size = 1,
    background_color = "white",
    axis_text = 8,
    axis_title = 8,
    show_plot = NA,
    return_plot = NA,
    save_plot = NA,
    save_param = list(),
    default_save_name = "dimPlot2D_single"
)

```

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>spat_enr_names</code>	names of spatial enrichment results to include
<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>cell_color_gradient</code>	vector with 3 colors for numeric data
<code>gradient_midpoint</code>	midpoint for color gradient
<code>gradient_limits</code>	vector with lower and upper limits
<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>other_point_size</code>	size of not selected cells
<code>show_cluster_center</code>	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_shape	point with border or not (border or no_border)
point_size	size of point (cell)
point_border_col	color of border around points
point_border_stroke	stroke size of border around points
title	title for plot, defaults to cell_color parameter
show_legend	show legend
legend_text	size of legend text
legend_symbol_size	size of legend symbols
background_color	color of plot background
axis_text	size of axis text
axis_title	size of axis title
show_plot	show plot
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param

Details

Description of parameters. For 3D plots see [dimPlot3D](#)

Value

ggplot

Examples

```
dimPlot2D_single(gobject)
```

dimPlot3D

dimPlot3D

Description

Visualize cells according to dimension reduction coordinates

Usage

```
dimPlot3D(
  gobject,
  dim_reduction_to_use = "umap",
  dim_reduction_name = "umap",
  dim1_to_use = 1,
  dim2_to_use = 2,
  dim3_to_use = 3,
  select_cell_groups = NULL,
  select_cells = NULL,
  show_other_cells = T,
  other_cell_color = "lightgrey",
  other_point_size = 2,
  show_NN_network = F,
  nn_network_to_use = "sNN",
  network_name = "sNN.pca",
  color_as_factor = T,
  cell_color = NULL,
  cell_color_code = NULL,
  show_cluster_center = F,
  show_center_label = T,
  center_point_size = 4,
  label_size = 4,
  edge_alpha = NULL,
  point_size = 3,
  show_plot = NA,
  return_plot = NA,
  save_plot = NA,
  save_param = list(),
  default_save_name = "dim3D"
)
```

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>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>other_point_size</code>	size of not selected cells
<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>color_as_factor</code>	convert color column to factor
<code>cell_color</code>	color for cells (see details)
<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_plot</code>	show plot
<code>return_plot</code>	return ggplot object
<code>save_plot</code>	directly save the plot [boolean]
<code>save_param</code>	list of saving parameters from all_plots_save_function
<code>default_save_name</code>	default save name for saving, don't change, change <code>save_name</code> in <code>save_param</code>
<code>show_legend</code>	show legend

Details

Description of parameters.

Value

plotly

Examples

```
dimPlot3D(gobject)
```

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

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

Details

Description on how to use Kmeans clustering method.

Value

giotto object with new clusters appended to cell metadata

See Also

[hclust](#)

Examples

```
doHclust(gobject)
```

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

Description

Run HMRF

Usage

```
doHMRF(
  gobject,
  expression_values = c("normalized", "scaled", "custom"),
  spatial_network_name = "Delaunay_network",
  spatial_genes = NULL,
  spatial_dimensions = c("sdimx", "sdimy", "sdimz"),
  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,
  overwrite_output = TRUE
)
```

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
spatial_dimensions select spatial dimensions to use, default is all possible dimensions
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
overwrite_output overwrite output folder

Details

Description of HMRF parameters ...

Value

Creates a directory with results that can be viewed with viewHMRResults

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

Details

Description on how to use Kmeans clustering method.

Value

giotto object with new clusters appended to cell metadata

See Also

[kmeans](#)

Examples

```
doKmeans(gobject)
```

doLeidenCluster	<i>doLeidenCluster</i>
-----------------	------------------------

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 to use for edges
<code>partition_type</code>	The type of partition to use for optimisation.
<code>init_membership</code>	initial membership of cells for the partition
<code>n_iterations</code>	number of iterations to run the Leiden algorithm. If the number of iterations is negative, the Leiden algorithm is run until an iteration in which there was no improvement.
<code>return_gobject</code>	boolean: return giotto object (default = TRUE)
<code>set_seed</code>	set seed
<code>seed_number</code>	number for seed

Details

This function is a wrapper for the Leiden algorithm implemented in python, which can detect communities in graphs of millions of nodes (cells), as long as they can fit in memory. See the <https://github.com/vtraag/leidenalg> github page or the <https://leidenalg.readthedocs.io/en/stable/index.html> readthedocs page for more information.

Partition types available and information:

- **RBConfigurationVertexPartition**: Implements Reichardt and Bornholdt's Potts model with a configuration null model. This quality function is well-defined only for positive edge weights. This quality function uses a linear resolution parameter.
- **ModularityVertexPartition**: Implements modularity. This quality function is well-defined only for positive edge weights. It does *not* use the resolution parameter

Set `weight_col = NULL` to give equal weight (=1) to each edge.

Value

giotto object with new clusters appended to cell metadata

Examples

```
doLeidenCluster(gobject)
```

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

Description

Further 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 to run the Leiden algorithm.
<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

Details

This function performs subclustering using the Leiden algorithm on selected clusters. The systematic steps are:

- 1. subset Giotto object
- 2. identify highly variable genes
- 3. run PCA
- 4. create nearest neighbouring network
- 5. do Leiden clustering

Value

giotto object with new subclusters appended to cell metadata

See Also[doLeidenCluster](#)**Examples**

```
doLeidenSubCluster(gobject)
```

doLouvainCluster

*doLouvainCluster***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	[community] specify specific path to python if required
resolution	[community] resolution
gamma	[multinet] Resolution parameter for modularity in the generalized louvain method.
omega	[multinet] Inter-layer weight parameter in the generalized louvain method.
return_gobject	boolean: return giotto object (default = TRUE)
set_seed	set seed
seed_number	number for seed

Details

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

Value

giotto object with new clusters appended to cell metadata

See Also

[doLouvainCluster_community](#) and [doLouvainCluster_multinet](#)

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

weight_col	weight column to use for edges
louv_random	Will randomize the node evaluation order and the community evaluation order to get different partitions at each call
return_gobject	boolean: return giotto object (default = TRUE)
set_seed	set seed
seed_number	number for seed

Details

This function is a wrapper for the Louvain algorithm implemented in Python, which can detect communities in graphs of nodes (cells). See the <https://python-louvain.readthedocs.io/en/latest/index.html> readthedocs page for more information.

Set *weight_col* = *NULL* to give equal weight (=1) to each edge.

Value

giotto object with new clusters appended to cell metadata

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",
  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>	Resolution parameter for modularity in the generalized louvain method.
<code>omega</code>	Inter-layer weight parameter in the generalized louvain method.
<code>return_gobject</code>	boolean: return giotto object (default = TRUE)
<code>set_seed</code>	set seed
<code>seed_number</code>	number for seed

Details

See [glouvain_ml](#) from the `multinet` package in R for more information.

Value

giotto object with new clusters appended to cell metadata

Examples

```
doLouvainCluster_multinet(gobject)
```

<code>doLouvainSubCluster</code>	<i>doLouvainSubCluster</i>
----------------------------------	----------------------------

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)
<code>network_name</code>	name of NN network to use
<code>return_gobject</code>	boolean: return giotto object (default = TRUE)
<code>verbose</code>	verbose

Details

This function performs subclustering using the Louvain algorithm on selected clusters. The systematic steps are:

- 1. subset Giotto object
- 2. identify highly variable genes
- 3. run PCA
- 4. create nearest neighbouring network
- 5. do Louvain clustering

Value

giotto object with new subclusters appended to cell metadata

See Also

[doLouvainCluster_multinet](#) and [doLouvainCluster_community](#)

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

<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
<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

Details

This function performs subclustering using the Louvain community algorithm on selected clusters. The systematic steps are:

- 1. subset Giotto object
- 2. identify highly variable genes
- 3. run PCA
- 4. create nearest neighbouring network
- 5. do Louvain community clustering

Value

giotto object with new subclusters appended to cell metadata

See Also

[doLouvainCluster_community](#)

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

nn_param	parameters for parameters for createNearestNetwork
k_neighbors	number of k for createNearestNetwork
gamma	gamma
omega	omega
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
python_path	specify specific path to python if required

Details

This function performs subclustering using the Louvain multinet algorithm on selected clusters. The systematic steps are:

- 1. subset Giotto object
- 2. identify highly variable genes
- 3. run PCA
- 4. create nearest neighbouring network
- 5. do Louvain multinet clustering

Value

giotto object with new subclusters appended to cell metadata

See Also

[doLouvainCluster_multinet](#)

Examples

```
doLouvainSubCluster_multinet(gobject)
```

doRandomWalkCluster *doRandomWalkCluster*

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

Details

See [cluster_walktrap](#) function from the `igraph` package in R for more information.

Value

giotto object with new clusters appended to cell metadata

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

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

Details

See [sNNclust](#) from dbscan package

Value

giotto object with new clusters appended to cell metadata

Examples

```
doSNNCluster(gobject)
```

```
do_cell_proximity_test
```

```
do_cell_proximity_test
```

Description

Performs a selected differential test on subsets of a matrix

Usage

```
do_cell_proximity_test(
  expr_values,
  select_ind,
  other_ind,
  diff_test = c("permutation", "limma", "t.test", "wilcox"),
  mean_method = c("arithmetic", "geometric"),
  offset = 0.1,
  n_perm = 100,
  adjust_method = c("bonferroni", "BH", "holm", "hochberg", "hommel", "BY", "fdr",
    "none"),
  cores = 2
)
```

Examples

```
do_cell_proximity_test()
```

```
do_limimatest
```

```
do_limimatest
```

Description

Performs limma t.test on subsets of a matrix

Usage

```
do_limimatest(expr_values, select_ind, other_ind, mean_method, offset = 0.1)
```

Examples

```
do_limimatest()
```

do_multi_permuttest_random	
	<i>do_multi_permuttest_random</i>

Description

calculate multiple random values

Usage

```
do_multi_permuttest_random(  
  expr_values,  
  select_ind,  
  other_ind,  
  mean_method,  
  offset = 0.1,  
  n = 100,  
  cores = 2  
)
```

Examples

```
do_multi_permuttest_random()
```

do_page_permutation	<i>do_page_permutation</i>
---------------------	----------------------------

Description

creates permutation for the PAGEEnrich test

Usage

```
do_page_permutation(gobject, sig_gene, ntimes)
```

Examples

```
do_page_permutation()
```

do_permuttest_original	<i>do_permuttest_original</i>
------------------------	-------------------------------

Description

calculate original values

Usage

```
do_permuttest_original(  
  expr_values,  
  select_ind,  
  other_ind,  
  name = "orig",  
  mean_method,  
  offset = 0.1  
)
```

Examples

```
do_permuttest_original()
```

do_permuttest_random	<i>do_permuttest_random</i>
----------------------	-----------------------------

Description

calculate random values

Performs permutation test on subsets of a matrix

Usage

```
do_permuttest_random(  
  expr_values,  
  select_ind,  
  other_ind,  
  name = "perm_1",  
  mean_method,  
  offset = 0.1  
)
```

```
do_permuttest(  
  expr_values,  
  select_ind,  
  other_ind,  
  n_perm = 1000,  
  adjust_method = "fdr",  
  mean_method,
```

```

    offset = 0.1,
    cores = 2
)

```

Examples

```

do_permuttest_random()
do_permuttest_random()

```

do_rank_permutation	<i>do_rank_permutation</i>
---------------------	----------------------------

Description

creates permutation for the rankEnrich test

Usage

```
do_rank_permutation(sc_gene, n)
```

Examples

```
do_rank_permutation()
```

do_spatial_grid_averaging	<i>do_spatial_grid_averaging</i>
---------------------------	----------------------------------

Description

smooth gene expression over a defined spatial grid

Usage

```

do_spatial_grid_averaging(
  gobject,
  expression_values = c("normalized", "scaled", "custom"),
  subset_genes = NULL,
  spatial_grid_name = "spatial_grid",
  min_cells_per_grid = 4
)

```

Arguments

gobject	giotto object
expression_values	gene expression values to use
subset_genes	subset of genes to use
spatial_grid_name	name of spatial grid to use
min_cells_per_grid	minimum number of cells to consider a grid

Value

matrix with smoothened gene expression values based on spatial grid

Examples

```
do_spatial_grid_averaging(gobject)
```

```
do_spatial_knn_smoothing
do_spatial_knn_smoothing
```

Description

smooth gene expression over a kNN spatial network

Usage

```
do_spatial_knn_smoothing(
  gobject,
  expression_values = c("normalized", "scaled", "custom"),
  subset_genes = NULL,
  spatial_network_name = "Delaunay_network",
  b = NULL
)
```

Arguments

gobject	giotto object
expression_values	gene expression values to use
subset_genes	subset of genes to use
spatial_network_name	name of spatial network to use
b	smoothing factor between 0 and 1 (default: automatic)

Details

This function will smoothen the gene expression values per cell according to its neighbors in the selected spatial network.

b is a smoothening factor that defaults to $1 - 1/k$, where k is the median number of k-neighbors in the selected spatial network. Setting b = 0 means no smoothing and b = 1 means no contribution from its own expression.

Value

matrix with smoothened gene expression values based on kNN spatial network

Examples

```
do_spatial_knn_smoothing(gobject)
```

`do_ttest`*do_ttest*

Description

Performs t.test on subsets of a matrix

Performs wilcoxon on subsets of a matrix

Usage

```
do_ttest(  
  expr_values,  
  select_ind,  
  other_ind,  
  adjust_method,  
  mean_method,  
  offset = 0.1  
)
```

```
do_wilctest(  
  expr_values,  
  select_ind,  
  other_ind,  
  adjust_method,  
  mean_method,  
  offset = 0.1  
)
```

Examples

```
do_ttest()  
do_ttest()
```

`DT_removeNA`*DT_removeNA*

Description

set NA values to 0 in a data.table object

Usage

```
DT_removeNA(DT)
```

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

Description

converts data.table to matrix

Usage

```
dt_to_matrix(x)
```

Examples

```
dt_to_matrix(x)
```

estimateCellCellDistance	<i>estimateCellCellDistance</i>
--------------------------	---------------------------------

Description

estimate average distance between neighboring cells

Usage

```
estimateCellCellDistance(  
  gobject,  
  spatial_network_name = "Delaunay_network",  
  method = c("mean", "median")  
)
```

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

Description

compute highly variable genes

Usage

```
exportGiottoViewer(
  gobject,
  output_directory = NULL,
  spat_enr_names = NULL,
  factor_annotations = NULL,
  numeric_annotations = NULL,
  dim_reductions,
  dim_reduction_names,
  expression_values = c("scaled", "normalized", "custom"),
  dim_red_rounding = NULL,
  dim_red_rescale = c(-20, 20),
  expression_rounding = 2,
  overwrite_dir = T,
  verbose = T
)
```

Arguments

<code>gobject</code>	giotto object
<code>output_directory</code>	directory where to save the files
<code>spat_enr_names</code>	spatial enrichment results to include for annotations
<code>factor_annotations</code>	giotto cell annotations to view as factor
<code>numeric_annotations</code>	giotto cell annotations to view as numeric
<code>dim_reductions</code>	high level dimension reductions to view
<code>dim_reduction_names</code>	specific dimension reduction names
<code>expression_values</code>	expression values to use in Viewer
<code>dim_red_rounding</code>	numerical indicating how to round the coordinates
<code>dim_red_rescale</code>	numericals to rescale the coordinates
<code>expression_rounding</code>	numerical indicating how to round the expression data
<code>overwrite_dir</code>	overwrite files in the directory if it already existed
<code>verbose</code>	be verbose

Details

Giotto Viewer expects the results from Giotto Analyzer in a specific format, which is provided by this function. To include enrichment results from [createSpatialEnrich](#) include the provided spatial enrichment name (default PAGE or rank) and add the gene signature names (.e.g cell types) to the numeric annotations parameter.

Value

writes the necessary output to use in Giotto Viewer

Examples

```
exportGiottoViewer(gobject)
```

exprCellCellcom	<i>exprCellCellcom</i>
-----------------	------------------------

Description

Cell-Cell communication scores based on expression only

Usage

```
exprCellCellcom(
  gobject,
  cluster_column = "cell_types",
  random_iter = 1000,
  gene_set_1,
  gene_set_2,
  log2FC_addendum = 0.1,
  adjust_method = c("fdr", "bonferroni", "BH", "holm", "hochberg", "hommel", "BY",
    "none"),
  adjust_target = c("genes", "cells"),
  verbose = T
)
```

Arguments

gobject	giotto object to use
cluster_column	cluster column with cell type information
random_iter	number of iterations
gene_set_1	first specific gene set from gene pairs
gene_set_2	second specific gene set from gene pairs
log2FC_addendum	addendum to add when calculating log2FC
adjust_method	which method to adjust p-values
adjust_target	adjust multiple hypotheses at the cell or gene level
verbose	verbose

Details

Statistical framework to identify if pairs of genes (such as ligand-receptor combinations) are expressed at higher levels than expected based on a reshuffled null distribution of gene expression values, without considering the spatial position of cells. More details will follow soon.

Value

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

Examples

```
exprCellCellcom(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

extend_vector	<i>extend_vector</i>
---------------	----------------------

Description

extend the range of a vector by a given ratio

Usage

```
extend_vector(x, extend_ratio)
```

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

Description

Extracts a NN-network from a Giotto object

Usage

```
extractNearestNetwork(
  gobject,
  nn_network_to_use = "sNN",
  network_name = "sNN.pca",
  output = c("igraph", "data.table")
)
```

Arguments

gobject	giotto object
nn_network_to_use	kNN or sNN
network_name	name of NN network to be used
output	return a igraph or data.table object

Value

igraph or data.table object

Examples

```
extractNearestNetwork(gobject)
```

fDataDT	<i>fDataDT</i>
---------	----------------

Description

show gene metadata

Usage

```
fDataDT(gobject)
```

Arguments

gobject giotto object

Value

data.table with gene metadata

Examples

```
pDataDT(gobject)
```

filterCellProximityGenes	<i>filterCellProximityGenes</i>
--------------------------	---------------------------------

Description

Filter cell proximity gene scores.

Usage

```
filterCellProximityGenes(
  cpgObject,
  min_cells = 4,
  min_cells_expr = 1,
  min_int_cells = 4,
  min_int_cells_expr = 1,
  min_fdr = 0.1,
  min_spat_diff = 0.2,
  min_log2_fc = 0.2,
  min_zscore = 2,
  zscores_column = c("cell_type", "genes"),
  direction = c("both", "up", "down")
)
```

Arguments

cpgObject	cell proximity gene score object
min_cells	minimum number of source cell type
min_cells_expr	minimum expression level for source cell type
min_int_cells	minimum number of interacting neighbor cell type
min_int_cells_expr	minimum expression level for interacting neighbor cell type
min_fdr	minimum adjusted p-value
min_spat_diff	minimum absolute spatial expression difference
min_log2_fc	minimum log2 fold-change
min_zscore	minimum z-score change
zscores_column	calculate z-scores over cell types or genes
direction	differential expression directions to keep

Value

cpgObject that contains the filtered differential gene scores

Examples

```
filterCellProximityGenes(gobject)
```

filterCombinations	<i>filterCombinations</i>
--------------------	---------------------------

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,
  return_plot = FALSE,
  save_plot = NA,
  save_param = list(),
  default_save_name = "filterCombinations"
)
```

Arguments

<code>gobject</code>	giotto object
<code>expression_values</code>	expression values to use
<code>expression_thresholds</code>	all thresholds to consider a gene expressed
<code>gene_det_in_min_cells</code>	minimum number of cells that should express a gene to consider that gene further
<code>min_det_genes_per_cell</code>	minimum number of expressed genes per cell to consider that cell further
<code>scale_x_axis</code>	ggplot transformation for x-axis (e.g. log2)
<code>x_axis_offset</code>	x-axis offset to be used together with the scaling transformation
<code>scale_y_axis</code>	ggplot transformation for y-axis (e.g. log2)
<code>y_axis_offset</code>	y-axis offset to be used together with the scaling transformation
<code>show_plot</code>	show plot
<code>return_plot</code>	return only ggplot object
<code>save_plot</code>	directly save the plot [boolean]
<code>save_param</code>	list of saving parameters from all_plots_save_function
<code>default_save_name</code>	default save name for saving, don't change, change <code>save_name</code> in <code>save_param</code>

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)
```

filterCPG

filterCPG

Description

Filter cell proximity gene scores.

Usage

```
filterCPG(
  cpgObject,
  min_cells = 4,
  min_cells_expr = 1,
  min_int_cells = 4,
  min_int_cells_expr = 1,
  min_fdr = 0.1,
  min_spat_diff = 0.2,
  min_log2_fc = 0.2,
  min_zscore = 2,
  zscores_column = c("cell_type", "genes"),
  direction = c("both", "up", "down")
)
```

Arguments

cpgObject	cell proximity gene score object
min_cells	minimum number of source cell type
min_cells_expr	minimum expression level for source cell type
min_int_cells	minimum number of interacting neighbor cell type
min_int_cells_expr	minimum expression level for interacting neighbor cell type
min_fdr	minimum adjusted p-value
min_spat_diff	minimum absolute spatial expression difference
min_log2_fc	minimum log2 fold-change
min_zscore	minimum z-score change
zscores_column	calculate z-scores over cell types or genes
direction	differential expression directions to keep

Value

cpgObject that contains the filtered differential gene scores

Examples

```
filterCPG(gobject)
```

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

Description

show gene or cell distribution after filtering on expression threshold

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 = NA,
  return_plot = NA,
  save_plot = NA,
  save_param = list(),
  default_save_name = "filterDistributions"
)
```

Arguments

<code>gobject</code>	giotto object
<code>expression_values</code>	expression values to use
<code>expression_threshold</code>	threshold to consider a gene expressed
<code>detection</code>	consider genes or cells
<code>plot_type</code>	type of plot
<code>nr_bins</code>	number of bins for histogram plot
<code>fill_color</code>	fill color for plots
<code>scale_axis</code>	ggplot transformation for axis (e.g. log2)
<code>axis_offset</code>	offset to be used together with the scaling transformation
<code>show_plot</code>	show plot
<code>return_plot</code>	return ggplot object
<code>save_plot</code>	directly save the plot [boolean]
<code>save_param</code>	list of saving parameters from all_plots_save_function
<code>default_save_name</code>	default save name for saving, don't change, change <code>save_name</code> in <code>save_param</code>

Value

ggplot object

Examples

```
filterDistributions(gobject)
```

filterGiotto

filterGiotto

Description

filter Giotto object based on expression threshold

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

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

Details

The function [filterCombinations](#) can be used to explore the effect of different parameter values.

Value

giotto object

Examples

```
filterGiotto(gobject)
```

filter_network	<i>filter_network</i>
----------------	-----------------------

Description

function to filter a spatial network

Usage

```
filter_network(networkDT, maximum_distance = NULL, minimum_k = NULL)
```

findCellProximityGenes	<i>findCellProximityGenes</i>
------------------------	-------------------------------

Description

Identifies genes that are differentially expressed due to proximity to other cell types.

Usage

```
findCellProximityGenes(
  gobject,
  expression_values = "normalized",
  selected_genes = NULL,
  cluster_column,
  spatial_network_name = "Delaunay_network",
  minimum_unique_cells = 1,
  minimum_unique_int_cells = 1,
  diff_test = c("permutation", "limma", "t.test", "wilcox"),
  mean_method = c("arithmetic", "geometric"),
  offset = 0.1,
  adjust_method = c("bonferroni", "BH", "holm", "hochberg", "hommel", "BY", "fdr",
    "none"),
  nr_permutations = 1000,
  exclude_selected_cells_from_test = T,
  do_parallel = TRUE,
  cores = NA
)
```

Arguments

gobject	giotto object
expression_values	expression values to use
selected_genes	subset of selected genes (optional)
cluster_column	name of column to use for cell types

<code>spatial_network_name</code>	name of spatial network to use
<code>minimum_unique_cells</code>	minimum number of target cells required
<code>minimum_unique_int_cells</code>	minimum number of interacting cells required
<code>diff_test</code>	which differential expression test
<code>mean_method</code>	method to use to calculate the mean
<code>offset</code>	offset value to use when calculating log2 ratio
<code>adjust_method</code>	which method to adjust p-values
<code>nr_permutations</code>	number of permutations if <code>diff_test = permutation</code>
<code>exclude_selected_cells_from_test</code>	exclude interacting cells other cells
<code>do_parallel</code>	run calculations in parallel with <code>mclapply</code>
<code>cores</code>	number of cores to use if <code>do_parallel = TRUE</code>

Details

Function to calculate if genes are differentially expressed in cell types when they interact (approximated by physical proximity) with other cell types. The results `data.table` in the `cpgObject` contains - at least - the following columns:

- `genes`: All or selected list of tested genes
- `sel`: average gene expression in the interacting cells from the target cell type
- `other`: average gene expression in the NOT-interacting cells from the target cell type
- `log2fc`: log2 fold-change between `sel` and `other`
- `diff`: spatial expression difference between `sel` and `other`
- `p.value`: associated p-value
- `p.adj`: adjusted p-value
- `cell_type`: target cell type
- `int_cell_type`: interacting cell type
- `nr_select`: number of cells for selected target cell type
- `int_nr_select`: number of cells for interacting cell type
- `nr_other`: number of other cells of selected target cell type
- `int_nr_other`: number of other cells for interacting cell type
- `unif_int`: cell-cell interaction

Value

`cpgObject` that contains the differential gene scores

Examples

```
findCellProximityGenes(gobject)
```

```
findCellProximityGenes_per_interaction
      findCellProximityGenes_per_interaction
```

Description

Identifies genes that are differentially expressed due to proximity to other cell types.

Usage

```
findCellProximityGenes_per_interaction(
  expr_values,
  cell_metadata,
  annot_spatnetwork,
  sel_int,
  minimum_unique_cells = 1,
  minimum_unique_int_cells = 1,
  exclude_selected_cells_from_test = T,
  diff_test = c("permutation", "limma", "t.test", "wilcox"),
  mean_method = c("arithmic", "geometric"),
  offset = 0.1,
  adjust_method = "bonferroni",
  nr_permutations = 100,
  cores = 1
)
```

Examples

```
findCellProximityGenes_per_interaction()
```

```
findCPG      findCPG
```

Description

Identifies genes that are differentially expressed due to proximity to other cell types.

Usage

```
findCPG(
  gobject,
  expression_values = "normalized",
  selected_genes = NULL,
  cluster_column,
  spatial_network_name = "Delaunay_network",
  minimum_unique_cells = 1,
  minimum_unique_int_cells = 1,
  diff_test = c("permutation", "limma", "t.test", "wilcox"),
  mean_method = c("arithmic", "geometric"),
```

```

offset = 0.1,
adjust_method = c("bonferroni", "BH", "holm", "hochberg", "hommel", "BY", "fdr",
  "none"),
nr_permutations = 100,
exclude_selected_cells_from_test = T,
do_parallel = TRUE,
cores = NA
)

```

Arguments

<code>gobject</code>	giotto object
<code>expression_values</code>	expression values to use
<code>selected_genes</code>	subset of selected genes (optional)
<code>cluster_column</code>	name of column to use for cell types
<code>spatial_network_name</code>	name of spatial network to use
<code>minimum_unique_cells</code>	minimum number of target cells required
<code>minimum_unique_int_cells</code>	minimum number of interacting cells required
<code>diff_test</code>	which differential expression test
<code>mean_method</code>	method to use to calculate the mean
<code>offset</code>	offset value to use when calculating log2 ratio
<code>adjust_method</code>	which method to adjust p-values
<code>nr_permutations</code>	number of permutations if <code>diff_test = permutation</code>
<code>exclude_selected_cells_from_test</code>	exclude interacting cells other cells
<code>do_parallel</code>	run calculations in parallel with <code>mclapply</code>
<code>cores</code>	number of cores to use if <code>do_parallel = TRUE</code>

Details

Function to calculate if genes are differentially expressed in cell types when they interact (approximated by physical proximity) with other cell types. The results `data.table` in the `cpgObject` contains - at least - the following columns:

- `genes`: All or selected list of tested genes
- `sel`: average gene expression in the interacting cells from the target cell type
- `other`: average gene expression in the NOT-interacting cells from the target cell type
- `log2fc`: log2 fold-change between `sel` and `other`
- `diff`: spatial expression difference between `sel` and `other`
- `p.value`: associated p-value
- `p.adj`: adjusted p-value
- `cell_type`: target cell type

- `int_cell_type`: interacting cell type
- `nr_select`: number of cells for selected target cell type
- `int_nr_select`: number of cells for interacting cell type
- `nr_other`: number of other cells of selected target cell type
- `int_nr_other`: number of other cells for interacting cell type
- `unif_int`: cell-cell interaction

Value

`cpgObject` that contains the differential gene scores

Examples

```
findCPG(gobject)
```

<code>findGiniMarkers</code>	<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 = 0.2,
  min_det_gini_score = 0.2,
  detection_threshold = 0,
  rank_score = 1,
  min_genes = 5
)
```

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>group_1</code>	group 1 cluster IDs from <code>cluster_column</code> for pairwise comparison
<code>group_2</code>	group 2 cluster IDs from <code>cluster_column</code> for pairwise comparison

min_expr_gini_score filter on minimum gini coefficient for expression
 min_det_gini_score filter on minimum gini coefficient for detection
 detection_threshold detection threshold for gene expression
 rank_score rank scores for both detection and expression to include
 min_genes minimum number of top genes to return

Details

Detection of marker genes using the https://en.wikipedia.org/wiki/Gini_coefficient gini coefficient is based on the following steps/principles per gene:

- 1. calculate average expression per cluster
- 2. calculate detection fraction per cluster
- 3. calculate gini-coefficient for av. expression values over all clusters
- 4. calculate gini-coefficient for detection fractions over all clusters
- 5. convert gini-scores to rank scores
- 6. for each gene create combined score = detection rank x expression rank x expr gini-coefficient x detection gini-coefficient
- 7. for each gene sort on expression and detection rank and combined score

As a results "top gini" genes are genes that are very selectively expressed in a specific cluster, however not always expressed in all cells of that cluster. In other words highly specific, but not necessarily sensitive at the single-cell level.

To perform differential expression between cluster groups you need to specify cluster IDs to the parameters *group_1* and *group_2*.

Value

data.table with marker genes

Examples

```
findGiniMarkers(gobject)
```

```
findGiniMarkers_one_vs_all
  findGiniMarkers_one_vs_all
```

Description

Identify marker genes for all clusters in a one vs all manner 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,  
  rank_score = 1,  
  min_genes = 4,  
  verbose = TRUE  
)
```

Arguments

gobject	giotto object
expression_values	gene expression values to use
cluster_column	clusters to use
subset_clusters	selection of clusters to compare
min_expr_gini_score	filter on minimum gini coefficient on expression
min_det_gini_score	filter on minimum gini coefficient on detection
detection_threshold	detection threshold for gene expression
rank_score	rank scores for both detection and expression to include
min_genes	minimum number of top genes to return
verbose	be verbose

Value

data.table with marker genes

See Also

[findGiniMarkers](#)

Examples

```
findGiniMarkers_one_vs_all(gobject)
```

findMarkers

*findMarkers***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,
  min_genes = 4,
  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
min_genes	minimum number of top genes to return (for gini)
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 scran or zlm function in
               MAST

```

Details

Wrapper for all individual functions to detect marker genes for clusters.

Value

data.table with marker genes

See Also

[findScranMarkers](#), [findGiniMarkers](#) and [findMastMarkers](#)

Examples

```
findMarkers(gobject)
```

```
findMarkers_one_vs_all
               findMarkers_one_vs_all
```

Description

Identify marker genes for all clusters in a one vs all manner.

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>	scan & 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 <code>findMarkers</code> function in <code>scan</code> or <code>zlm</code> function in <code>MAST</code>

Details

Wrapper for all one vs all functions to detect marker genes for clusters.

Value

data.table with marker genes

See Also

[findScranMarkers_one_vs_all](#), [findGiniMarkers_one_vs_all](#) and [findMastMarkers_one_vs_all](#)

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 [zlm](#) from the MAST package 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 in a one vs all manner 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

Value

data.table with marker genes

See Also

[findMastMarkers](#)

Examples

```
findMastMarkers_one_vs_all(gobject)
```

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

Description

Identify marker genes for all or 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

<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>group_1</code>	group 1 cluster IDs from <code>cluster_column</code> for pairwise comparison
<code>group_2</code>	group 2 cluster IDs from <code>cluster_column</code> for pairwise comparison
<code>...</code>	additional parameters for the <code>findMarkers</code> function in <code>scran</code>

Details

This is a minimal convenience wrapper around the [findMarkers](#) function from the `scran` package.

To perform differential expression between cluster groups you need to specify cluster IDs to the parameters `group_1` and `group_2`.

Value

data.table with marker genes

Examples

```
findScranMarkers(gobject)
```

```
findScranMarkers_one_vs_all
      findScranMarkers_one_vs_all
```

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

Value

data.table with marker genes

See Also

[findScranMarkers](#)

Examples

```
findScranMarkers_one_vs_all(gobject)
```

find_grid_2D	<i>find_grid_2D</i>
--------------	---------------------

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)
```

find_x_y_ranges	<i>find_x_y_ranges</i>
-----------------	------------------------

Description

get the extended ranges of x and y

Usage

```
find_x_y_ranges(data, extend_ratio)
```

general_save_function	<i>general_save_function</i>
-----------------------	------------------------------

Description

Function to automatically save plots to directory of interest

Usage

```
general_save_function(
  gobject,
  plot_object,
  save_dir = NULL,
  save_folder = NULL,
  save_name = NULL,
  default_save_name = "giotto_plot",
  save_format = c("png", "tiff", "pdf", "svg"),
  show_saved_plot = F,
  base_width = NULL,
  base_height = NULL,
  base_aspect_ratio = NULL,
  units = NULL,
  dpi = NULL,
  ...
)
```


Arguments

<code>gobject</code>	giotto object
<code>plot_object</code>	non-ggplot object to plot
<code>save_dir</code>	directory to save to
<code>save_folder</code>	folder in <code>save_dir</code> to save to
<code>save_name</code>	name of plot
<code>save_format</code>	format (e.g. png, tiff, pdf, ...)
<code>show_saved_plot</code>	load & display the saved plot
<code>base_width</code>	width
<code>base_height</code>	height
<code>base_aspect_ratio</code>	aspect ratio
<code>units</code>	units
<code>dpi</code>	Plot resolution

Examples

```
general_save_function(gobject)
```

<code>get10Xmatrix</code>	<i>get10Xmatrix</i>
---------------------------	---------------------

Description

This function creates an expression matrix from a 10X structured folder

Usage

```
get10Xmatrix(path_to_data, gene_column_index = 1)
```

Arguments

<code>path_to_data</code>	path to the 10X folder
<code>gene_column_index</code>	which column from the features or genes .tsv file to use for row ids

Details

A typical 10X folder is named `raw_feature_bc_matrix` or `raw_feature_bc_matrix` and it has 3 files:

- `barcodes.tsv(.gz)`
- `features.tsv(.gz)` or `genes.tsv(.gz)`
- `matrix.mtx(.gz)`

By default the first column of the features or genes .tsv file will be used, however if multiple annotations are provided (e.g. ensembl gene ids and gene symbols) the user can select another column.

Value

expression matrix from 10X

Examples

```
get10Xmatrix(10Xmatrix)
```

```
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	<i>getDendrogramSplits</i>
---------------------	----------------------------

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)
```

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

Description

Returns a number of distinct colors based on the RGB scale

Usage

```
getDistinctColors(n)
```

Arguments

n	number of colors wanted
---	-------------------------

Value

number of distinct colors

get_cross_section_coordinates	<i>get_cross_section_coordinates</i>
-------------------------------	--------------------------------------

Description

get local coordinates within cross section plane

Usage

```
get_cross_section_coordinates(cell_subset_projection_locations)
```

get_distance	<i>get_distance</i>
--------------	---------------------

Description

estimate average distance between neighboring cells with network table as input

Usage

```
get_distance(networkDT, method = c("mean", "median"))
```

get_sectionThickness *get_sectionThickness*

Description

get section thickness

Usage

```
get_sectionThickness(  
  gobject,  
  thickness_unit = c("cell", "natural"),  
  slice_thickness = 2,  
  spatial_network_name = "Delaunay_network",  
  cell_distance_estimate_method = c("mean", "median"),  
  plane_equation = NULL  
)
```

ggplot_save_function *ggplot_save_function*

Description

Function to automatically save plots to directory of interest

Usage

```
ggplot_save_function(  
  gobject,  
  plot_object,  
  save_dir = NULL,  
  save_folder = NULL,  
  save_name = NULL,  
  default_save_name = "giotto_plot",  
  save_format = NULL,  
  show_saved_plot = F,  
  ncol = 1,  
  nrow = 1,  
  scale = 1,  
  base_width = NULL,  
  base_height = NULL,  
  base_aspect_ratio = NULL,  
  units = NULL,  
  dpi = NULL,  
  limitsize = TRUE,  
  ...  
)
```

Arguments

<code>gobject</code>	giotto object
<code>plot_object</code>	ggplot object to plot
<code>save_dir</code>	directory to save to
<code>save_folder</code>	folder in <code>save_dir</code> to save to
<code>save_name</code>	name of plot
<code>save_format</code>	format (e.g. png, tiff, pdf, ...)
<code>show_saved_plot</code>	load & display the saved plot
<code>ncol</code>	number of columns
<code>nrow</code>	number of rows
<code>scale</code>	scale
<code>base_width</code>	width
<code>base_height</code>	height
<code>base_aspect_ratio</code>	aspect ratio
<code>units</code>	units
<code>dpi</code>	Plot resolution
<code>limitsize</code>	When TRUE (the default), ggsave will not save images larger than 50x50 inches, to prevent the common error of specifying dimensions in pixels.

See Also

[cowplot::save_plot](#)

Examples

```
ggplot_save_function(gobject)
```

giotto-class

S4 giotto Class

Description

Framework of giotto object to store and work with spatial expression data

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
 instructions slot for global function instructions
 offset_file offset file used to stitch together image fields
 OS_platform Operating System to run Giotto analysis on

heatmSpatialCorGenes *heatmSpatialCorGenes*

Description

Create heatmap of spatially correlated genes

Usage

```

heatmSpatialCorGenes(
  gobject,
  spatCorObject,
  use_clus_name = NULL,
  show_cluster_annot = TRUE,
  show_row_dend = T,
  show_column_dend = F,
  show_row_names = F,
  show_column_names = F,
  show_plot = NA,
  return_plot = NA,
  save_plot = NA,
  save_param = list(),
  default_save_name = "heatmSpatialCorGenes",
  ...
)

```

Arguments

gobject	giotto object
spatCorObject	spatial correlation object
use_clus_name	name of clusters to visualize (from clusterSpatialCorGenes())
show_cluster_annot	show cluster annotation on top of heatmap
show_row_dend	show row dendrogram
show_column_dend	show column dendrogram

```

show_row_names  show row names
show_column_names
                  show column names
show_plot       show plot
return_plot     return ggplot object
save_plot       directly save the plot [boolean]
save_param      list of saving parameters from all\_plots\_save\_function
default_save_name
                  default save name for saving, don't change, change save_name in save_param
...             additional parameters to the Heatmap function from ComplexHeatmap

```

Value

Heatmap generated by ComplexHeatmap

Examples

```
heatmSpatialCorGenes(gobject)
```

```
hyperGeometricEnrich  hyperGeometricEnrich
```

Description

Function to calculate gene signature enrichment scores per spatial position using a hypergeometric test.

Usage

```

hyperGeometricEnrich(
  gobject,
  sign_matrix,
  expression_values = c("normalized", "scaled", "custom"),
  reverse_log_scale = TRUE,
  logbase = 2,
  top_percentage = 5,
  output_enrichment = c("original", "zscore")
)

```

Arguments

```

gobject          Giotto object
sign_matrix      Matrix of signature genes for each cell type / process
expression_values
                  expression values to use
reverse_log_scale
                  reverse expression values from log scale
logbase          log base to use if reverse_log_scale = TRUE
top_percentage   percentage of cells that will be considered to have gene expression with matrix
                  binarization
output_enrichment
                  how to return enrichment output

```


Details

The enrichment score is calculated based on the p-value from the hypergeometric test, $-\log_{10}(\text{p-value})$.

Value

data.table with enrichment results

Examples

```
hyperGeometricEnrich(gobject)
```

```
insertCrossSectionGenePlot3D
```

```
insertCrossSectionGenePlot3D
```

Description

Visualize cells and gene expression in a virtual cross section according to spatial coordinates

Usage

```
insertCrossSectionGenePlot3D(
  gobject,
  crossSection_obj = NULL,
  name = NULL,
  spatial_network_name = "Delaunay_network",
  mesh_grid_color = "#1f77b4",
  mesh_grid_width = 3,
  mesh_grid_style = "dot",
  sdimx = "sdimx",
  sdimy = "sdimy",
  sdimz = "sdimz",
  expression_values = c("normalized", "scaled", "custom"),
  genes,
  show_network = F,
  network_color = NULL,
  edge_alpha = NULL,
  show_grid = F,
  cluster_column = NULL,
  select_cell_groups = NULL,
  select_cells = NULL,
  show_other_cells = F,
  other_cell_color = "lightgrey",
  other_point_size = 1,
  genes_high_color = NULL,
  genes_mid_color = "white",
  genes_low_color = "darkblue",
  spatial_grid_name = "spatial_grid",
  point_size = 2,
  show_legend = T,
```

```

axis_scale = c("cube", "real", "custom"),
custom_ratio = NULL,
x_ticks = NULL,
y_ticks = NULL,
z_ticks = NULL,
show_plot = NA,
return_plot = NA,
save_plot = NA,
save_param = list(),
default_save_name = "spatGenePlot3D_with_cross_section"
)

```

Arguments

<code>gobject</code>	giotto object
<code>name</code>	name of virtual cross section to use
<code>spatial_network_name</code>	name of spatial network to use
<code>mesh_grid_color</code>	color for the meshgrid lines
<code>mesh_grid_width</code>	width for the meshgrid lines
<code>mesh_grid_style</code>	style for the meshgrid lines
<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 = 'sdimy')
<code>expression_values</code>	gene expression values to use
<code>genes</code>	genes to show
<code>show_network</code>	show underlying spatial network
<code>network_color</code>	color of spatial network
<code>show_grid</code>	show spatial grid
<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>spatial_grid_name</code>	name of spatial grid to use
<code>point_size</code>	size of point (cell)
<code>show_legend</code>	show legend
<code>show_plot</code>	show plots
<code>return_plot</code>	return ggplot object
<code>save_plot</code>	directly save the plot [boolean]
<code>save_param</code>	list of saving parameters from all_plots_save_function

default_save_name	default save name for saving, don't change, change save_name in save_param
grid_color	color of spatial grid
midpoint	expression midpoint
scale_alpha_with_expression	scale expression with ggplot alpha parameter
...	parameters for cowplot::save_plot()

Details

Description of parameters.

Value

ggplot

Examples

```
insertCrossSectionGenePlot3D(gobject)
```

```
insertCrossSectionSpatPlot3D
```

```
insertCrossSectionSpatPlot3D
```

Description

Visualize the meshgrid lines of cross section together with cells

Usage

```
insertCrossSectionSpatPlot3D(
  gobject,
  crossSection_obj = NULL,
  name = NULL,
  spatial_network_name = "Delaunay_network",
  mesh_grid_color = "#1f77b4",
  mesh_grid_width = 3,
  mesh_grid_style = "dot",
  sdimx = "sdimx",
  sdimy = "sdimy",
  sdimz = "sdimz",
  point_size = 2,
  cell_color = NULL,
  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_network = F,
  network_color = NULL,
```

```

network_alpha = 1,
other_cell_alpha = 0.5,
show_grid = F,
grid_color = NULL,
spatial_grid_name = "spatial_grid",
title = "",
show_legend = T,
axis_scale = c("cube", "real", "custom"),
custom_ratio = NULL,
x_ticks = NULL,
y_ticks = NULL,
z_ticks = NULL,
show_plot = NA,
return_plot = NA,
save_plot = NA,
save_param = list(),
default_save_name = "spat3D_with_cross_section"
)

```

Arguments

<code>gobject</code>	giotto object
<code>name</code>	name of virtual cross section to use
<code>spatial_network_name</code>	name of spatial network to use
<code>mesh_grid_color</code>	color for the meshgrid lines
<code>mesh_grid_width</code>	width for the meshgrid lines
<code>mesh_grid_style</code>	style for the meshgrid lines
<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 = '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>select_cell_groups</code>	select subset of cells/clusters based on cell_color parameter
<code>show_other_cells</code>	display not selected cells
<code>other_cell_color</code>	color of not selected cells
<code>other_point_size</code>	point size of not selected cells
<code>network_color</code>	color of spatial network
<code>show_grid</code>	show spatial grid
<code>grid_color</code>	color of spatial grid

spatial_grid_name	name of spatial grid to use
title	title of plot
show_legend	show legend
axis_scale	the way to scale the axis
custom_ratio	customize the scale of the plot
x_ticks	set the number of ticks on the x-axis
y_ticks	set the number of ticks on the y-axis
z_ticks	set the number of ticks on the z-axis
show_plot	show plot
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param

Details

Description of parameters.

Value

ggplot

Examples

```
insertCrossSectionSpatPlot3D(gobject)
```

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

Description

create binarized scores from a vector 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 doHRMF

Examples

```
loadHMRF(gobject)
```

makeSignMatrixPAGE	<i>makeSignMatrixPAGE</i>
--------------------	---------------------------

Description

Function to convert a list of signature genes (e.g. for cell types or processes) into a binary matrix format that can be used with the PAGE enrichment option. Each cell type or process should have a vector of cell-type or process specific genes. These vectors need to be combined into a list (sign_list). The names of the cell types or processes that are provided in the list need to be given (sign_names).

Usage

```
makeSignMatrixPAGE(sign_names, sign_list)
```

Arguments

sign_names	vector with names for each provided gene signature
sign_list	list of genes (signature)

Value

matrix

See Also

[PAGEEnrich](#)

Examples

```
makeSignMatrixPAGE()
```

makeSignMatrixRank	<i>makeSignMatrixRank</i>
--------------------	---------------------------

Description

Function to convert a single-cell count matrix and a corresponding single-cell cluster vector into a rank matrix that can be used with the Rank enrichment option.

Usage

```
makeSignMatrixRank(sc_matrix, sc_cluster_ids, gobject = NULL)
```

Arguments

sc_matrix	matrix of single-cell RNAseq expression data
sc_cluster_ids	vector of cluster ids
gobject	if giotto object is given then only genes present in both datasets will be considered

Value

matrix

See Also

[rankEnrich](#)

Examples

```
makeSignMatrixRank()
```

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

Description

Simulate random network.

Usage

```
make_simulated_network(
  gobject,
  spatial_network_name = "Delaunay_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 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

mygini_fun

Description

calculate gini coefficient

Usage

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

Value

gini coefficient

my_arowMeans	<i>my_arowMeans</i>
--------------	---------------------

Description

arithmetic rowMeans that works for a single column

Usage

```
my_arowMeans(x)
```

Examples

```
my_arowMeans(x)
```

my_growMeans	<i>my_growMeans</i>
--------------	---------------------

Description

geometric rowMeans that works for a single column

Usage

```
my_growMeans(x, offset = 0.1)
```

Examples

```
my_growMeans(x)
```

my_rowMeans	<i>my_rowMeans</i>
-------------	--------------------

Description

arithmetic or geometric rowMeans that works for a single column

Usage

```
my_rowMeans(x, method = c("arithmetic", "geometric"), offset = 0.1)
```

Examples

```
my_rowMeans(x)
```

nnDT_to_kNN

nnDT_to_kNN

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

node_clusters

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

fast normalize and/or scale expression values of Giotto object

Usage

```
normalizeGiotto(
  gobject,
  norm_methods = c("standard", "osmFISH"),
  library_size_norm = TRUE,
  scalefactor = 6000,
  log_norm = TRUE,
  log_offset = 1,
  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
log_offset	offset value to add to expression matrix, default = 1
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)
```

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

Description

normalize and/or scale expresion values of Giotto object

Usage

```
normalizeGiottoOld(
  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)
```

PAGEEnrich

PAGEEnrich

Description

Function to calculate gene signature enrichment scores per spatial position using PAGE.

Usage

```
PAGEEnrich(
  gobject,
  sign_matrix,
  expression_values = c("normalized", "scaled", "custom"),
  reverse_log_scale = TRUE,
  logbase = 2,
  output_enrichment = c("original", "zscore")
)
```

Arguments

gobject Giotto object
sign_matrix Matrix of signature genes for each cell type / process
expression_values expression values to use
reverse_log_scale reverse expression values from log scale
logbase log base to use if reverse_log_scale = TRUE
output_enrichment how to return enrichment output

Details

sign_matrix: a binary matrix with genes as row names and cell-types as column names. Alternatively a list of signature genes can be provided to `makeSignMatrixPAGE`, which will create the matrix for you.

The enrichment Z score is calculated by using method (PAGE) from Kim SY et al., BMC bioinformatics, 2005 as $Z = ((Sm \sim \mu) * m^{(1/2)}) / \delta$. For each gene in each spot, μ is the fold change values versus the mean expression and δ is the standard deviation. Sm is the mean fold change value of a specific marker gene set and m is the size of a given marker gene set.

Value

data.table with enrichment results

See Also

[makeSignMatrixPAGE](#)

Examples

```
PAGEEnrich(gobject)
```

pDataDT

pDataDT

Description

show cell metadata

Usage

```
pDataDT(gobject)
```

Arguments

gobject giotto object

Value

data.table with cell metadata

Examples

```
pDataDT(gobject)
```

plotCCcomDotplot	<i>plotCCcomDotplot</i>
------------------	-------------------------

Description

Plots dotplot for ligand-receptor communication scores in cell-cell interactions

Usage

```
plotCCcomDotplot(  
  gobject,  
  comScores,  
  selected_LR = NULL,  
  selected_cell_LR = NULL,  
  show_LR_names = TRUE,  
  show_cell_LR_names = TRUE,  
  cluster_on = c("PI", "LR_expr", "log2fc"),  
  cor_method = c("pearson", "kendall", "spearman"),  
  aggl_method = c("ward.D", "ward.D2", "single", "complete", "average", "mcquitty",  
    "median", "centroid"),  
  show_plot = NA,  
  return_plot = NA,  
  save_plot = NA,  
  save_param = list(),  
  default_save_name = "plotCCcomDotplot"  
)
```

Arguments

gobject	giotto object
comScores	communication scores from exprCellCellcom or spatCellCellcom
selected_LR	selected ligand-receptor combinations
selected_cell_LR	selected cell-cell combinations for ligand-receptor combinations
show_LR_names	show ligand-receptor names
show_cell_LR_names	show cell-cell names
cluster_on	values to use for clustering of cell-cell and ligand-receptor pairs
cor_method	correlation method used for clustering
aggl_method	agglomeration method used by hclust
show_plot	show plots

return_plot	return plotting object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param
show	values to show on heatmap

Value

ggplot

Examples

```
plotCCcomDotplot(CPGscores)
```

plotCCcomHeatmap	<i>plotCCcomHeatmap</i>
------------------	-------------------------

Description

Plots heatmap for ligand-receptor communication scores in cell-cell interactions

Usage

```
plotCCcomHeatmap(
  gobject,
  comScores,
  selected_LR = NULL,
  selected_cell_LR = NULL,
  show_LR_names = TRUE,
  show_cell_LR_names = TRUE,
  show = c("PI", "LR_expr", "log2fc"),
  cor_method = c("pearson", "kendall", "spearman"),
  aggl_method = c("ward.D", "ward.D2", "single", "complete", "average", "mcquitty",
    "median", "centroid"),
  show_plot = NA,
  return_plot = NA,
  save_plot = NA,
  save_param = list(),
  default_save_name = "plotCCcomHeatmap"
)
```

Arguments

gobject	giotto object
comScores	communication scores from exprCellCellcom or spatCellCellcom
selected_LR	selected ligand-receptor combinations
selected_cell_LR	selected cell-cell combinations for ligand-receptor combinations

show_LR_names	show ligand-receptor names
show_cell_LR_names	show cell-cell names
show	values to show on heatmap
cor_method	correlation method used for clustering
aggl_method	agglomeration method used by hclust
show_plot	show plots
return_plot	return plotting object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param

Value

ggplot

Examples

```
plotCCcomHeatmap(CPGscores)
```

```
plotCellProximityGenes
      plotCellProximityGenes
```

Description

Create visualization for cell proximity gene scores

Usage

```
plotCellProximityGenes(
  gobject,
  cpgObject,
  method = c("volcano", "cell_barplot", "cell-cell", "cell_sankey", "heatmap",
    "dotplot"),
  min_cells = 4,
  min_cells_expr = 1,
  min_int_cells = 4,
  min_int_cells_expr = 1,
  min_fdr = 0.1,
  min_spat_diff = 0.2,
  min_log2_fc = 0.2,
  min_zscore = 2,
  zscores_column = c("cell_type", "genes"),
  direction = c("both", "up", "down"),
  cell_color_code = NULL,
  show_plot = NA,
  return_plot = NA,
```

```

    save_plot = NA,
    save_param = list(),
    default_save_name = "plotCellProximityGenes"
  )

```

Arguments

<code>gobject</code>	giotto object
<code>cpgObject</code>	cell proximity gene score object
<code>method</code>	plotting method to use
<code>min_cells</code>	minimum number of source cell type
<code>min_cells_expr</code>	minimum expression level for source cell type
<code>min_int_cells</code>	minimum number of interacting neighbor cell type
<code>min_int_cells_expr</code>	minimum expression level for interacting neighbor cell type
<code>min_fdr</code>	minimum adjusted p-value
<code>min_spat_diff</code>	minimum absolute spatial expression difference
<code>min_log2_fc</code>	minimum log2 fold-change
<code>min_zscore</code>	minimum z-score change
<code>zscores_column</code>	calculate z-scores over cell types or genes
<code>direction</code>	differential expression directions to keep
<code>cell_color_code</code>	vector of colors with cell types as names
<code>show_plot</code>	show plots
<code>return_plot</code>	return plotting object
<code>save_plot</code>	directly save the plot [boolean]
<code>save_param</code>	list of saving parameters from all_plots_save_function
<code>default_save_name</code>	default save name for saving, don't change, change save_name in save_param

Value

plot

Examples

```
plotCellProximityGenes(CPGscores)
```

plotCombineCCcom	<i>plotCombineCCcom</i>
------------------	-------------------------

Description

Create visualization for combined (pairwise) cell proximity gene scores

Usage

```
plotCombineCCcom(
  gobject,
  combCCcom,
  selected_LR = NULL,
  selected_cell_LR = NULL,
  detail_plot = T,
  simple_plot = F,
  simple_plot_facet = c("interaction", "genes"),
  facet_scales = "fixed",
  facet_ncol = length(selected_LR),
  facet_nrow = length(selected_cell_LR),
  colors = c("#9932CC", "#FF8C00"),
  show_plot = NA,
  return_plot = NA,
  save_plot = NA,
  save_param = list(),
  default_save_name = "plotCombineCCcom"
)
```

Arguments

gobject	giotto object
combCCcom	combined communication scores, output from combCCcom()
selected_LR	selected ligand-receptor pair
selected_cell_LR	selected cell-cell interaction pair for ligand-receptor pair
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 parameter
facet_ncol	ggplot facet ncol parameter
facet_nrow	ggplot facet nrow parameter
colors	vector with two colors to use
show_plot	show plots
return_plot	return plotting object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param

Value

ggplot

Examples

```
plotCombineCCcom(CPGscores)
```

```
plotCombineCellCellCommunication
```

```
plotCombineCellCellCommunication
```

Description

Create visualization for combined (pairwise) cell proximity gene scores

Usage

```
plotCombineCellCellCommunication(
  gobject,
  combCCcom,
  selected_LR = NULL,
  selected_cell_LR = NULL,
  detail_plot = T,
  simple_plot = F,
  simple_plot_facet = c("interaction", "genes"),
  facet_scales = "fixed",
  facet_ncol = length(selected_LR),
  facet_nrow = length(selected_cell_LR),
  colors = c("#9932CC", "#FF8C00"),
  show_plot = NA,
  return_plot = NA,
  save_plot = NA,
  save_param = list(),
  default_save_name = "plotCombineCellCellCommunication"
)
```

Arguments

gobject	giotto object
combCCcom	combined communication scores, output from combCCcom()
selected_LR	selected ligand-receptor pair
selected_cell_LR	selected cell-cell interaction pair for ligand-receptor pair
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 parameter
facet_ncol	ggplot facet ncol parameter

facet_nrow	ggplot facet nrow parameter
colors	vector with two colors to use
show_plot	show plots
return_plot	return plotting object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param

Value

ggplot

Examples

```
plotCombineCellCellCommunication(CPGscores)
```

```
plotCombineCellProximityGenes
      plotCombineCellProximityGenes
```

Description

Create visualization for combined (pairwise) cell proximity gene scores

Usage

```
plotCombineCellProximityGenes(
  gobject,
  combCpgObject,
  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("#9932CC", "#FF8C00"),
  show_plot = NA,
  return_plot = NA,
  save_plot = NA,
  save_param = list(),
  default_save_name = "plotCombineCPG"
)
```

Arguments

gobject giotto object
combCpgObject CPGscores, output from combineCellProximityGenes()
selected_interactions interactions to show
selected_gene_to_gene pairwise gene combinations 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 two colors to use
show_plot show plots
return_plot return plotting object
save_plot directly save the plot [boolean]
save_param list of saving parameters from [all_plots_save_function](#)
default_save_name default save name for saving, don't change, change save_name in save_param

Value

ggplot

Examples

```
plotCombineCellProximityGenes(CPGscores)
```

plotCombineCPG

plotCombineCPG

Description

Create visualization for combined (pairwise) cell proximity gene scores

Usage

```
plotCombineCPG(
  gobject,
  combCpgObject,
  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("#9932CC", "#FF8C00"),
show_plot = NA,
return_plot = NA,
save_plot = NA,
save_param = list(),
default_save_name = "plotCombineCPG"
)

```

Arguments

<code>gobject</code>	giotto object
<code>combCpgObject</code>	CPGscores, output from <code>combineCellProximityGenes()</code>
<code>selected_interactions</code>	interactions to show
<code>selected_gene_to_gene</code>	pairwise gene combinations to show
<code>detail_plot</code>	show detailed info in both interacting cell types
<code>simple_plot</code>	show a simplified plot
<code>simple_plot_facet</code>	facet on interactions or genes with simple plot
<code>facet_scales</code>	ggplot facet scales paramter
<code>facet_ncol</code>	ggplot facet ncol parameter
<code>facet_nrow</code>	ggplot facet nrow parameter
<code>colors</code>	vector with two colors to use
<code>show_plot</code>	show plots
<code>return_plot</code>	return plotting object
<code>save_plot</code>	directly save the plot [boolean]
<code>save_param</code>	list of saving parameters from all_plots_save_function
<code>default_save_name</code>	default save name for saving, don't change, change <code>save_name</code> in <code>save_param</code>

Value

ggplot

Examples

```
plotCombineCPG(CPGscores)
```


plotCPG

*plotCPG***Description**

Create visualization for cell proximity gene scores

Usage

```
plotCPG(
  gobject,
  cpgObject,
  method = c("volcano", "cell_barplot", "cell-cell", "cell_sankey", "heatmap",
    "dotplot"),
  min_cells = 5,
  min_cells_expr = 1,
  min_int_cells = 3,
  min_int_cells_expr = 1,
  min_fdr = 0.05,
  min_spat_diff = 0.2,
  min_log2_fc = 0.2,
  min_zscore = 2,
  zscores_column = c("cell_type", "genes"),
  direction = c("both", "up", "down"),
  cell_color_code = NULL,
  show_plot = NA,
  return_plot = NA,
  save_plot = NA,
  save_param = list(),
  default_save_name = "plotCPG"
)
```

Arguments

<code>gobject</code>	giotto object
<code>cpgObject</code>	cell proximity gene score object
<code>method</code>	plotting method to use
<code>min_cells</code>	minimum number of source cell type
<code>min_cells_expr</code>	minimum expression level for source cell type
<code>min_int_cells</code>	minimum number of interacting neighbor cell type
<code>min_int_cells_expr</code>	minimum expression level for interacting neighbor cell type
<code>min_fdr</code>	minimum adjusted p-value
<code>min_spat_diff</code>	minimum absolute spatial expression difference
<code>min_log2_fc</code>	minimum log2 fold-change
<code>min_zscore</code>	minimum z-score change
<code>zscores_column</code>	calculate z-scores over cell types or genes

direction differential expression directions to keep
 cell_color_code vector of colors with cell types as names
 show_plot show plots
 return_plot return plotting object
 save_plot directly save the plot [boolean]
 save_param list of saving parameters from [all_plots_save_function](#)
 default_save_name default save name for saving, don't change, change save_name in save_param

Value

plot

Examples

```
plotCPG(CPGscores)
```

plotHeatmap

plotHeatmap

Description

Creates heatmap for genes and 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("correlation", "custom"),
  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 = NA,
  return_plot = NA,
  save_plot = NA,

```

```

    save_param = list(),
    default_save_name = "plotHeatmap"
)

```

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
cluster_color_code	color code for clusters
cluster_cor_method	method for cluster correlation
cluster_hclust_method	method for hierarchical clustering of clusters
gene_order	method to determine gene order
gene_custom_order	custom order for genes
gene_cor_method	method for gene correlation
gene_hclust_method	method for hierarchical clustering of genes
show_values	which values to show on heatmap
size_vertical_lines	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
show_plot	show plot
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name

Details

If you want to display many genes there are 2 ways to proceed:

- 1. set axis_text_y_size to a really small value and show all genes
- 2. provide a subset of genes to display to gene_label_selection

Value

ggplot

Examples

```
plotHeatmap(gobject)
```

plotICG	<i>plotICG</i>
---------	----------------

Description

Create barplot to visualize interaction changed genes

Usage

```
plotICG(  
  gobject,  
  cpqObject,  
  source_type,  
  source_markers,  
  ICG_genes,  
  cell_color_code = NULL,  
  show_plot = NA,  
  return_plot = NA,  
  save_plot = NA,  
  save_param = list(),  
  default_save_name = "plotICG"  
)
```

Arguments

- | | |
|-------------------|--|
| gobject | giotto object |
| cpqObject | cell proximity gene score object |
| source_type | cell type of the source cell |
| source_markers | markers for the source cell type |
| ICG_genes | named character vector of ICG genes |
| cell_color_code | cell color code for the interacting cell types |
| show_plot | show plots |
| return_plot | return plotting object |
| save_plot | directly save the plot [boolean] |
| save_param | list of saving parameters from all_plots_save_function |
| default_save_name | default save name for saving, don't change, change save_name in save_param |

Value

plot

Examples

```
plotICG(CPGscores)
```

```
plotInteractionChangedGenes
      plotInteractionChangedGenes
```

Description

Create barplot to visualize interaction changed genes

Usage

```
plotInteractionChangedGenes(
  gobject,
  cpgObject,
  source_type,
  source_markers,
  ICG_genes,
  cell_color_code = NULL,
  show_plot = NA,
  return_plot = NA,
  save_plot = NA,
  save_param = list(),
  default_save_name = "plotInteractionChangedGenes"
)
```

Arguments

gobject	giotto object
cpgObject	cell proximity gene score object
source_type	cell type of the source cell
source_markers	markers for the source cell type
ICG_genes	named character vector of ICG genes
cell_color_code	cell color code for the interacting cell types
show_plot	show plots
return_plot	return plotting object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param

Value

plot

Examples

```
plotInteractionChangedGenes(CPGscores)
```

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	<i>plotly_grid</i>
-------------	--------------------

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)
```

plotMetaDataCellsHeatmap	<i>plotMetaDataCellsHeatmap</i>
--------------------------	---------------------------------

Description

Creates heatmap for numeric cell metadata within aggregated clusters.

Usage

```
plotMetaDataCellsHeatmap(
  gobject,
  metadata_cols = NULL,
  spat_enr_names = NULL,
  value_cols = 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_values_order = NULL,
values_cor_method = "pearson",
values_cluster_method = "complete",
midpoint = 0,
x_text_size = 8,
x_text_angle = 45,
y_text_size = 8,
strip_text_size = 8,
show_plot = NA,
return_plot = NA,
save_plot = NA,
save_param = list(),
default_save_name = "plotMetaDataCellsHeatmap"
)

```

Arguments

gobject	giotto object
metadata_cols	annotation columns found in pDataDT(gobject)
spat_enr_names	spatial enrichment results to include
value_cols	value columns to use
first_meta_col	if more than 1 metadata column, select the x-axis factor
second_meta_col	if more than 1 metadata column, select the facetting factor
show_values	which values to show on heatmap
custom_cluster_order	custom cluster order (default = NULL)
clus_cor_method	correlation method for clusters
clus_cluster_method	hierarchical cluster method for the clusters
midpoint	midpoint of show_values
x_text_size	size of x-axis text
x_text_angle	angle of x-axis text
y_text_size	size of y-axis text
strip_text_size	size of strip text
show_plot	show plot
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param
custom_gene_order	custom gene order (default = NULL)

gene_cor_method
correlation method for genes

gene_cluster_method
hierarchical cluster method for the genes

Details

Creates heatmap for the average values of selected value columns in the different annotation groups.

Value

ggplot or data.table

See Also

[plotMetaDataHeatmap](#) for gene expression instead of numeric cell annotation data.

Examples

```
plotMetaDataCellsHeatmap(gobject)
```

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

Description

Creates heatmap for genes within aggregated 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",
  gradient_color = c("blue", "white", "red"),
  gradient_midpoint = 0,
  gradient_limits = NULL,
  x_text_size = 10,
  x_text_angle = 45,
  y_text_size = 10,
  strip_text_size = 8,
  show_plot = NA,
```

```

    return_plot = NA,
    save_plot = NA,
    save_param = list(),
    default_save_name = "plotMetaDataHeatmap"
  )

```

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>gradient_color</code>	vector with 3 colors for numeric data
<code>gradient_midpoint</code>	midpoint for color gradient
<code>gradient_limits</code>	vector with lower and upper limits
<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>	show plot
<code>return_plot</code>	return ggplot object
<code>save_plot</code>	directly save the plot [boolean]
<code>save_param</code>	list of saving parameters from all_plots_save_function
<code>default_save_name</code>	default save name

Details

Creates heatmap for the average expression of selected genes in the different annotation/cluster groups. Calculation of cluster or gene order is done on the provided expression values, but visualization is by default on the z-scores. Other options are the original values or z-scores rescaled per gene (-1 to 1).

Value

ggplot or data.table

See Also

[plotMetaDataCellsHeatmap](#) for numeric cell annotation instead of gene expression.

Examples

```
plotMetaDataHeatmap(gobject)
```

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

Description

Short wrapper for PCA visualization

Usage

```
plotPCA(gobject, dim_reduction_name = "pca", default_save_name = "PCA", ...)
```

Arguments

gobject	giotto object
dim_reduction_name	dimension reduction name
default_save_name	default save name for saving, don't change, change save_name in save_param
groub_by	create multiple plots based on cell annotation column
group_by_subset	subset the group_by factor column
dim1_to_use	dimension to use on x-axis
dim2_to_use	dimension to use on y-axis
spat_enr_names	names of spatial enrichment results to include
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
cell_color_gradient	vector with 3 colors for numeric data
gradient_midpoint	midpoint for color gradient
gradient_limits	vector with lower and upper limits
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_shape	point with border or not (border or no_border)
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
title	title for plot, defaults to cell_color parameter
legend_text	size of legend text
legend_symbol_size	size of legend symbols
background_color	color of plot background
axis_text	size of axis text
axis_title	size of axis title
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_plot	show plot
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function

Details

Description of parameters, see [dimPlot2D](#). For 3D plots see [plotPCA_3D](#)

Value

ggplot

Examples

```
plotPCA(gobject)
```

plotPCA_2D	<i>plotPCA_2D</i>
------------	-------------------

Description

Short wrapper for PCA visualization

Usage

```
plotPCA_2D(  
  gobject,  
  dim_reduction_name = "pca",  
  default_save_name = "PCA_2D",  
  ...  
)
```

Arguments

gobject	giotto object
dim_reduction_name	dimension reduction name
default_save_name	default save name for saving, don't change, change save_name in save_param
groub_by	create multiple plots based on cell annotation column
group_by_subset	subset the group_by factor column
dim1_to_use	dimension to use on x-axis
dim2_to_use	dimension to use on y-axis
spat_enr_names	names of spatial enrichment results to include
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
cell_color_gradient	vector with 3 colors for numeric data
gradient_midpoint	midpoint for color gradient
gradient_limits	vector with lower and upper limits
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_shape	point with border or not (border or no_border)
point_size	size of point (cell)
point_border_col	color of border around points
point_border_stroke	stroke size of border around points
title	title for plot, defaults to cell_color parameter
show_legend	show legend
legend_text	size of legend text
legend_symbol_size	size of legend symbols
background_color	color of plot background
axis_text	size of axis text

axis_title	size of axis title
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_plot	show plot
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function

Details

Description of parameters, see [dimPlot2D](#). For 3D plots see [plotPCA_3D](#)

Value

ggplot

Examples

```
plotPCA_2D(gobject)
```

plotPCA_3D	<i>plotPCA_3D</i>
------------	-------------------

Description

Visualize cells according to 3D PCA dimension reduction

Usage

```
plotPCA_3D(
  gobject,
  dim_reduction_name = "pca",
  default_save_name = "PCA_3D",
  ...
)
```

Arguments

gobject	giotto object
dim_reduction_name	pca dimension reduction name
default_save_name	default save name for saving, ideally change save_name in save_param
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
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
edge_alpha	column to use for alpha of the edges
point_size	size of point (cell)
show_legend	show legend
show_plot	show plot
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function

Details

Description of parameters.

Value

plotly

Examples

```
plotPCA_3D(gobject)
```

plotRankSpatvsExpr	<i>plotRankSpatvsExpr</i>
--------------------	---------------------------

Description

Plots dotplot to compare ligand-receptor rankings from spatial and expression information

Usage

```
plotRankSpatvsExpr(
  gobject,
  combCC,
  expr_rnk_column = "LR_expr_rnk",
  spat_rnk_column = "LR_spat_rnk",
  midpoint = 10,
  size_range = c(0.01, 1.5),
  xlims = NULL,
  ylims = NULL,
  selected_ranks = c(1, 10, 20),
  show_plot = NA,
  return_plot = NA,
  save_plot = NA,
  save_param = list(),
  default_save_name = "plotRankSpatvsExpr"
)
```

Arguments

gobject	giotto object
combCC	combined communication scores from combCCcom
expr_rnk_column	column with expression rank information to use
spat_rnk_column	column with spatial rank information to use
midpoint	midpoint of colors
size_range	size ranges of dotplot
xlims	x-limits, numerical vector of 2
ylims	y-limits, numerical vector of 2
selected_ranks	numerical vector, will be used to print out the percentage of top spatial ranks are recovered
show_plot	show plots
return_plot	return plotting object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param

Value

ggplot

Examples

```
plotRankSpatvsExpr(CPGscores)
```

plotRecovery	<i>plotRecovery</i>
--------------	---------------------

Description

Plots recovery plot to compare ligand-receptor rankings from spatial and expression information

Usage

```
plotRecovery(
  gobject,
  combCC,
  expr_rnk_column = "exprPI_rnk",
  spat_rnk_column = "spatPI_rnk",
  ground_truth = c("spatial", "expression"),
  show_plot = NA,
  return_plot = NA,
  save_plot = NA,
  save_param = list(),
  default_save_name = "plotRecovery"
)
```

Arguments

gobject	giotto object
combCC	combined communication scores from combCCcom
expr_rnk_column	column with expression rank information to use
spat_rnk_column	column with spatial rank information to use
ground_truth	what to consider as ground truth (default: spatial)
show_plot	show plots
return_plot	return plotting object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param

Value

ggplot

Examples

```
plotRecovery(CPGscores)
```

plotRecovery_sub	<i>plotRecovery_sub</i>
------------------	-------------------------

Description

Plots recovery plot to compare ligand-receptor rankings from spatial and expression information

Usage

```
plotRecovery_sub(combCC, first_col = "LR_expr_rnk", second_col = "LR_spat_rnk")
```

Arguments

combCC	combined communication scores from combCCcom
first_col	first column to use
second_col	second column to use

Examples

```
plotRecovery_sub(CPGscores)
```

plotStatDelaunayNetwork	<i>plotStatDelaunayNetwork</i>
-------------------------	--------------------------------

Description

Plots network statistics for a Delaunay network..

Usage

```
plotStatDelaunayNetwork(  
  gobject,  
  method = c("delaunayn_geometry", "RTriangle", "deldir"),  
  dimensions = "all",  
  maximum_distance = "auto",  
  minimum_k = 0,  
  options = "Pp",  
  Y = TRUE,  
  j = TRUE,  
  S = 0,  
  show_plot = NA,  
  return_plot = NA,  
  save_plot = NA,  
  save_param = list(),  
  default_save_name = "plotStatDelaunayNetwork",  
  ...  
)
```

Arguments

<code>gobject</code>	giotto object
<code>dimensions</code>	which spatial dimensions to use (maximum 2 dimensions)
<code>maximum_distance</code>	distance cutoff for Delaunay neighbors to consider
<code>minimum_k</code>	minimum neighbours if <code>maximum_distance</code> != NULL
<code>options</code>	(geometry) String containing extra control options for the underlying Qhull command; see the Qhull documentation (../doc/qhull/html/qdelaun.html) for the available options. (default = 'Pp', do not report precision problems)
<code>Y</code>	(RTriangle) If TRUE prohibits the insertion of Steiner points on the mesh boundary.
<code>j</code>	(RTriangle) If TRUE jettisons vertices that are not part of the final triangulation from the output.
<code>S</code>	(RTriangle) Specifies the maximum number of added Steiner points.
<code>show_plot</code>	show plots
<code>return_plot</code>	return ggplot object
<code>save_plot</code>	directly save the plot [boolean]
<code>save_param</code>	list of saving parameters from all_plots_save_function
<code>default_save_name</code>	default save name for saving, don't change, change <code>save_name</code> in <code>save_param</code>
<code>...</code>	Other parameters of the triangulate function
<code>name</code>	name for spatial network (default = 'delaunay_network')

Details

Plots statistics for a spatial Delaunay network as explained in [triangulate](#). This can be used to further finetune the [createDelaunayNetwork](#) function.

Value

giotto object with updated spatial network slot

Examples

```
plotStatDelaunayNetwork(gobject)
```

plotTSNE

plotTSNE

Description

Short wrapper for tSNE visualization

Usage

```
plotTSNE(gobject, dim_reduction_name = "tsne", default_save_name = "tSNE", ...)
```

Arguments

<code>gobject</code>	giotto object
<code>dim_reduction_name</code>	dimension reduction name
<code>default_save_name</code>	default save name for saving, don't change, change <code>save_name</code> in <code>save_param</code>
<code>groub_by</code>	create multiple plots based on cell annotation column
<code>group_by_subset</code>	subset the <code>group_by</code> factor column
<code>dim1_to_use</code>	dimension to use on x-axis
<code>dim2_to_use</code>	dimension to use on y-axis
<code>spat_enr_names</code>	names of spatial enrichment results to include
<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</code> = TRUE
<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>cell_color_gradient</code>	vector with 3 colors for numeric data
<code>gradient_midpoint</code>	midpoint for color gradient
<code>gradient_limits</code>	vector with lower and upper limits
<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>other_point_size</code>	size of not selected cells
<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

point_shape	point with border or not (border or no_border)
point_size	size of point (cell)
point_border_col	color of border around points
point_border_stroke	stroke size of border around points
title	title for plot, defaults to cell_color parameter
show_legend	show legend
legend_text	size of legend text
legend_symbol_size	size of legend symbols
background_color	color of plot background
axis_text	size of axis text
axis_title	size of axis title
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_plot	show plot
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function

Details

Description of parameters, see [dimPlot2D](#). For 3D plots see [plotTSNE_3D](#)

Value

ggplot

Examples

```
plotTSNE(gobject)
```

plotTSNE_2D

plotTSNE_2D

Description

Short wrapper for tSNE visualization

Usage

```
plotTSNE_2D(
  gobject,
  dim_reduction_name = "tsne",
  default_save_name = "tSNE_2D",
  ...
)
```

Arguments

<code>gobject</code>	giotto object
<code>dim_reduction_name</code>	dimension reduction name
<code>default_save_name</code>	default save name for saving, don't change, change <code>save_name</code> in <code>save_param</code>
<code>groub_by</code>	create multiple plots based on cell annotation column
<code>group_by_subset</code>	subset the <code>group_by</code> factor column
<code>dim1_to_use</code>	dimension to use on x-axis
<code>dim2_to_use</code>	dimension to use on y-axis
<code>spat_enr_names</code>	names of spatial enrichment results to include
<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</code> = TRUE
<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>cell_color_gradient</code>	vector with 3 colors for numeric data
<code>gradient_midpoint</code>	midpoint for color gradient
<code>gradient_limits</code>	vector with lower and upper limits
<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>other_point_size</code>	size of not selected cells
<code>show_cluster_center</code>	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_shape	point with border or not (border or no_border)
point_size	size of point (cell)
point_border_col	color of border around points
point_border_stroke	stroke size of border around points
title	title for plot, defaults to cell_color parameter
show_legend	show legend
legend_text	size of legend text
legend_symbol_size	size of legend symbols
background_color	color of plot background
axis_text	size of axis text
axis_title	size of axis title
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_plot	show plot
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function

Details

Description of parameters, see [dimPlot2D](#). For 3D plots see [plotTSNE_3D](#)

Value

ggplot

Examples

```
plotTSNE_2D(gobject)
```

plotTSNE_3D

plotTSNE_3D

Description

Visualize cells according to dimension reduction coordinates

Usage

```
plotTSNE_3D(
  gobject,
  dim_reduction_name = "tsne",
  default_save_name = "TSNE_3D",
  ...
)
```

Arguments

gobject	giotto object
dim_reduction_name	tsne dimension reduction name
default_save_name	default save name for saving, don't change, change save_name in save_param
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
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
edge_alpha	column to use for alpha of the edges
point_size	size of point (cell)
show_legend	show legend
show_plot	show plot
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function

Details

Description of parameters.

Value

plotly

Examples

```
plotTSNE_3D(gobject)
```

plotUMAP

plotUMAP

Description

Short wrapper for UMAP visualization

Usage

```
plotUMAP(gobject, dim_reduction_name = "umap", default_save_name = "UMAP", ...)
```

Arguments

gobject	giotto object
dim_reduction_name	dimension reduction name
default_save_name	default save name for saving, don't change, change save_name in save_param
groub_by	create multiple plots based on cell annotation column
group_by_subset	subset the group_by factor column
dim1_to_use	dimension to use on x-axis

dim2_to_use	dimension to use on y-axis
spat_enr_names	names of spatial enrichment results to include
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
cell_color_gradient	vector with 3 colors for numeric data
gradient_midpoint	midpoint for color gradient
gradient_limits	vector with lower and upper limits
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_shape	point with border or not (border or no_border)
point_size	size of point (cell)
point_border_col	color of border around points
point_border_stroke	stroke size of border around points
title	title for plot, defaults to cell_color parameter
show_legend	show legend
legend_text	size of legend text
legend_symbol_size	size of legend symbols

background_color	color of plot background
axis_text	size of axis text
axis_title	size of axis title
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_plot	show plot
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function

Details

Description of parameters, see [dimPlot2D](#). For 3D plots see [plotUMAP_3D](#)

Value

ggplot

Examples

```
plotUMAP(gobject)
```

plotUMAP_2D

plotUMAP_2D

Description

Short wrapper for UMAP visualization

Usage

```
plotUMAP_2D(
  gobject,
  dim_reduction_name = "umap",
  default_save_name = "UMAP_2D",
  ...
)
```

Arguments

<code>gobject</code>	giotto object
<code>dim_reduction_name</code>	dimension reduction name
<code>default_save_name</code>	default save name for saving, don't change, change <code>save_name</code> in <code>save_param</code>
<code>groub_by</code>	create multiple plots based on cell annotation column
<code>group_by_subset</code>	subset the <code>group_by</code> factor column
<code>dim1_to_use</code>	dimension to use on x-axis
<code>dim2_to_use</code>	dimension to use on y-axis
<code>spat_enr_names</code>	names of spatial enrichment results to include
<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</code> = TRUE
<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>cell_color_gradient</code>	vector with 3 colors for numeric data
<code>gradient_midpoint</code>	midpoint for color gradient
<code>gradient_limits</code>	vector with lower and upper limits
<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>other_point_size</code>	size of not selected cells
<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

point_shape	point with border or not (border or no_border)
point_size	size of point (cell)
point_border_col	color of border around points
point_border_stroke	stroke size of border around points
title	title for plot, defaults to cell_color parameter
show_legend	show legend
legend_text	size of legend text
legend_symbol_size	size of legend symbols
background_color	color of plot background
axis_text	size of axis text
axis_title	size of axis title
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_plot	show plot
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function

Details

Description of parameters, see [dimPlot2D](#). For 3D plots see [plotUMAP_3D](#)

Value

ggplot

Examples

```
plotUMAP_2D(gobject)
```

plotUMAP_3D

plotUMAP_3D

Description

Visualize cells according to dimension reduction coordinates

Usage

```
plotUMAP_3D(
  gobject,
  dim_reduction_name = "umap",
  default_save_name = "UMAP_3D",
  ...
)
```

Arguments

<code>gobject</code>	giotto object
<code>dim_reduction_name</code>	umap dimension reduction name
<code>default_save_name</code>	default save name for saving, don't change, change <code>save_name</code> in <code>save_param</code>
<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>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>other_point_size</code>	size of not selected cells
<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

show_plot	show plot
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function

Details

Description of parameters.

Value

plotly

Examples

```
plotUMAP_3D(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(
  gobject,
  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,
  cell_color_gradient = c("blue", "white", "red"),
  gradient_midpoint = 0,
  gradient_limits = NULL,
  select_cell_groups = NULL,
  select_cells = NULL,
  point_size = 1,
  point_border_col = "black",
  point_border_stroke = 0.1,
  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,
  show_other_cells = T,
  other_cell_color = "lightgrey",
  other_point_size = 0.5,
  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
cell_color_gradient	vector with 3 colors for numeric data
gradient_midpoint	midpoint for color gradient
gradient_limits	vector with lower and upper limits
select_cell_groups	select subset of cells/clusters based on cell_color parameter
select_cells	select subset of cells based on cell IDs
point_size	size of point (cell)
point_border_col	color of border around points
point_border_stroke	stroke size of border around points
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
show_other_cells	display not selected cells
other_cell_color	color of not selected cells
other_point_size	size of not selected cells
show_legend	show legend
gobject	giotto object

Details

Description of parameters.

Value

ggplot

Examples

```
plot_point_layer_ggplot(gobject)
```

```
plot_point_layer_ggplot_noFILL
      plot_point_layer_ggplot_noFILL
```

Description

Visualize cells in point layer according to dimension reduction coordinates without borders

Usage

```
plot_point_layer_ggplot_noFILL(
  ggobject,
  annotated_DT_selected,
  annotated_DT_other,
  cell_color = NULL,
  color_as_factor = T,
  cell_color_code = NULL,
  cell_color_gradient = c("blue", "white", "red"),
  gradient_midpoint = 0,
  gradient_limits = NULL,
  select_cell_groups = NULL,
  select_cells = NULL,
  point_size = 1,
  show_cluster_center = F,
  show_center_label = T,
  center_point_size = 4,
  label_size = 4,
  label_fontface = "bold",
  edge_alpha = NULL,
  show_other_cells = T,
  other_cell_color = "lightgrey",
  other_point_size = 0.5,
  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
cell_color_gradient	vector with 3 colors for numeric data
gradient_midpoint	midpoint for color gradient

gradient_limits	vector with lower and upper limits
select_cell_groups	select subset of cells/clusters based on cell_color parameter
select_cells	select subset of cells based on cell IDs
point_size	size of point (cell)
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
show_other_cells	display not selected cells
other_cell_color	color of not selected cells
other_point_size	size of not selected cells
show_legend	show legend
gobject	giotto object

Details

Description of parameters.

Value

ggplot

Examples

```
plot_point_layer_ggplot_noFILL(gobject)
```

```
plot_spat_point_layer_ggplot
      plot_spat_point_layer_ggplot
```

Description

creat ggplot point layer for spatial coordinates

Usage

```

plot_spat_point_layer_ggplot(
  ggobject,
  sdimx = NULL,
  sdimy = NULL,
  cell_locations_metadata_selected,
  cell_locations_metadata_other,
  cell_color = NULL,
  color_as_factor = T,
  cell_color_code = NULL,
  cell_color_gradient = c("blue", "white", "red"),
  gradient_midpoint = NULL,
  gradient_limits = NULL,
  select_cell_groups = NULL,
  select_cells = NULL,
  point_size = 2,
  point_border_col = "lightgrey",
  point_border_stroke = 0.1,
  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",
  show_other_cells = T,
  other_cell_color = "lightgrey",
  other_point_size = 1,
  show_legend = TRUE
)

```

Arguments

<code>sdimx</code>	x-axis dimension name (default = 'sdimx')
<code>sdimy</code>	y-axis dimension name (default = 'sdimy')
<code>cell_locations_metadata_selected</code>	annotated location from selected cells
<code>cell_locations_metadata_other</code>	annotated location from non-selected cells
<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>cell_color_gradient</code>	vector with 3 colors for numeric data
<code>gradient_midpoint</code>	midpoint for color gradient
<code>gradient_limits</code>	vector with lower and upper limits

select_cell_groups	select subset of cells/clusters based on cell_color parameter
select_cells	select subset of cells based on cell IDs
point_size	size of point (cell)
point_border_col	color of border around points
point_border_stroke	stroke size of border around points
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
show_other_cells	display not selected cells
other_cell_color	color for not selected cells
other_point_size	point size for not selected cells
show_legend	show legend
gobject	giotto object

Details

Description of parameters.

Value

ggplot

Examples

```
plot_spat_point_layer_ggplot(gobject)
```

```
plot_spat_point_layer_ggplot_noFILL
  plot_spat_point_layer_ggplot_noFILL
```

Description

creat ggplot point layer for spatial coordinates without borders

Usage

```
plot_spat_point_layer_ggplot_noFILL(
  ggobject,
  sdimx = NULL,
  sdimy = NULL,
  cell_locations_metadata_selected,
  cell_locations_metadata_other,
  cell_color = NULL,
  color_as_factor = T,
  cell_color_code = NULL,
  cell_color_gradient = c("blue", "white", "red"),
  gradient_midpoint = NULL,
  gradient_limits = NULL,
  select_cell_groups = NULL,
  select_cells = NULL,
  point_size = 2,
  show_cluster_center = F,
  show_center_label = T,
  center_point_size = 4,
  label_size = 4,
  label_fontface = "bold",
  show_other_cells = T,
  other_cell_color = "lightgrey",
  other_point_size = 1,
  show_legend = TRUE
)
```

Arguments

sdimx	x-axis dimension name (default = 'sdimx')
sdimy	y-axis dimension name (default = 'sdimy')
cell_locations_metadata_selected	annotated location from selected cells
cell_locations_metadata_other	annotated location from non-selected cells
cell_color	color for cells (see details)
color_as_factor	convert color column to factor
cell_color_code	named vector with colors
cell_color_gradient	vector with 3 colors for numeric data
gradient_midpoint	midpoint for color gradient
gradient_limits	vector with lower and upper limits
select_cell_groups	select subset of cells/clusters based on cell_color parameter
select_cells	select subset of cells based on cell IDs
point_size	size of point (cell)

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
show_other_cells	display not selected cells
other_cell_color	color for not selected cells
other_point_size	point size for not selected cells
show_legend	show legend
gobject	giotto object

Details

Description of parameters.

Value

ggplot

Examples

```
plot_spat_point_layer_ggplot_noFILL(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

projection_fun	<i>projection_fun</i>
----------------	-----------------------

Description

project a point onto a plane

Usage

```
projection_fun(point_to_project, plane_point, plane_norm)
```

rankEnrich	<i>rankEnrich</i>
------------	-------------------

Description

Function to calculate gene signature enrichment scores per spatial position using a rank based approach.

Usage

```
rankEnrich(
  gobject,
  sign_matrix,
  expression_values = c("normalized", "scaled", "custom"),
  reverse_log_scale = TRUE,
  logbase = 2,
  output_enrichment = c("original", "zscore")
)
```

Arguments

gobject	Giotto object
sign_matrix	Matrix of signature genes for each cell type / process
expression_values	expression values to use
reverse_log_scale	reverse expression values from log scale
logbase	log base to use if reverse_log_scale = TRUE
output_enrichment	how to return enrichment output

Details

sign_matrix: a rank-fold matrix with genes as row names and cell-types as column names. Alternatively a scRNA-seq matrix and vector with clusters can be provided to makeSignMatrixRank, which will create the matrix for you.

First a new rank is calculated as $R = (R1 * R2)^{(1/2)}$, where R1 is the rank of fold-change for each gene in each spot and R2 is the rank of each marker in each cell type. The Rank-Biased Precision is then calculated as: $RBP = (1 - 0.99) * (0.99)^{(R - 1)}$ and the final enrichment score is then calculated as the sum of top 100 RBPs.

Value

data.table with enrichment results

See Also

[makeSignMatrixRank](#)

Examples

```
rankEnrich(gobject)
```

rankSpatialCorGroups	<i>rankSpatialCorGroups</i>
----------------------	-----------------------------

Description

Rank spatial correlated clusters according to correlation structure

Usage

```
rankSpatialCorGroups(
  gobject,
  spatCorObject,
  use_clus_name = NULL,
  show_plot = NA,
  return_plot = FALSE,
  save_plot = NA,
  save_param = list(),
  default_save_name = "rankSpatialCorGroups"
)
```

Arguments

gobject	giotto object
spatCorObject	spatial correlation object
use_clus_name	name of clusters to visualize (from clusterSpatialCorGenes())
show_plot	show plot
return_plot	return ggplot object

save_plot directly save the plot [boolean]
save_param list of saving parameters from [all_plots_save_function](#)
default_save_name default save name for saving, don't change, change save_name in save_param

Value

data.table with positive (within group) and negative (outside group) scores

Examples

rankSpatialCorGroups(gobject)

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

Description

create binarized scores from a vector using arbitrary rank

Usage

rank_binarize(x, max_rank = 200)

readGiottoInstructions	<i>readGiottoInstructions</i>
------------------------	-------------------------------

Description

Retrieves the instruction associated with the provided parameter

Usage

readGiottoInstructions(giotto_instructions, param = NULL)

Arguments

giotto_instructions giotto object or result from createGiottoInstructions()
param parameter to retrieve

Value

specific parameter

Examples

readGiottoInstructions()

read_crossSection	<i>read_crossSection</i>
-------------------	--------------------------

Description

read a cross section object from a giotto object

Usage

```
read_crossSection(gobject, name = NULL, spatial_network_name = NULL)
```

removeCellAnnotation	<i>removeCellAnnotation</i>
----------------------	-----------------------------

Description

removes cell annotation of giotto object

Usage

```
removeCellAnnotation(gobject, columns = NULL, return_gobject = TRUE)
```

Arguments

gobject	giotto object
columns	names of columns to remove
return_gobject	boolean: return giotto object (default = TRUE)

Details

if return_gobject = FALSE, it will return the cell metadata

Value

giotto object

Examples

```
removeCellAnnotation(gobject)
```

removeGeneAnnotation	<i>removeGeneAnnotation</i>
----------------------	-----------------------------

Description

removes gene annotation of giotto object

Usage

```
removeGeneAnnotation(gobject, columns = NULL, return_gobject = TRUE)
```

Arguments

gobject	giotto object
columns	names of columns to remove
return_gobject	boolean: return giotto object (default = TRUE)

Details

if return_gobject = FALSE, it will return the gene metadata

Value

giotto object

Examples

```
removeGeneAnnotation(gobject)
```

replaceGiottoInstructions	<i>replaceGiottoInstructions</i>
---------------------------	----------------------------------

Description

Function to replace all instructions from giotto object

Usage

```
replaceGiottoInstructions(gobject, instructions = NULL)
```

Arguments

gobject	giotto object
instructions	new instructions (e.g. result from createGiottoInstructions)

Value

named vector with giotto instructions

Examples

```
replaceGiottoInstructions()
```

```
reshape_to_data_point  reshape_to_data_point
```

Description

reshape a mesh grid line object to data point matrix

Usage

```
reshape_to_data_point(mesh_grid_obj)
```

```
reshape_to_mesh_grid_obj  
      reshape_to_mesh_grid_obj
```

Description

reshape a data point matrix to a mesh grid line object

Usage

```
reshape_to_mesh_grid_obj(data_points, mesh_grid_n)
```

```
runPCA      runPCA
```

Description

runs a Principal Component Analysis

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

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

Details

See [PCA](#) for more information about other parameters.

Value

giotto object with updated PCA dimension reduction

Examples

```
runPCA(gobject)
```

<code>runtSNE</code>	<i>runtSNE</i>
----------------------	----------------

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

See [Rtsne](#) for more information about these and other parameters.

- Input for tSNE dimension reduction can be another dimension reduction (default = 'pca')
- To use gene expression as input set `dim_reduction_to_use = NULL`
- multiple tSNE results can be stored by changing the *name* of the analysis

Value

giotto object with updated tSNE dimension reduction

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

gobject	giotto object
expression_values	expression values to use
reduction	cells or genes
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	arbitrary name for UMAP run
genes_to_use	if dim_reduction_to_use = NULL, which genes to use
return_gobject	boolean: return giotto object (default = TRUE)
n_neighbors	UMAP param: number of neighbors
n_components	UMAP param: number of components
n_epochs	UMAP param: number of epochs

min_dist	UMAP param: minimum distance
n_threads	UMAP param: threads to use
spread	UMAP param: spread
set_seed	use of seed
seed_number	seed number to use
...	additional UMAP parameters

Details

See [umap](#) for more information about these and other parameters.

- Input for UMAP dimension reduction can be another dimension reduction (default = 'pca')
- To use gene expression as input set `dim_reduction_to_use = NULL`
- multiple UMAP results can be stored by changing the *name* of the analysis

Value

giotto object with updated UMAP dimension reduction

Examples

```
runUMAP(gobject)
```

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

Description

Select genes correlated with spatial patterns

Usage

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

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

Data.table with genes associated with selected dimension (PC).

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

```
select_spatialNetwork  select_spatialNetwork
```

Description

function to select a spatial network

Usage

```
select_spatialNetwork(gobject, name = NULL, return_network_Obj = FALSE)
```

```
set_giotto_python_path
    set_giotto_python_path
```

Description

sets the python path and/or install miniconda and the python modules

Usage

```
set_giotto_python_path(
    python_path = NULL,
    packages_to_install = c("pandas", "networkx", "python-igraph", "leidenalg",
        "python-louvain")
)
```

```
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 for selected clusters.

Usage

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

```

    return_plot = NA,
    save_plot = NA,
    save_param = list(),
    default_save_name = "showClusterDendrogram",
    ...
  )

```

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>rotate</code>	rotate dendrogram 90 degrees
<code>show_plot</code>	show plot
<code>return_plot</code>	return ggplot object
<code>save_plot</code>	directly save the plot [boolean]
<code>save_param</code>	list of saving parameters from all_plots_save_function
<code>default_save_name</code>	default save name for saving, don't change, change save_name in save_param
<code>...</code>	additional parameters for <code>ggdendrogram()</code>

Details

Expression correlation dendrogram for selected clusters.

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"),
  genes = "all",
  cluster_column,
  cor = c("pearson", "spearman"),
  distance = "ward.D",
  show_plot = NA,
  return_plot = NA,
  save_plot = NA,
  save_param = list(),
  default_save_name = "showClusterHeatmap",
  ...
)
```

Arguments

<code>gobject</code>	giotto object
<code>expression_values</code>	expression values to use
<code>genes</code>	vector of genes to use, default to 'all'
<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>show_plot</code>	show plot
<code>return_plot</code>	return ggplot object
<code>save_plot</code>	directly save the plot [boolean]
<code>save_param</code>	list of saving parameters from all_plots_save_function
<code>default_save_name</code>	default save name for saving, don't change, change <code>save_name</code> in <code>save_param</code>
<code>...</code>	additional parameters for the Heatmap function from ComplexHeatmap

Details

Correlation heatmap of selected clusters.

Value

ggplot

Examples

```
showClusterHeatmap(gobject)
```

showGiottoInstructions	<i>showGiottoInstructions</i>
------------------------	-------------------------------

Description

Function to display all instructions from giotto object

Usage

```
showGiottoInstructions(gobject)
```

Arguments

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

Value

named vector with giotto instructions

Examples

```
showGiottoInstructions()
```

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

Description

show patterns for 2D spatial data

Usage

```
showPattern(gobject, spatPatObj, ...)
```

Arguments

gobject	giotto object
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 plot
return_plot	return ggplot object

save_plot directly save the plot [boolean]
save_param list of saving parameters from [all_plots_save_function](#)
default_save_name default save name for saving, don't change, change save_name in save_param

Value

ggplot

See Also

[showPattern2D](#)

Examples

showPattern(gobject)

showPattern2D	<i>showPattern2D</i>
---------------	----------------------

Description

show patterns for 2D spatial data

Usage

```
showPattern2D(  
  gobject,  
  spatPatObj,  
  dimension = 1,  
  trim = c(0.02, 0.98),  
  background_color = "white",  
  grid_border_color = "grey",  
  show_legend = T,  
  point_size = 1,  
  show_plot = NA,  
  return_plot = NA,  
  save_plot = NA,  
  save_param = list(),  
  default_save_name = "showPattern2D"  
)
```

Arguments

gobject giotto object
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 plot

return_plot return ggplot object

save_plot directly save the plot [boolean]

save_param list of saving parameters from [all_plots_save_function](#)

default_save_name default save name for saving, don't change, change save_name in save_param

Value

ggplot

Examples

showPattern2D(gobject)

showPattern3D	<i>showPattern3D</i>
---------------	----------------------

Description

show patterns for 3D spatial data

Usage

```
showPattern3D(  
  gobject,  
  spatPatObj,  
  dimension = 1,  
  trim = c(0.02, 0.98),  
  background_color = "white",  
  grid_border_color = "grey",  
  show_legend = T,  
  point_size = 1,  
  axis_scale = c("cube", "real", "custom"),  
  custom_ratio = NULL,  
  x_ticks = NULL,  
  y_ticks = NULL,  
  z_ticks = NULL,  
  show_plot = NA,  
  return_plot = NA,  
  save_plot = NA,  
  save_param = list(),  
  default_save_name = "showPattern3D"  
)
```

Arguments

gobject	giotto object
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 plot
point_size	adjust the point size
axis_scale	scale the axis
custom_ratio	customize the scale of the axis
x_ticks	the tick number of x_axis
y_ticks	the tick number of y_axis
z_ticks	the tick number of z_axis
show_plot	show plot
return_plot	return plot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param

Value

plotly

Examples

```
showPattern3D(gobject)
```

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

Description

show genes correlated with spatial patterns

Usage

```
showPatternGenes(
  gobject,
  spatPatObj,
  dimension = 1,
  top_pos_genes = 5,
  top_neg_genes = 5,
  point_size = 1,
  return_DT = FALSE,
  show_plot = NA,
  return_plot = NA,
  save_plot = NA,
  save_param = list(),
  default_save_name = "showPatternGenes"
)
```

Arguments

<code>gobject</code>	giotto object
<code>spatPatObj</code>	Output from <code>detectSpatialPatterns</code>
<code>dimension</code>	dimension to plot genes for.
<code>top_pos_genes</code>	Top positively correlated genes.
<code>top_neg_genes</code>	Top negatively correlated genes.
<code>point_size</code>	size of points
<code>return_DT</code>	if TRUE, it will return the data.table used to generate the plots
<code>show_plot</code>	show plot
<code>return_plot</code>	return ggplot object
<code>save_plot</code>	directly save the plot [boolean]
<code>save_param</code>	list of saving parameters from <code>all_plots_save_function()</code>
<code>default_save_name</code>	default save name for saving, don't change, change <code>save_name</code> in <code>save_param</code>

Value

ggplot

Examples

```
showPatternGenes(gobject)
```

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

Description

shows the sequential processing steps that were performed in a summarized format

Usage

```
showProcessingSteps(gobject)
```

Arguments

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

Value

list of processing steps and names

Examples

```
showProcessingSteps(gobject)
```

showSpatialCorGenes	<i>showSpatialCorGenes</i>
---------------------	----------------------------

Description

Shows and filters spatially correlated genes

Usage

```
showSpatialCorGenes(  
  spatCorObject,  
  use_clus_name = NULL,  
  selected_clusters = NULL,  
  genes = NULL,  
  min_spat_cor = 0.5,  
  min_expr_cor = NULL,  
  min_cor_diff = NULL,  
  min_rank_diff = NULL,  
  show_top_genes = NULL  
)
```

Arguments

spatCorObject spatial correlation object
 use_clus_name cluster information to show
 selected_clusters subset of clusters to show
 genes subset of genes to show
 min_spat_cor filter on minimum spatial correlation
 min_expr_cor filter on minimum single-cell expression correlation
 min_cor_diff filter on minimum correlation difference (spatial vs expression)
 min_rank_diff filter on minimum correlation rank difference (spatial vs expression)
 show_top_genes show top genes per gene

Value

data.table with filtered information

Examples

```
showSpatialCorGenes(gobject)
```

 signPCA

signPCA

Description

identify significant principal 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 = NA,
  return_plot = NA,
  save_plot = NA,
  save_param = list(),
  default_save_name = "signPCA",
  ...
)
```

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 plot
<code>return_plot</code>	return ggplot object
<code>save_plot</code>	directly save the plot [boolean]
<code>save_param</code>	list of saving parameters from <code>all_plots_save_function()</code>
<code>default_save_name</code>	default save name for saving, don't change, change <code>save_name</code> in <code>save_param</code>
<code>...</code>	additional parameters for PCA

Details

Two different methods can be used to assess the number of relevant or significant principal components (PC's).

1. Screeplot works by plotting the explained variance of each individual PC in a barplot allowing you to identify which PC does not show a significant contribution anymore (= 'elbow method').
2. The Jackstraw method uses the [permutationPA](#) function. By systematically permuting genes it identifies robust, and thus significant, PCs.

multiple PCA results can be stored by changing the *name* parameter

Value

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

Examples

```
signPCA(gobject)
```

silhouetteRank	<i>silhouetteRank</i>
----------------	-----------------------

Description

Previously: `calculate_spatial_genes_python`. This method computes a silhouette score per gene based on the spatial distribution of two partitions of cells (expressed L1, and non-expressed L0). Here, rather than L2 Euclidean norm, it uses a rank-transformed, exponentially weighted function to represent the local physical distance between two cells.

Usage

```
silhouetteRank(
  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

Value

data.table with spatial scores

Examples

```
silhouetteRank(gobject)
```

```
sort_combine_two_DT_columns
      sort_combine_two_DT_columns
```

Description

fast sorting and pasting of 2 character columns

Usage

```
sort_combine_two_DT_columns(DT, column1, column2, myname = "unif_gene_gene")
```

Examples

```
sort_combine_two_DT_columns()
```

```
spatCellCellcom      spatCellCellcom
```

Description

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

Usage

```
spatCellCellcom(
  gobject,
  spatial_network_name = "Delaunay_network",
  cluster_column = "cell_types",
  random_iter = 1000,
  gene_set_1,
  gene_set_2,
  log2FC_addendum = 0.1,
  min_observations = 2,
  adjust_method = c("fdr", "bonferroni", "BH", "holm", "hochberg", "hommel", "BY",
    "none"),
  adjust_target = c("genes", "cells"),
  do_parallel = TRUE,
  cores = NA,
  verbose = c("a little", "a lot", "none")
)
```

Arguments

```
gobject      giotto object to use
spatial_network_name
              spatial network to use for identifying interacting cells
cluster_column cluster column with cell type information
random_iter   number of iterations
```

gene_set_1	first specific gene set from gene pairs
gene_set_2	second specific gene set from gene pairs
log2FC_addendum	addendum to add when calculating log2FC
min_observations	minimum number of interactions needed to be considered
adjust_method	which method to adjust p-values
adjust_target	adjust multiple hypotheses at the cell or gene level
do_parallel	run calculations in parallel with mclapply
cores	number of cores to use if do_parallel = TRUE
verbose	verbose

Details

Statistical framework to identify if pairs of genes (such as ligand-receptor combinations) are expressed at higher levels than expected based on a reshuffled null distribution of gene expression values in cells that are spatially in proximity to eachother.. More details will follow soon.

Value

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

Examples

```
spatCellCellcom(gobject)
```

spatCellPlot	<i>spatCellPlot</i>
--------------	---------------------

Description

Visualize cells according to spatial coordinates

Usage

```
spatCellPlot(  
  gobject,  
  sdimx = "sdimx",  
  sdimy = "sdimy",  
  spat_enr_names = NULL,  
  cell_annotation_values = NULL,  
  cell_color_gradient = c("blue", "white", "red"),  
  gradient_midpoint = NULL,  
  gradient_limits = NULL,  
  select_cell_groups = NULL,  
  select_cells = NULL,  
  point_shape = c("border", "no_border"),  
  point_size = 3,  
  point_border_col = "black",
```

```

    point_border_stroke = 0.1,
    show_cluster_center = F,
    show_center_label = F,
    center_point_size = 4,
    center_point_border_col = "black",
    center_point_border_stroke = 0.1,
    label_size = 4,
    label_fontface = "bold",
    show_network = F,
    spatial_network_name = "Delaunay_network",
    network_color = NULL,
    network_alpha = 1,
    show_grid = F,
    spatial_grid_name = "spatial_grid",
    grid_color = NULL,
    show_other_cells = T,
    other_cell_color = "lightgrey",
    other_point_size = 1,
    other_cells_alpha = 0.1,
    coord_fix_ratio = NULL,
    show_legend = T,
    legend_text = 8,
    legend_symbol_size = 1,
    background_color = "white",
    axis_text = 8,
    axis_title = 8,
    cow_n_col = 2,
    cow_rel_h = 1,
    cow_rel_w = 1,
    cow_align = "h",
    show_plot = NA,
    return_plot = NA,
    save_plot = NA,
    save_param = list(),
    default_save_name = "spatCellPlot"
  )

```

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>spat_enr_names</code>	names of spatial enrichment results to include
<code>cell_annotation_values</code>	numeric cell annotation columns
<code>cell_color_gradient</code>	vector with 3 colors for numeric data
<code>gradient_midpoint</code>	midpoint for color gradient
<code>gradient_limits</code>	vector with lower and upper limits

select_cell_groups	select subset of cells/clusters based on cell_color parameter
select_cells	select subset of cells based on cell IDs
point_shape	point with border or not (border or no_border)
point_size	size of point (cell)
point_border_col	color of border around points
point_border_stroke	stroke size of border around points
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
show_network	show underlying spatial network
spatial_network_name	name of spatial network to use
network_color	color of spatial network
network_alpha	alpha of spatial network
show_grid	show spatial grid
spatial_grid_name	name of spatial grid to use
grid_color	color of spatial grid
show_other_cells	display not selected cells
other_cell_color	color of not selected cells
other_point_size	point size of not selected cells
other_cells_alpha	alpha of not selected cells
coord_fix_ratio	fix ratio between x and y-axis
show_legend	show legend
legend_text	size of legend text
legend_symbol_size	size of legend symbols
background_color	color of plot background
axis_text	size of axis text
axis_title	size of axis title
show_plot	show plot

return_plot return ggplot object
save_plot directly save the plot [boolean]
save_param list of saving parameters from [all_plots_save_function](#)
default_save_name default save name for saving, don't change, change save_name in save_param

Details

Description of parameters.

Value

ggplot

Examples

spatCellPlot(gobject)

spatCellPlot2D	<i>spatCellPlot2D</i>
----------------	-----------------------

Description

Visualize cells according to spatial coordinates

Usage

```
spatCellPlot2D(  
  gobject,  
  sdimx = "sdimx",  
  sdimy = "sdimy",  
  spat_enr_names = NULL,  
  cell_annotation_values = NULL,  
  cell_color_gradient = c("blue", "white", "red"),  
  gradient_midpoint = NULL,  
  gradient_limits = NULL,  
  select_cell_groups = NULL,  
  select_cells = NULL,  
  point_shape = c("border", "no_border"),  
  point_size = 3,  
  point_border_col = "black",  
  point_border_stroke = 0.1,  
  show_cluster_center = F,  
  show_center_label = F,  
  center_point_size = 4,  
  center_point_border_col = "black",  
  center_point_border_stroke = 0.1,  
  label_size = 4,  
  label_fontface = "bold",  
  show_network = F,  
  spatial_network_name = "Delaunay_network",
```

```

network_color = NULL,
network_alpha = 1,
show_grid = F,
spatial_grid_name = "spatial_grid",
grid_color = NULL,
show_other_cells = T,
other_cell_color = "lightgrey",
other_point_size = 1,
other_cells_alpha = 0.1,
coord_fix_ratio = NULL,
show_legend = T,
legend_text = 8,
legend_symbol_size = 1,
background_color = "white",
axis_text = 8,
axis_title = 8,
cow_n_col = 2,
cow_rel_h = 1,
cow_rel_w = 1,
cow_align = "h",
show_plot = NA,
return_plot = NA,
save_plot = NA,
save_param = list(),
default_save_name = "spatCellPlot2D"
)

```

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>spat_enr_names</code>	names of spatial enrichment results to include
<code>cell_annotation_values</code>	numeric cell annotation columns
<code>cell_color_gradient</code>	vector with 3 colors for numeric data
<code>gradient_midpoint</code>	midpoint for color gradient
<code>gradient_limits</code>	vector with lower and upper limits
<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>point_shape</code>	point with border or not (border or no_border)
<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

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
show_network	show underlying spatial network
spatial_network_name	name of spatial network to use
network_color	color of spatial network
network_alpha	alpha of spatial network
show_grid	show spatial grid
spatial_grid_name	name of spatial grid to use
grid_color	color of spatial grid
show_other_cells	display not selected cells
other_cell_color	color of not selected cells
other_point_size	point size of not selected cells
other_cells_alpha	alpha of not selected cells
coord_fix_ratio	fix ratio between x and y-axis
show_legend	show legend
legend_text	size of legend text
legend_symbol_size	size of legend symbols
background_color	color of plot background
axis_text	size of axis text
axis_title	size of axis title
show_plot	show plot
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param

Details

Description of parameters.

Value

ggplot

Examples

```
spatCellPlot2D(gobject)
```

spatDimCellPlot

spatDimCellPlot

Description

Visualize numerical features of cells according to spatial AND dimension reduction coordinates in 2D

Usage

```
spatDimCellPlot(
  gobject,
  plot_alignment = c("vertical", "horizontal"),
  spat_enr_names = NULL,
  cell_annotation_values = NULL,
  dim_reduction_to_use = "umap",
  dim_reduction_name = "umap",
  dim1_to_use = 1,
  dim2_to_use = 2,
  sdimx = "sdimx",
  sdimy = "sdimy",
  cell_color_gradient = c("blue", "white", "red"),
  gradient_midpoint = NULL,
  gradient_limits = NULL,
  select_cell_groups = NULL,
  select_cells = NULL,
  dim_point_shape = c("border", "no_border"),
  dim_point_size = 1,
  dim_point_border_col = "black",
  dim_point_border_stroke = 0.1,
  spat_point_shape = c("border", "no_border"),
  spat_point_size = 1,
  spat_point_border_col = "black",
  spat_point_border_stroke = 0.1,
  dim_show_cluster_center = F,
  dim_show_center_label = T,
  dim_center_point_size = 4,
  dim_center_point_border_col = "black",
  dim_center_point_border_stroke = 0.1,
  dim_label_size = 4,
  dim_label_fontface = "bold",
  spat_show_cluster_center = F,
  spat_show_center_label = F,
  spat_center_point_size = 4,
```



```

    spat_center_point_border_col = "black",
    spat_center_point_border_stroke = 0.1,
    spat_label_size = 4,
    spat_label_fontface = "bold",
    show_NN_network = F,
    nn_network_to_use = "sNN",
    nn_network_name = "sNN.pca",
    dim_edge_alpha = 0.5,
    spat_show_network = F,
    spatial_network_name = "Delaunay_network",
    spat_network_color = "red",
    spat_network_alpha = 0.5,
    spat_show_grid = F,
    spatial_grid_name = "spatial_grid",
    spat_grid_color = "green",
    show_other_cells = TRUE,
    other_cell_color = "grey",
    dim_other_point_size = 0.5,
    spat_other_point_size = 0.5,
    spat_other_cells_alpha = 0.5,
    coord_fix_ratio = NULL,
    cow_n_col = 2,
    cow_rel_h = 1,
    cow_rel_w = 1,
    cow_align = "h",
    show_legend = T,
    legend_text = 8,
    legend_symbol_size = 1,
    dim_background_color = "white",
    spat_background_color = "white",
    axis_text = 8,
    axis_title = 8,
    show_plot = NA,
    return_plot = NA,
    save_plot = NA,
    save_param = list(),
    default_save_name = "spatDimCellPlot"
)

```

Arguments

<code>gobject</code>	giotto object
<code>plot_alignment</code>	direction to align plot
<code>spat_enr_names</code>	names of spatial enrichment results to include
<code>cell_annotation_values</code>	numeric cell annotation columns
<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

dim2_to_use	dimension to use on y-axis
sdimx	= spatial dimension to use on x-axis
sdimy	= spatial dimension to use on y-axis
cell_color_gradient	vector with 3 colors for numeric data
gradient_midpoint	midpoint for color gradient
gradient_limits	vector with lower and upper limits
select_cell_groups	select subset of cells/clusters based on cell_color parameter
select_cells	select subset of cells based on cell IDs
dim_point_shape	spatial points with border or not (border or no_border)
dim_point_size	size of points in dim. reduction space
dim_point_border_col	border color of points in dim. reduction space
dim_point_border_stroke	border stroke of points in dim. reduction space
spat_point_shape	spatial points with border or not (border or no_border)
spat_point_size	size of spatial points
spat_point_border_col	border color of spatial points
spat_point_border_stroke	border stroke of spatial points
dim_show_cluster_center	show the center of each cluster
dim_show_center_label	provide a label for each cluster
dim_center_point_size	size of the center point
dim_center_point_border_col	border color of center point
dim_center_point_border_stroke	stroke size of center point
dim_label_size	size of the center label
dim_label_fontface	font of the center label
spat_show_cluster_center	show the center of each cluster
spat_show_center_label	provide a label for each cluster
spat_center_point_size	size of the center point
spat_label_size	size of the center label

spat_label_fontface	font of the center label
show_NN_network	show underlying NN network
nn_network_to_use	type of NN network to use (kNN vs sNN)
nn_network_name	name of NN network to use, if show_NN_network = TRUE
dim_edge_alpha	column to use for alpha of the edges
spat_show_network	show spatial network
spatial_network_name	name of spatial network to use
spat_network_color	color of spatial network
spat_show_grid	show spatial grid
spatial_grid_name	name of spatial grid to use
spat_grid_color	color of spatial grid
show_other_cells	display not selected cells
other_cell_color	color of not selected cells
dim_other_point_size	size of not selected dim cells
spat_other_point_size	size of not selected spat cells
spat_other_cells_alpha	alpha of not selected spat cells
coord_fix_ratio	ratio for coordinates
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
legend_text	size of legend text
legend_symbol_size	size of legend symbols
dim_background_color	background color of points in dim. reduction space
spat_background_color	background color of spatial points
axis_text	size of axis text
axis_title	size of axis title

show_plot	show plot
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param

Details

Description of parameters.

Value

ggplot

Examples

spatDimCellPlot(gobject)

spatDimCellPlot2D	<i>spatDimCellPlot2D</i>
-------------------	--------------------------

Description

Visualize numerical features of cells according to spatial AND dimension reduction coordinates in 2D

Usage

```
spatDimCellPlot2D(  
  gobject,  
  plot_alignment = c("vertical", "horizontal"),  
  spat_enr_names = NULL,  
  cell_annotation_values = NULL,  
  dim_reduction_to_use = "umap",  
  dim_reduction_name = "umap",  
  dim1_to_use = 1,  
  dim2_to_use = 2,  
  sdimx = "sdimx",  
  sdimy = "sdimy",  
  cell_color_gradient = c("blue", "white", "red"),  
  gradient_midpoint = NULL,  
  gradient_limits = NULL,  
  select_cell_groups = NULL,  
  select_cells = NULL,  
  dim_point_shape = c("border", "no_border"),  
  dim_point_size = 1,  
  dim_point_border_col = "black",  
  dim_point_border_stroke = 0.1,  
  spat_point_shape = c("border", "no_border"),
```

```

    spat_point_size = 1,
    spat_point_border_col = "black",
    spat_point_border_stroke = 0.1,
    dim_show_cluster_center = F,
    dim_show_center_label = T,
    dim_center_point_size = 4,
    dim_center_point_border_col = "black",
    dim_center_point_border_stroke = 0.1,
    dim_label_size = 4,
    dim_label_fontface = "bold",
    spat_show_cluster_center = F,
    spat_show_center_label = F,
    spat_center_point_size = 4,
    spat_center_point_border_col = "black",
    spat_center_point_border_stroke = 0.1,
    spat_label_size = 4,
    spat_label_fontface = "bold",
    show_NN_network = F,
    nn_network_to_use = "sNN",
    nn_network_name = "sNN.pca",
    dim_edge_alpha = 0.5,
    spat_show_network = F,
    spatial_network_name = "Delaunay_network",
    spat_network_color = "red",
    spat_network_alpha = 0.5,
    spat_show_grid = F,
    spatial_grid_name = "spatial_grid",
    spat_grid_color = "green",
    show_other_cells = TRUE,
    other_cell_color = "grey",
    dim_other_point_size = 0.5,
    spat_other_point_size = 0.5,
    spat_other_cells_alpha = 0.5,
    show_legend = T,
    legend_text = 8,
    legend_symbol_size = 1,
    dim_background_color = "white",
    spat_background_color = "white",
    axis_text = 8,
    axis_title = 8,
    coord_fix_ratio = NULL,
    cow_n_col = 2,
    cow_rel_h = 1,
    cow_rel_w = 1,
    cow_align = "h",
    show_plot = NA,
    return_plot = NA,
    save_plot = NA,
    save_param = list(),
    default_save_name = "spatDimCellPlot2D"
  )

```

Arguments

<code>gobject</code>	giotto object
<code>plot_alignment</code>	direction to align plot
<code>spat_enr_names</code>	names of spatial enrichment results to include
<code>cell_annotation_values</code>	numeric cell annotation columns
<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>sdimx</code>	= spatial dimension to use on x-axis
<code>sdimy</code>	= spatial dimension to use on y-axis
<code>cell_color_gradient</code>	vector with 3 colors for numeric data
<code>gradient_midpoint</code>	midpoint for color gradient
<code>gradient_limits</code>	vector with lower and upper limits
<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>dim_point_shape</code>	dim reduction points with border or not (<code>border</code> or <code>no_border</code>)
<code>dim_point_size</code>	size of points in dim. reduction space
<code>dim_point_border_col</code>	border color of points in dim. reduction space
<code>dim_point_border_stroke</code>	border stroke of points in dim. reduction space
<code>spat_point_shape</code>	spatial points with border or not (<code>border</code> or <code>no_border</code>)
<code>spat_point_size</code>	size of spatial points
<code>spat_point_border_col</code>	border color of spatial points
<code>spat_point_border_stroke</code>	border stroke of spatial points
<code>dim_show_cluster_center</code>	show the center of each cluster
<code>dim_show_center_label</code>	provide a label for each cluster
<code>dim_center_point_size</code>	size of the center point
<code>dim_center_point_border_col</code>	border color of center point

```

dim_center_point_border_stroke
    stroke size of center point
dim_label_size  size of the center label
dim_label_fontface
    font of the center label
spat_show_cluster_center
    show the center of each cluster
spat_show_center_label
    provide a label for each cluster
spat_center_point_size
    size of the center point
spat_label_size
    size of the center label
spat_label_fontface
    font of the center label
show_NN_network
    show underlying NN network
nn_network_to_use
    type of NN network to use (kNN vs sNN)
nn_network_name
    name of NN network to use, if show_NN_network = TRUE
dim_edge_alpha  column to use for alpha of the edges
spat_show_network
    show spatial network
spatial_network_name
    name of spatial network to use
spat_network_color
    color of spatial network
spat_show_grid  show spatial grid
spatial_grid_name
    name of spatial grid to use
spat_grid_color
    color of spatial grid
show_other_cells
    display not selected cells
other_cell_color
    color of not selected cells
dim_other_point_size
    size of not selected dim cells
spat_other_point_size
    size of not selected spat cells
spat_other_cells_alpha
    alpha of not selected spat cells
show_legend     show legend
legend_text     size of legend text
legend_symbol_size
    size of legend symbols

```

dim_background_color	background color of points in dim. reduction space
spat_background_color	background color of spatial points
axis_text	size of axis text
axis_title	size of axis title
coord_fix_ratio	ratio for coordinates
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_plot	show plot
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param

Details

Description of parameters.

Value

ggplot

Examples

spatDimCellPlot2D(gobject)

spatDimGenePlot	<i>spatDimGenePlot</i>
-----------------	------------------------

Description

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

Usage

```
spatDimGenePlot(  
  gobject,  
  expression_values = c("normalized", "scaled", "custom"),  
  plot_alignment = c("vertical", "horizontal"),  
  genes,  
  dim_reduction_to_use = "umap",  
  dim_reduction_name = "umap",  
  dim1_to_use = 1,  
)
```



```

dim2_to_use = 2,
dim_point_shape = c("border", "no_border"),
dim_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 = FALSE,
spatial_network_name = "Delaunay_network",
spatial_grid_name = "spatial_grid",
spat_point_shape = c("border", "no_border"),
spat_point_size = 1,
spat_point_border_col = "black",
spat_point_border_stroke = 0.1,
midpoint = 0,
genes_high_color = "red",
genes_mid_color = "white",
genes_low_color = "blue",
show_legend = T,
legend_text = 8,
dim_background_color = "white",
spat_background_color = "white",
axis_text = 8,
axis_title = 8,
cow_n_col = 2,
cow_rel_h = 1,
cow_rel_w = 1,
cow_align = "h",
show_plot = NA,
return_plot = NA,
save_plot = NA,
save_param = list(),
default_save_name = "spatDimGenePlot"
)

```

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

dim2_to_use	dimension to use on y-axis
dim_point_shape	dimension points with border or not (border or no_border)
dim_point_size	dim reduction plot: point size
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
spatial_network_name	name of spatial network to use
spatial_grid_name	name of spatial grid to use
spat_point_shape	spatial points with border or not (border or no_border)
spat_point_size	spatial plot: point size
spat_point_border_col	color of border around points
spat_point_border_stroke	stroke size of border around points
midpoint	size of point (cell)
show_legend	show legend
legend_text	size of legend text
dim_background_color	color of plot background for dimension plot
spat_background_color	color of plot background for spatial plot
axis_text	size of axis text
axis_title	size of axis title
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_plot	show plots
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param

Details

Description of parameters.

Value

ggplot

See Also

[spatDimGenePlot3D](#)

Examples

```
spatDimGenePlot(gobject)
```

spatDimGenePlot2D	<i>spatDimGenePlot2D</i>
-------------------	--------------------------

Description

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

Usage

```
spatDimGenePlot2D(
  gobject,
  expression_values = c("normalized", "scaled", "custom"),
  plot_alignment = c("vertical", "horizontal"),
  genes,
  dim_reduction_to_use = "umap",
  dim_reduction_name = "umap",
  dim1_to_use = 1,
  dim2_to_use = 2,
  dim_point_shape = c("border", "no_border"),
  dim_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 = FALSE,
  spatial_network_name = "Delaunay_network",
  spatial_grid_name = "spatial_grid",
  spat_point_shape = c("border", "no_border"),
  spat_point_size = 1,
  spat_point_border_col = "black",
  spat_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",
show_legend = T,
legend_text = 8,
dim_background_color = "white",
spat_background_color = "white",
axis_text = 8,
axis_title = 8,
show_plot = NA,
return_plot = NA,
save_plot = NA,
save_param = list(),
default_save_name = "spatDimGenePlot2D"
)

```

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>dim_point_shape</code>	dim reduction points with border or not (border or no_border)
<code>dim_point_size</code>	dim reduction plot: point size
<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

spatial_network_name	name of spatial network to use
spatial_grid_name	name of spatial grid to use
spat_point_shape	spatial points with border or not (border or no_border)
spat_point_size	spatial plot: point size
spat_point_border_col	color of border around points
spat_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
legend_text	size of legend text
dim_background_color	color of plot background for dimension plot
spat_background_color	color of plot background for spatial plot
axis_text	size of axis text
axis_title	size of axis title
show_plot	show plots
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param

Details

Description of parameters.

Value

ggplot

See Also

[spatDimGenePlot3D](#)

Examples

```
spatDimGenePlot2D(gobject)
```

spatDimGenePlot3D	<i>spatDimGenePlot3D</i>
-------------------	--------------------------

Description

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

Usage

```
spatDimGenePlot3D(
  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 = "sdimx",
  sdimy = "sdimy",
  sdimz = "sdimz",
  genes,
  cluster_column = NULL,
  select_cell_groups = NULL,
  select_cells = NULL,
  show_other_cells = T,
  other_cell_color = "lightgrey",
  other_point_size = 1.5,
  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 = "Delaunay_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,
    show_plot = NA,
    return_plot = NA,
    save_plot = NA,
    save_param = list(),
    default_save_name = "spatDimGenePlot3D"
)

```

Arguments

<code>gobject</code>	giotto object
<code>expression_values</code>	gene expression values to use
<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>genes</code>	genes to show
<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>dim_point_size</code>	dim reduction plot: point size
<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>show_plot</code>	show plots
<code>return_plot</code>	return plotly object
<code>save_plot</code>	directly save the plot [boolean]
<code>save_param</code>	list of saving parameters from all_plots_save_function
<code>default_save_name</code>	default save name for saving, don't change, change <code>save_name</code> in <code>save_param</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>point_size</code>	size of point (cell)
<code>show_legend</code>	show legend

Details

Description of parameters.

Value

plotly

Examples

spatDimGenePlot3D(gobject)

spatDimPlot	<i>spatDimPlot</i>
-------------	--------------------

Description

Visualize cells according to spatial AND dimension reduction coordinates 2D

Usage

```
spatDimPlot(  
  gobject,  
  plot_alignment = c("vertical", "horizontal"),  
  dim_reduction_to_use = "umap",  
  dim_reduction_name = "umap",  
  dim1_to_use = 1,  
  dim2_to_use = 2,  
  sdimx = "sdimx",  
  sdimy = "sdimy",  
  spat_enr_names = NULL,  
  cell_color = NULL,  
  color_as_factor = T,  
  cell_color_code = NULL,  
  cell_color_gradient = c("blue", "white", "red"),  
  gradient_midpoint = NULL,  
  gradient_limits = NULL,  
  select_cell_groups = NULL,  
  select_cells = NULL,  
  dim_point_shape = c("border", "no_border"),  
  dim_point_size = 1,  
  dim_point_border_col = "black",  
  dim_point_border_stroke = 0.1,  
  spat_point_shape = c("border", "no_border"),  
  spat_point_size = 1,  
  spat_point_border_col = "black",  
  spat_point_border_stroke = 0.1,  
  dim_show_cluster_center = F,  
  dim_show_center_label = T,  
  dim_center_point_size = 4,  
  dim_center_point_border_col = "black",  
  dim_center_point_border_stroke = 0.1,
```



```

dim_label_size = 4,
dim_label_fontface = "bold",
spat_show_cluster_center = F,
spat_show_center_label = F,
spat_center_point_size = 4,
spat_label_size = 4,
spat_label_fontface = "bold",
show_NN_network = F,
nn_network_to_use = "sNN",
network_name = "sNN.pca",
nn_network_alpha = 0.05,
show_spatial_network = F,
spat_network_name = "spatial_network",
spat_network_color = "blue",
spat_network_alpha = 0.5,
show_spatial_grid = F,
spat_grid_name = "spatial_grid",
spat_grid_color = "blue",
show_other_cells = T,
other_cell_color = "lightgrey",
dim_other_point_size = 1,
spat_other_point_size = 1,
spat_other_cells_alpha = 0.5,
dim_show_legend = F,
spat_show_legend = F,
legend_text = 8,
legend_symbol_size = 1,
dim_background_color = "white",
spat_background_color = "white",
axis_text = 8,
axis_title = 8,
show_plot = NA,
return_plot = NA,
save_plot = NA,
save_param = list(),
default_save_name = "spatDimPlot"
)

```

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>sdimx</code>	= spatial dimension to use on x-axis
<code>sdimy</code>	= spatial dimension to use on y-axis
<code>spat_enr_names</code>	names of spatial enrichment results to include

cell_color	color for cells (see details)
color_as_factor	convert color column to factor
cell_color_code	named vector with colors
cell_color_gradient	vector with 3 colors for numeric data
gradient_midpoint	midpoint for color gradient
gradient_limits	vector with lower and upper limits
select_cell_groups	select subset of cells/clusters based on cell_color parameter
select_cells	select subset of cells based on cell IDs
dim_point_shape	point with border or not (border or no_border)
dim_point_size	size of points in dim. reduction space
dim_point_border_col	border color of points in dim. reduction space
dim_point_border_stroke	border stroke of points in dim. reduction space
spat_point_shape	point with border or not (border or no_border)
spat_point_size	size of spatial points
spat_point_border_col	border color of spatial points
spat_point_border_stroke	border stroke of spatial points
dim_show_cluster_center	show the center of each cluster
dim_show_center_label	provide a label for each cluster
dim_center_point_size	size of the center point
dim_center_point_border_col	border color of center point
dim_center_point_border_stroke	stroke size of center point
dim_label_size	size of the center label
dim_label_fontface	font of the center label
spat_show_cluster_center	show the center of each cluster
spat_show_center_label	provide a label for each cluster
spat_center_point_size	size of the center point

```

spat_label_size      size of the center label
spat_label_fontface  font of the center label
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
nn_network_alpha     column to use for alpha of the edges
show_spatial_network show spatial network
spat_network_name    name of spatial network to use
spat_network_color   color of spatial network
show_spatial_grid   show spatial grid
spat_grid_name       name of spatial grid to use
spat_grid_color      color of spatial grid
show_other_cells     display not selected cells
other_cell_color     color of not selected cells
dim_other_point_size size of not selected dim cells
spat_other_point_size size of not selected spat cells
spat_other_cells_alpha alpha of not selected spat cells
dim_show_legend      show legend of dimension reduction plot
spat_show_legend     show legend of spatial plot
legend_text          size of legend text
legend_symbol_size   size of legend symbols
dim_background_color background color of points in dim. reduction space
spat_background_color background color of spatial points
axis_text            size of axis text
axis_title           size of axis title
show_plot            show plot
return_plot          return ggplot object
save_plot            directly save the plot [boolean]
save_param           list of saving parameters from all\_plots\_save\_function
default_save_name    default save name for saving, don't change, change save_name in save_param

```

Details

Description of parameters.

Value

ggplot

See Also

[spatDimPlot2D](#) and [spatDimPlot3D](#) for 3D visualization.

Examples

spatDimPlot(gobject)

spatDimPlot2D	<i>spatDimPlot2D</i>
---------------	----------------------

Description

Visualize cells according to spatial AND dimension reduction coordinates 2D

Usage

```
spatDimPlot2D(  
  gobject,  
  plot_alignment = c("vertical", "horizontal"),  
  dim_reduction_to_use = "umap",  
  dim_reduction_name = "umap",  
  dim1_to_use = 1,  
  dim2_to_use = 2,  
  sdimx = "sdimx",  
  sdimy = "sdimy",  
  spat_enr_names = NULL,  
  cell_color = NULL,  
  color_as_factor = T,  
  cell_color_code = NULL,  
  cell_color_gradient = c("blue", "white", "red"),  
  gradient_midpoint = NULL,  
  gradient_limits = NULL,  
  select_cell_groups = NULL,  
  select_cells = NULL,  
  dim_point_shape = c("border", "no_border"),  
  dim_point_size = 1,  
  dim_point_border_col = "black",  
  dim_point_border_stroke = 0.1,  
  spat_point_shape = c("border", "no_border"),  
  spat_point_size = 1,  
  spat_point_border_col = "black",  
  spat_point_border_stroke = 0.1,  
  dim_show_cluster_center = F,
```

```

dim_show_center_label = T,
dim_center_point_size = 4,
dim_center_point_border_col = "black",
dim_center_point_border_stroke = 0.1,
dim_label_size = 4,
dim_label_fontface = "bold",
spat_show_cluster_center = F,
spat_show_center_label = F,
spat_center_point_size = 4,
spat_label_size = 4,
spat_label_fontface = "bold",
show_NN_network = F,
nn_network_to_use = "sNN",
network_name = "sNN.pca",
nn_network_alpha = 0.05,
show_spatial_network = F,
spat_network_name = "spatial_network",
spat_network_color = "blue",
spat_network_alpha = 0.5,
show_spatial_grid = F,
spat_grid_name = "spatial_grid",
spat_grid_color = "blue",
show_other_cells = T,
other_cell_color = "lightgrey",
dim_other_point_size = 1,
spat_other_point_size = 1,
spat_other_cells_alpha = 0.5,
dim_show_legend = F,
spat_show_legend = F,
legend_text = 8,
legend_symbol_size = 1,
dim_background_color = "white",
spat_background_color = "white",
axis_text = 8,
axis_title = 8,
show_plot = NA,
return_plot = NA,
save_plot = NA,
save_param = list(),
default_save_name = "spatDimPlot2D"
)

```

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>sdimx</code>	= spatial dimension to use on x-axis
<code>sdimy</code>	= spatial dimension to use on y-axis
<code>spat_enr_names</code>	names of spatial enrichment results to include
<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>cell_color_gradient</code>	vector with 3 colors for numeric data
<code>gradient_midpoint</code>	midpoint for color gradient
<code>gradient_limits</code>	vector with lower and upper limits
<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>dim_point_shape</code>	point with border or not (<code>border</code> or <code>no_border</code>)
<code>dim_point_size</code>	size of points in dim. reduction space
<code>dim_point_border_col</code>	border color of points in dim. reduction space
<code>dim_point_border_stroke</code>	border stroke of points in dim. reduction space
<code>spat_point_shape</code>	point with border or not (<code>border</code> or <code>no_border</code>)
<code>spat_point_size</code>	size of spatial points
<code>spat_point_border_col</code>	border color of spatial points
<code>spat_point_border_stroke</code>	border stroke of spatial points
<code>dim_show_cluster_center</code>	show the center of each cluster
<code>dim_show_center_label</code>	provide a label for each cluster
<code>dim_center_point_size</code>	size of the center point
<code>dim_center_point_border_col</code>	border color of center point
<code>dim_center_point_border_stroke</code>	stroke size of center point
<code>dim_label_size</code>	size of the center label
<code>dim_label_fontface</code>	font of the center label
<code>spat_show_cluster_center</code>	show the center of each cluster

spat_show_center_label
 provide a label for each cluster
 spat_center_point_size
 size of the center point
 spat_label_size
 size of the center label
 spat_label_fontface
 font of the center label
 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
 nn_network_alpha
 column to use for alpha of the edges
 show_spatial_network
 show spatial network
 spat_network_name
 name of spatial network to use
 spat_network_color
 color of spatial network
 show_spatial_grid
 show spatial grid
 spat_grid_name name of spatial grid to use
 spat_grid_color
 color of spatial grid
 show_other_cells
 display not selected cells
 other_cell_color
 color of not selected cells
 dim_other_point_size
 size of not selected dim cells
 spat_other_point_size
 size of not selected spat cells
 spat_other_cells_alpha
 alpha of not selected spat cells
 dim_show_legend
 show legend of dimension reduction plot
 spat_show_legend
 show legend of spatial plot
 legend_text size of legend text
 legend_symbol_size
 size of legend symbols
 dim_background_color
 background color of points in dim. reduction space
 spat_background_color
 background color of spatial points
 axis_text size of axis text

axis_title	size of axis title
show_plot	show plot
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param

Details

Description of parameters.

Value

ggplot

See Also

[spatDimPlot3D](#)

Examples

spatDimPlot2D(gobject)

spatDimPlot3D	<i>spatDimPlot3D</i>
---------------	----------------------

Description

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

Usage

```
spatDimPlot3D(  
  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 = 3,  
  sdimx = "sdimx",  
  sdimy = "sdimy",  
  sdimz = "sdimz",  
  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,
    select_cell_groups = NULL,
    select_cells = NULL,
    show_other_cells = T,
    other_cell_color = "lightgrey",
    other_point_size = 1.5,
    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 = "Delaunay_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,
    show_plot = NA,
    return_plot = NA,
    save_plot = NA,
    save_param = list(),
    default_save_name = "spatDimPlot3D"
)

```

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>sdimx</code>	= spatial dimension to use on x-axis
<code>sdimy</code>	= spatial dimension to use on y-axis
<code>sdimz</code>	= spatial 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)

network_name	name of NN network to use, if show_NN_network = TRUE
show_cluster_center	show the center of each cluster
show_center_label	provide a label for each cluster
center_point_size	size of the center point
label_size	size of the center label
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
cell_color	color for cells (see details)
color_as_factor	convert color column to factor
cell_color_code	named vector with colors
dim_point_size	size of points in dim. reduction space
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_point_size	size of spatial points
show_plot	show plot
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param
dim_point_border_col	border color of points in dim. reduction space

dim_point_border_stroke
 border stroke of points in dim. reduction space
 spatial_network_color
 color of spatial network
 spatial_other_point_size
 size of not selected spatial points
 spatial_other_cells_alpha
 alpha of not selected spatial points
 dim_other_point_size
 size of not selected dim. reduction points
 show_legend show legend

Details

Description of parameters.

Value

plotly

Examples

```
spatDimPlot3D(gobject)
```

spatGenePlot	<i>spatGenePlot</i>
--------------	---------------------

Description

Visualize cells and gene expression according to spatial coordinates

Usage

```

spatGenePlot(
  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 = "Delaunay_network",
  edge_alpha = NULL,
  show_grid = F,
  grid_color = NULL,
  spatial_grid_name = "spatial_grid",
  midpoint = 0,
  scale_alpha_with_expression = FALSE,
  point_shape = c("border", "no_border"),
  point_size = 1,

```

```

point_border_col = "black",
point_border_stroke = 0.1,
show_legend = T,
legend_text = 8,
background_color = "white",
axis_text = 8,
axis_title = 8,
cow_n_col = 2,
cow_rel_h = 1,
cow_rel_w = 1,
cow_align = "h",
show_plot = NA,
return_plot = NA,
save_plot = NA,
save_param = list(),
default_save_name = "spatGenePlot"
)

```

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_shape</code>	point with border or not (border or no_border)
<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

legend_text	size of legend text
background_color	color of plot background
axis_text	size of axis text
axis_title	size of axis title
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_plot	show plots
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param
...	parameters for cowplot::save_plot()

Details

Description of parameters.

Value

ggplot

See Also

[spatGenePlot3D](#) and [spatGenePlot2D](#)

Examples

spatGenePlot(gobject)

spatGenePlot2D	<i>spatGenePlot2D</i>
----------------	-----------------------

Description

Visualize cells and gene expression according to spatial coordinates

Usage

```

spatGenePlot2D(
  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 = "Delaunay_network",
  edge_alpha = NULL,
  show_grid = F,
  grid_color = NULL,
  spatial_grid_name = "spatial_grid",
  midpoint = 0,
  scale_alpha_with_expression = FALSE,
  point_shape = c("border", "no_border"),
  point_size = 1,
  point_border_col = "black",
  point_border_stroke = 0.1,
  show_legend = T,
  legend_text = 8,
  background_color = "white",
  axis_text = 8,
  axis_title = 8,
  cow_n_col = 2,
  cow_rel_h = 1,
  cow_rel_w = 1,
  cow_align = "h",
  show_plot = NA,
  return_plot = NA,
  save_plot = NA,
  save_param = list(),
  default_save_name = "spatGenePlot2D"
)

```

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

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
midpoint	expression midpoint
scale_alpha_with_expression	scale expression with ggplot alpha parameter
point_shape	point with border or not (border or no_border)
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
legend_text	size of legend text
background_color	color of plot background
axis_text	size of axis text
axis_title	size of axis title
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_plot	show plots
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param
...	parameters for cowplot::save_plot()

Details

Description of parameters.

Value

ggplot

See Also

[spatGenePlot3D](#)

Examples

```
spatGenePlot2D(gobject)
```

spatGenePlot3D

spatGenePlot3D

Description

Visualize cells and gene expression according to spatial coordinates

Usage

```
spatGenePlot3D(
  gobject,
  expression_values = c("normalized", "scaled", "custom"),
  genes,
  show_network = F,
  network_color = NULL,
  spatial_network_name = "Delaunay_network",
  edge_alpha = NULL,
  show_grid = F,
  cluster_column = NULL,
  select_cell_groups = NULL,
  select_cells = NULL,
  show_other_cells = T,
  other_cell_color = "lightgrey",
  other_point_size = 1,
  genes_high_color = NULL,
  genes_mid_color = "white",
  genes_low_color = "blue",
  spatial_grid_name = "spatial_grid",
  point_size = 2,
  show_legend = T,
  axis_scale = c("cube", "real", "custom"),
  custom_ratio = NULL,
  x_ticks = NULL,
  y_ticks = NULL,
  z_ticks = NULL,
  show_plot = NA,
  return_plot = NA,
  save_plot = NA,
  save_param = list(),
  default_save_name = "spatGenePlot3D"
)
```

Arguments

<code>gobject</code>	giotto object
<code>expression_values</code>	gene expression values to use
<code>genes</code>	genes to show
<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
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
show_plot	show plots
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param
grid_color	color of spatial grid
midpoint	expression midpoint
scale_alpha_with_expression	scale expression with ggplot alpha parameter
...	parameters for cowplot::save_plot()

Details

Description of parameters.

Value

ggplot

Examples

```
spatGenePlot3D(gobject)
```

spatialAEH	<i>spatialAEH</i>
------------	-------------------

Description

Compute spatial variable genes with spatialDE method

Usage

```
spatialAEH(  
  gobject = NULL,  
  SpatialDE_results = NULL,  
  name_pattern = "AEH_patterns",  
  expression_values = c("raw", "normalized", "scaled", "custom"),  
  pattern_num = 6,  
  l = 1.05,  
  python_path = NULL,  
  return_gobject = TRUE  
)
```

Arguments

- gobject Giotto object
- SpatialDE_results results of [SpatialDE](#) function
- name_pattern name for the computed spatial patterns
- expression_values gene expression values to use
- pattern_num number of spatial patterns to look for
- l lengthscale
- python_path specify specific path to python if required
- return_gobject show plot

Details

This function is a wrapper for the SpatialAEH method implemented in the ...

Value

An updated giotto object

Examples

```
spatialAEH(gobject)
```

spatialDE	<i>spatialDE</i>
-----------	------------------

Description

Compute spatial variable genes with spatialDE method

Usage

```
spatialDE(
  gobject = NULL,
  expression_values = c("raw", "normalized", "scaled", "custom"),
  size = c(4, 2, 1),
  color = c("blue", "green", "red"),
  sig_alpha = 0.5,
  unsig_alpha = 0.5,
  python_path = NULL,
  show_plot = NA,
  return_plot = NA,
  save_plot = NA,
  save_param = list(),
  default_save_name = "SpatialDE"
)
```

Arguments

<code>gobject</code>	Giotto object
<code>expression_values</code>	gene expression values to use
<code>size</code>	size of plot
<code>color</code>	low/medium/high color scheme for plot
<code>sig_alpha</code>	alpha value for significance
<code>unsig_alpha</code>	alpha value for unsignificance
<code>python_path</code>	specify specific path to python if required
<code>show_plot</code>	show plot
<code>return_plot</code>	return ggplot object
<code>save_plot</code>	directly save the plot [boolean]
<code>save_param</code>	list of saving parameters from <code>all_plots_save_function()</code>
<code>default_save_name</code>	default save name for saving, don't change, change <code>save_name</code> in <code>save_param</code>

Details

This function is a wrapper for the SpatialDE method implemented in the ...

Value

a list of data.frames with results and plot (optional)

Examples

```
spatialDE(gobject)
```

Spatial_AEH	<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

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

Details

Description.

Value

a list or a dataframe of SVs

Examples

```
Spatial_AEH(gobject)
```

Spatial_DE

Spatial_DE

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

<code>gobject</code>	Giotto object
<code>show_plot</code>	show FSV 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)
```

spatNetwDistributions *spatNetwDistributionsDistance*

Description

This function return histograms displaying the distance distribution for each spatial k-neighbor

Usage

```

spatNetwDistributions(
  gobject,
  spatial_network_name = "spatial_network",
  distribution = c("distance", "k_neighbors"),
  hist_bins = 30,
  test_distance_limit = NULL,
  ncol = 1,
  show_plot = NA,
  return_plot = NA,
  save_plot = NA,
  save_param = list(),
  default_save_name = "spatNetwDistributions"
)

```

Arguments

<code>gobject</code>	Giotto object
<code>spatial_network_name</code>	name of spatial network
<code>distribution</code>	show the distribution of cell-to-cell distance or number of k neighbors
<code>hist_bins</code>	number of binds to use for the histogram
<code>test_distance_limit</code>	effect of different distance threshold on k-neighbors
<code>ncol</code>	number of columns to visualize the histograms in
<code>show_plot</code>	show plot
<code>return_plot</code>	return ggplot object
<code>save_plot</code>	directly save the plot [boolean]
<code>save_param</code>	list of saving parameters from all_plots_save_function
<code>default_save_name</code>	default save name for saving, alternatively change <code>save_name</code> in <code>save_param</code>

Details

The **distance** option shows the spatial distance distribution for each nearest neighbor rank (1st, 2nd, 3th, ... neighbor). With this option the user can also test the effect of a distance limit on the spatial network. This distance limit can be used to remove neighbor cells that are considered to far away. The **k_neighbors** option shows the number of k neighbors distribution over all cells.

Value

ggplot plot

Examples

```
spatNetwDistributionsDistance(gobject)
```

```
spatNetwDistributionsDistance
      spatNetwDistributionsDistance
```

Description

This function return histograms displaying the distance distribution for each spatial k-neighbor

Usage

```
spatNetwDistributionsDistance(
  gobject,
  spatial_network_name = "spatial_network",
  hist_bins = 30,
  test_distance_limit = NULL,
  ncol = 1,
  show_plot = NA,
  return_plot = NA,
  save_plot = NA,
  save_param = list(),
  default_save_name = "spatNetwDistributionsDistance"
)
```

Arguments

<code>gobject</code>	Giotto object
<code>spatial_network_name</code>	name of spatial network
<code>hist_bins</code>	number of binds to use for the histogram
<code>test_distance_limit</code>	effect of different distance threshold on k-neighbors
<code>ncol</code>	number of columns to visualize the histograms in
<code>show_plot</code>	show plot
<code>return_plot</code>	return ggplot object
<code>save_plot</code>	directly save the plot [boolean]
<code>save_param</code>	list of saving parameters from all_plots_save_function
<code>default_save_name</code>	default save name for saving, alternatively change <code>save_name</code> in <code>save_param</code>

Value

ggplot plot

Examples

```
spatNetwDistributionsDistance(gobject)
```

```
spatNetwDistributionsKneighbors
      spatNetwDistributionsKneighbors
```

Description

This function returns a histogram displaying the number of k-neighbors distribution for each cell

Usage

```
spatNetwDistributionsKneighbors(
  gobject,
  spatial_network_name = "spatial_network",
  hist_bins = 30,
  show_plot = NA,
  return_plot = NA,
  save_plot = NA,
  save_param = list(),
  default_save_name = "spatNetwDistributionsKneighbors"
)
```

Arguments

<code>gobject</code>	Giotto object
<code>spatial_network_name</code>	name of spatial network
<code>hist_bins</code>	number of binds to use for the histogram
<code>show_plot</code>	show plot
<code>return_plot</code>	return ggplot object
<code>save_plot</code>	directly save the plot [boolean]
<code>save_param</code>	list of saving parameters from all_plots_save_function
<code>default_save_name</code>	default save name for saving, alternatively change <code>save_name</code> in <code>save_param</code>

Value

ggplot plot

Examples

```
spatNetwDistributionsKneighbors(gobject)
```

spatPlot

spatPlot

Description

Visualize cells according to spatial coordinates

Usage

```
spatPlot(
  gobject,
  group_by = NULL,
  group_by_subset = NULL,
  sdimx = "sdimx",
  sdimy = "sdimy",
  spat_enr_names = NULL,
  cell_color = NULL,
  color_as_factor = T,
  cell_color_code = NULL,
  cell_color_gradient = c("blue", "white", "red"),
  gradient_midpoint = NULL,
  gradient_limits = NULL,
  select_cell_groups = NULL,
  select_cells = NULL,
  point_shape = c("border", "no_border"),
  point_size = 3,
  point_border_col = "black",
  point_border_stroke = 0.1,
  show_cluster_center = F,
  show_center_label = F,
  center_point_size = 4,
  center_point_border_col = "black",
  center_point_border_stroke = 0.1,
  label_size = 4,
  label_fontface = "bold",
  show_network = F,
  spatial_network_name = NULL,
  network_color = NULL,
  network_alpha = 1,
  show_grid = F,
  spatial_grid_name = "spatial_grid",
  grid_color = NULL,
  show_other_cells = T,
  other_cell_color = "lightgrey",
  other_point_size = 1,
  other_cells_alpha = 0.1,
  coord_fix_ratio = NULL,
  title = NULL,
  show_legend = T,
  legend_text = 8,
  legend_symbol_size = 1,
```

```

background_color = "white",
axis_text = 8,
axis_title = 8,
cow_n_col = 2,
cow_rel_h = 1,
cow_rel_w = 1,
cow_align = "h",
show_plot = NA,
return_plot = NA,
save_plot = NA,
save_param = list(),
default_save_name = "spatPlot"
)

```

Arguments

<code>gobject</code>	giotto object
<code>group_by_subset</code>	subset the <code>group_by</code> factor column
<code>sdimx</code>	x-axis dimension name (default = 'sdimx')
<code>sdimy</code>	y-axis dimension name (default = 'sdimy')
<code>spat_enr_names</code>	names of spatial enrichment results to include
<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>cell_color_gradient</code>	vector with 3 colors for numeric data
<code>gradient_midpoint</code>	midpoint for color gradient
<code>gradient_limits</code>	vector with lower and upper limits
<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>point_shape</code>	point with border or not (border or no_border)
<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_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

label_size	size of labels
label_fontface	font of labels
show_network	show underlying spatial network
spatial_network_name	name of spatial network to use
network_color	color of spatial network
network_alpha	alpha of spatial network
show_grid	show spatial grid
spatial_grid_name	name of spatial grid to use
grid_color	color of spatial grid
show_other_cells	display not selected cells
other_cell_color	color of not selected cells
other_point_size	point size of not selected cells
other_cells_alpha	alpha of not selected cells
coord_fix_ratio	fix ratio between x and y-axis
title	title of plot
show_legend	show legend
legend_text	size of legend text
legend_symbol_size	size of legend symbols
background_color	color of plot background
axis_text	size of axis text
axis_title	size of axis title
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_plot	show plot
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param
groub_by	create multiple plots based on cell annotation column

Details

Description of parameters.

Value

ggplot

See Also

[spatPlot3D](#)

Examples

```
spatPlot(gobject)
```

spatPlot2D	<i>spatPlot2D</i>
------------	-------------------

Description

Visualize cells according to spatial coordinates

Usage

```
spatPlot2D(  
  gobject,  
  group_by = NULL,  
  group_by_subset = NULL,  
  sdimx = "sdimx",  
  sdimy = "sdimy",  
  spat_enr_names = NULL,  
  cell_color = NULL,  
  color_as_factor = T,  
  cell_color_code = NULL,  
  cell_color_gradient = c("blue", "white", "red"),  
  gradient_midpoint = NULL,  
  gradient_limits = NULL,  
  select_cell_groups = NULL,  
  select_cells = NULL,  
  point_shape = c("border", "no_border"),  
  point_size = 3,  
  point_border_col = "black",  
  point_border_stroke = 0.1,  
  show_cluster_center = F,  
  show_center_label = F,  
  center_point_size = 4,  
  center_point_border_col = "black",  
  center_point_border_stroke = 0.1,  
  label_size = 4,  
  label_fontface = "bold",  
  show_network = F,  
  spatial_network_name = NULL,  
  network_color = NULL,  
  network_alpha = 1,  
  show_grid = F,
```

```

    spatial_grid_name = "spatial_grid",
    grid_color = NULL,
    show_other_cells = T,
    other_cell_color = "lightgrey",
    other_point_size = 1,
    other_cells_alpha = 0.1,
    coord_fix_ratio = NULL,
    title = NULL,
    show_legend = T,
    legend_text = 8,
    legend_symbol_size = 1,
    background_color = "white",
    axis_text = 8,
    axis_title = 8,
    cow_n_col = 2,
    cow_rel_h = 1,
    cow_rel_w = 1,
    cow_align = "h",
    show_plot = NA,
    return_plot = NA,
    save_plot = NA,
    save_param = list(),
    default_save_name = "spatPlot2D"
  )

```

Arguments

<code>gobject</code>	giotto object
<code>group_by_subset</code>	subset the <code>group_by</code> factor column
<code>sdimx</code>	x-axis dimension name (default = 'sdimx')
<code>sdimy</code>	y-axis dimension name (default = 'sdimy')
<code>spat_enr_names</code>	names of spatial enrichment results to include
<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>cell_color_gradient</code>	vector with 3 colors for numeric data
<code>gradient_midpoint</code>	midpoint for color gradient
<code>gradient_limits</code>	vector with lower and upper limits
<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>point_shape</code>	point with border or not (border or no_border)
<code>point_size</code>	size of point (cell)

point_border_col	color of border around points
point_border_stroke	stroke size of border around points
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
show_network	show underlying spatial network
spatial_network_name	name of spatial network to use
network_color	color of spatial network
network_alpha	alpha of spatial network
show_grid	show spatial grid
spatial_grid_name	name of spatial grid to use
grid_color	color of spatial grid
show_other_cells	display not selected cells
other_cell_color	color of not selected cells
other_point_size	point size of not selected cells
other_cells_alpha	alpha of not selected cells
coord_fix_ratio	fix ratio between x and y-axis
title	title of plot
show_legend	show legend
legend_text	size of legend text
legend_symbol_size	size of legend symbols
background_color	color of plot background
axis_text	size of axis text
axis_title	size of axis title
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_plot	show plot

return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param
groub_by	create multiple plots based on cell annotation column

Details

Description of parameters.

Value

ggplot

See Also

[spatPlot3D](#)

Examples

spatPlot2D(gobject)

spatPlot2D_single	<i>spatPlot2D_single</i>
-------------------	--------------------------

Description

Visualize cells according to spatial coordinates

Usage

```
spatPlot2D_single(  
  gobject,  
  sdimx = "sdimx",  
  sdimy = "sdimy",  
  spat_enr_names = NULL,  
  cell_color = NULL,  
  color_as_factor = T,  
  cell_color_code = NULL,  
  cell_color_gradient = c("blue", "white", "red"),  
  gradient_midpoint = NULL,  
  gradient_limits = NULL,  
  select_cell_groups = NULL,  
  select_cells = NULL,  
  point_shape = c("border", "no_border"),  
  point_size = 3,  
  point_border_col = "black",  
  point_border_stroke = 0.1,  
  show_cluster_center = F,  
  show_center_label = F,  
)
```

```

center_point_size = 4,
center_point_border_col = "black",
center_point_border_stroke = 0.1,
label_size = 4,
label_fontface = "bold",
show_network = F,
spatial_network_name = NULL,
network_color = NULL,
network_alpha = 1,
show_grid = F,
spatial_grid_name = "spatial_grid",
grid_color = NULL,
show_other_cells = T,
other_cell_color = "lightgrey",
other_point_size = 1,
other_cells_alpha = 0.1,
coord_fix_ratio = NULL,
title = NULL,
show_legend = T,
legend_text = 8,
legend_symbol_size = 1,
background_color = "white",
axis_text = 8,
axis_title = 8,
show_plot = NA,
return_plot = NA,
save_plot = NA,
save_param = list(),
default_save_name = "spatPlot2D_single"
)

```

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>spat_enr_names</code>	names of spatial enrichment results to include
<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>cell_color_gradient</code>	vector with 3 colors for numeric data
<code>gradient_midpoint</code>	midpoint for color gradient
<code>gradient_limits</code>	vector with lower and upper limits
<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

point_shape	point with border or not (border or no_border)
point_size	size of point (cell)
point_border_col	color of border around points
point_border_stroke	stroke size of border around points
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
show_network	show underlying spatial network
spatial_network_name	name of spatial network to use
network_color	color of spatial network
network_alpha	alpha of spatial network
show_grid	show spatial grid
spatial_grid_name	name of spatial grid to use
grid_color	color of spatial grid
show_other_cells	display not selected cells
other_cell_color	color of not selected cells
other_point_size	point size of not selected cells
other_cells_alpha	alpha of not selected cells
coord_fix_ratio	fix ratio between x and y-axis
title	title of plot
show_legend	show legend
legend_text	size of legend text
legend_symbol_size	size of legend symbols
background_color	color of plot background
axis_text	size of axis text
axis_title	size of axis title
show_plot	show plot
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param

Details

Description of parameters.

Value

ggplot

See Also

[spatPlot3D](#)

Examples

```
spatPlot2D_single(gobject)
```

spatPlot3D

spatPlot3D

Description

Visualize cells according to spatial coordinates

Usage

```
spatPlot3D(
  gobject,
  sdimx = "sdimx",
  sdimy = "sdimy",
  sdimz = "sdimz",
  point_size = 3,
  cell_color = NULL,
  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_network = F,
  network_color = NULL,
  network_alpha = 1,
  other_cell_alpha = 0.5,
  spatial_network_name = "Delaunay_network",
  show_grid = F,
  grid_color = NULL,
  spatial_grid_name = "spatial_grid",
  title = "",
  show_legend = T,
  axis_scale = c("cube", "real", "custom"),
  custom_ratio = NULL,
  x_ticks = NULL,
  y_ticks = NULL,
```

```

    z_ticks = NULL,
    show_plot = NA,
    return_plot = NA,
    save_plot = NA,
    save_param = list(),
    default_save_name = "spat3D"
)

```

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 = '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>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>title</code>	title of plot
<code>show_legend</code>	show legend
<code>axis_scale</code>	the way to scale the axis
<code>custom_ratio</code>	customize the scale of the plot
<code>x_ticks</code>	set the number of ticks on the x-axis
<code>y_ticks</code>	set the number of ticks on the y-axis
<code>z_ticks</code>	set the number of ticks on the z-axis
<code>show_plot</code>	show plot
<code>return_plot</code>	return ggplot object
<code>save_plot</code>	directly save the plot [boolean]
<code>save_param</code>	list of saving parameters from all_plots_save_function
<code>default_save_name</code>	default save name for saving, don't change, change <code>save_name</code> in <code>save_param</code>

Details

Description of parameters.

Value

ggplot

Examples

```
spatPlot3D(gobject)
```

spat_fish_func	<i>spat_fish_func</i>
----------------	-----------------------

Description

performs fisher exact test

Usage

```
spat_fish_func(gene, bin_matrix, spat_mat, calc_hub = F, hub_min_int = 3)
```

spat_OR_func	<i>spat_OR_func</i>
--------------	---------------------

Description

calculate odds-ratio

Usage

```
spat_OR_func(gene, bin_matrix, spat_mat, calc_hub = F, hub_min_int = 3)
```

```
specificCellCellcommunicationScores
      specificCellCellcommunicationScores
```

Description

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

Usage

```
specificCellCellcommunicationScores(
  gobject,
  spatial_network_name = "Delaunay_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,
  adjust_method = c("fdr", "bonferroni", "BH", "holm", "hochberg", "hommel", "BY",
    "none"),
  adjust_target = c("genes", "cells"),
  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>adjust_method</code>	which method to adjust p-values
<code>adjust_target</code>	adjust multiple hypotheses at the cell or gene level
<code>verbose</code>	verbose

Details

Statistical framework to identify if pairs of genes (such as ligand-receptor combinations) are expressed at higher levels than expected based on a reshuffled null distribution of gene expression values in cells that are spatially in proximity to eachother.. More details will follow soon.

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

dend 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 the 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

Stitching of fields:

- 1. have cell locations: at least 3 columns: field, X, Y
- 2. create offset file: offset file has 3 columns: field, x_offset, y_offset
- 3. create new cell location file by stitching original cell locations with stitchFieldCoordinates
- 4. provide new cell location file to [createGiottoObject](#)

Value

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

Examples

```
stitchFieldCoordinates(gobject)
```

stitchTileCoordinates	<i>stitchTileCoordinates</i>
-----------------------	------------------------------

Description

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

Usage

```
stitchTileCoordinates(location_file, Xtilespan, Ytilespan)
```

Arguments

- location_file location dataframe with X and Y coordinates
- Xtilespan numerical value specifying the width of each tile
- Ytilespan numerical value specifying the height of each tile

Details

...

Examples

```
stitchTileCoordinates(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

<code>gobject</code>	giotto object
<code>name</code>	name for new clustering result
<code>cluster_method</code>	clustering method 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
<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)
<code>network_name</code>	name of NN network to use
<code>return_gobject</code>	boolean: return giotto object (default = TRUE)
<code>verbose</code>	verbose

Details

This function performs subclustering on selected clusters. The systematic steps are:

- 1. subset Giotto object
- 2. identify highly variable genes
- 3. run PCA
- 4. create nearest neighbouring network
- 5. do clustering

Value

giotto object with new subclusters appended to cell metadata

See Also

[doLouvainCluster_multinet](#), [doLouvainCluster_community](#) and [@seealso doLeidenCluster](#)

Examples

```
subClusterCells(gobject)
```

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

Description

subsets Giotto object including previous analyses.

Usage

```
subsetGiotto(gobject, cell_ids = NULL, gene_ids = NULL, verbose = FALSE)
```

Arguments

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

Value

giotto object

Examples

```
subsetGiotto(gobject)
```

subsetGiottoLocs	<i>subsetGiottoLocs</i>
------------------	-------------------------

Description

subsets Giotto object based on spatial locations

Usage

```
subsetGiottoLocs(  
  gobject,  
  x_max = NULL,  
  x_min = NULL,  
  y_max = NULL,  
  y_min = NULL,  
  z_max = NULL,  
  z_min = NULL,  
  return_gobject = T,  
  verbose = FALSE  
)
```

Arguments

gobject	giotto object
x_max	maximum x-coordinate
x_min	minimum x-coordinate
y_max	maximum y-coordinate
y_min	minimum y-coordinate
z_max	maximum z-coordinate
z_min	minimum z-coordinate
return_gobject	return Giotto object

Details

if return_gobject = FALSE, then a filtered combined metadata data.table will be returned

Value

giotto object

Examples

```
subsetGiottoLocs(gobject)
```

```
transform_2d_mesh_to_3d_mesh
      transform_2d_mesh_to_3d_mesh
```

Description

transform 2d mesh to 3d mesh by reversing PCA

Usage

```
transform_2d_mesh_to_3d_mesh(
  mesh_line_obj_2d,
  pca_out,
  center_vec,
  mesh_grid_n
)
```

```
trendSceek      trendSceek
```

Description

Compute spatial variable genes with trendsceek method

Usage

```
trendSceek(
  gobject,
  expression_values = c("normalized", "raw"),
  subset_genes = NULL,
  nrand = 100,
  ncores = 8,
  ...
)
```

Arguments

<code>gobject</code>	Giotto object
<code>expression_values</code>	gene expression values to use
<code>subset_genes</code>	subset of genes to run trendsceek on
<code>nrand</code>	An integer specifying the number of random resamplings of the mark distribution as to create the null-distribution.
<code>ncores</code>	An integer specifying the number of cores to be used by BiocParallel
<code>...</code>	Additional parameters to the trendsceek_test function

Details

This function is a wrapper for the `trendsceek_test` method implemented in the `trendsceek` package

Value

data.frame with trendsceek spatial genes results

Examples

```
trendSceek(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)
```

viewHMRFresults2D	<i>viewHMRFresults2D</i>
-------------------	--------------------------

Description

View results from doHMRF.

Usage

```
viewHMRFresults2D(  
  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

[spatPlot2D](#)

Examples

```
viewHMRFresults2D(gobject)
```

viewHMRFresults3D	<i>viewHMRFresults3D</i>
-------------------	--------------------------

Description

View results from doHMRF.

Usage

```
viewHMRFresults3D(  
  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

[spatPlot3D](#)

Examples

```
viewHMRFresults3D(gobject)
```

violinPlot

violinPlot

Description

Creates violinplot for selected 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_position = c("top", "right", "left", "bottom"),
  strip_text = 7,
  axis_text_x_size = 10,
  axis_text_y_size = 6,
  show_plot = NA,
  return_plot = NA,
  save_plot = NA,
  save_param = list(),
  default_save_name = "violinPlot"
)
```

Arguments

gobject	giotto object
expression_values	expression values to use
genes	genes to plot
cluster_column	name of column to use for clusters
cluster_custom_order	custom order of clusters
color_violin	color violin according to genes or clusters
cluster_color_code	color code for clusters
strip_position	position of gene labels
strip_text	size of strip text
axis_text_x_size	size of x-axis text
axis_text_y_size	size of y-axis text
show_plot	show plot
return_plot	return ggplot object

save_plot	directly save the plot [boolean]
save_param	list of saving parameters from all_plots_save_function
default_save_name	default save name for saving, don't change, change save_name in save_param

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 = FALSE,
  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

<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>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>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
<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>show_plots</code>	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 = FALSE,
  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

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
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_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

<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>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>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
<code>show_plots</code>	show plots

Details

Description of parameters.

Value

ggplot

Examples

```
visDimGenePlot_3D_plotly(gobject)
```

visDimPlot	<i>visDimPlot</i>
------------	-------------------

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,  
  show_plot = F,  
  return_plot = TRUE,  
  save_plot = F,  
  save_dir = NULL,  
)
```

```

    save_folder = NULL,
    save_name = NULL,
    save_format = NULL,
    show_saved_plot = F,
    ...
)

```

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
<code>show_plot</code>	show plot
<code>return_plot</code>	return ggplot object
<code>save_plot</code>	directly save the plot [boolean]
<code>save_dir</code>	directory to save the plot

save_folder	(optional) folder in directory to save the plot
save_name	name of plot
save_format	format of plot (e.g. tiff, png, pdf, ...)
show_saved_plot	load & display the saved plot

Details

Description of parameters.

Value

ggplot or plotly

Examples

```
visDimPlot(gobject)
```

visDimPlot_2D_ggplot	<i>visDimPlot_2D_ggplot</i>
----------------------	-----------------------------

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,
    show_plot = F,
    return_plot = TRUE,
    save_plot = F,
    save_dir = NULL,
    save_folder = NULL,
    save_name = NULL,
    save_format = NULL,
    show_saved_plot = F,
    ...
)

```

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>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>other_point_size</code>	size of not selected cells
<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

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 *visDimPlot_2D_plotly*

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,
  select_cell_groups = NULL,
  select_cells = NULL,
  show_other_cells = T,
  other_cell_color = "lightgrey",
  other_point_size = 0.5,
  show_NN_network = F,
  nn_network_to_use = "sNN",
  network_name = "sNN.pca",
  color_as_factor = T,
  cell_color = NULL,
  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
    )

```

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>color_as_factor</code>	convert color column to factor
<code>cell_color</code>	color for cells (see details)
<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)

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,
  select_cell_groups = NULL,
  select_cells = NULL,
  show_other_cells = T,
  other_cell_color = "lightgrey",
  other_point_size = 0.5,
  show_NN_network = F,
  nn_network_to_use = "sNN",
  network_name = "sNN.pca",
  color_as_factor = T,
  cell_color = NULL,
  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
)
```

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
color_as_factor	convert color column to factor
cell_color	color for cells (see details)
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)

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 = TRUE,
  cell_color_code = NULL,
  edge_alpha = NULL,
  point_size = 1,
```

```

    point_border_col = "black",
    point_border_stroke = 0.1,
    show_legend = T,
    show_plot = F,
    return_plot = TRUE,
    save_plot = F,
    save_dir = NULL,
    save_folder = NULL,
    save_name = NULL,
    save_format = NULL,
    show_saved_plot = F,
    ...
)

```

Arguments

<code>gobject</code>	giotto object
<code>nn_network_to_use</code>	type of NN network to use (kNN vs sNN)
<code>network_name</code>	NN network to use
<code>layout_name</code>	name of layout to use
<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>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>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
<code>show_plot</code>	show plot
<code>return_plot</code>	return ggplot object
<code>save_plot</code>	directly save the plot [boolean]
<code>save_dir</code>	directory to save the plot
<code>save_folder</code>	(optional) folder in directory to save the plot
<code>save_name</code>	name of plot
<code>save_format</code>	format of plot (e.g. tiff, png, pdf, ...)
<code>show_saved_plot</code>	load & display the saved plot

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 = FALSE,
  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
<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>axis_scale</code>	three mode to adjust axis scale
<code>x_ticks</code>	number of ticks on x axis
<code>y_ticks</code>	number of ticks on y axis
<code>z_ticks</code>	number of ticks on z axis
<code>plot_method</code>	two methods of plot
<code>show_plots</code>	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 = FALSE,
  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

genes_low_color	color represents low gene expression
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
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
show_plots	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

<code>gobject</code>	giotto object
<code>expression_values</code>	gene expression values to use
<code>genes</code>	genes to show
<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>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>spatial_grid_name</code>	name of spatial grid to use
<code>point_size</code>	size of point (cell)
<code>show_legend</code>	show legend
<code>axis_scale</code>	three mode to adjust axis scale
<code>x_ticks</code>	number of ticks on x axis
<code>y_ticks</code>	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_cell_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,
    return_plot = TRUE,
    save_plot = F,
    save_dir = NULL,
    save_folder = NULL,
    save_name = NULL,
    save_format = NULL,
    show_saved_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

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
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_dir	directory to save the plot
save_folder	(optional) folder in directory to save the plot
save_name	name of plot
save_format	format of plot (e.g. tiff, png, pdf, ...)
show_saved_plot	load & display the saved 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,  
  sdimx = 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,
    return_plot = TRUE,
    save_plot = F,
    save_dir = NULL,
    save_folder = NULL,
    save_name = NULL,
    save_format = NULL,
    show_saved_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>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

show_other_cells	display not selected cells
other_cell_color	color of not selected cells
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
coord_fix_ratio	fix ratio between x and y-axis
title	title of plot
show_legend	show legend
show_plot	show plot
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_dir	directory to save the plot
save_folder	(optional) folder in directory to save the plot
save_name	name of plot
save_format	format of plot (e.g. tiff, png, pdf, ...)
show_saved_plot	load & display the saved 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,
  sdimx = 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,
  other_cell_color = "lightgrey",
  other_point_size = 0.5,
  show_network = F,
  network_color = "lightgray",
  network_alpha = 1,
  other_cell_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

select_cell_groups	select a subset of the groups from cell_color
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
grid_alpha	alpha of spatial grid
spatial_grid_name	name of spatial grid to use
show_legend	show legend
show_plot	show plot

Details

Description of parameters.

Value

plotly

Examples

```
visPlot_2D_plotly(gobject)
```

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

Description

Visualize cells according to spatial coordinates

Usage

```
visPlot_3D_plotly(
  gobject,
  sdimx = NULL,
  sdimy = NULL,
  sdimz = NULL,
  point_size = 3,
  cell_color = NULL,
  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_network = F,
```

```

    network_color = NULL,
    network_alpha = 1,
    other_cell_alpha = 0.5,
    spatial_network_name = "spatial_network",
    spatial_grid_name = "spatial_grid",
    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>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>cell_color</code>	color for cells (see details)
<code>cell_color_code</code>	named vector with colors
<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>spatial_grid_name</code>	name of spatial grid to use
<code>title</code>	title of plot
<code>show_legend</code>	show legend
<code>show_plot</code>	show plot
<code>point_border_col</code>	color of border around points
<code>point_border_stroke</code>	stroke size of border around points
<code>color_as_factor</code>	convert color column to factor
<code>show_grid</code>	show spatial grid
<code>grid_color</code>	color of spatial grid
<code>coord_fix_ratio</code>	fix ratio between x and y-axis

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 = FALSE,
  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

<code>gobject</code>	giotto object
<code>expression_values</code>	gene expression values to use
<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>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>genes</code>	genes to show
<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>

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 = FALSE,
  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

gobject giotto object

expression_values	gene expression values to use
plot_alignment	direction to align plot
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
point_size	size of point (cell)
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
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
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
dim_point_size	dim reduction plot: point size
show_plot	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
<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>genes_low_color</code>	color represent high gene expression (see details)
<code>genes_high_color</code>	color represent high gene expression (see details)
<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>network_color</code>	color of spatial/nn network
<code>spatial_network_alpha</code>	alpha of spatial network
<code>show_spatial_grid</code>	show spatial grid
<code>spatial_grid_name</code>	name of spatial grid to use
<code>spatial_grid_color</code>	color of spatial grid
<code>spatial_grid_alpha</code>	alpha of spatial grid
<code>legend_text_size</code>	text size of legend
<code>show_legend</code>	show legend
<code>show_plot</code>	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,
  sdims = NULL,
  sdims = NULL,
  sdims = 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)
<code>color_as_factor</code>	convert color column to factor
<code>cell_color_code</code>	named vector with colors
<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>nn_network_alpha</code>	column to use for alpha of the edges
<code>show_spatial_network</code>	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,
  sdims = 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)

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_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
return_plot	return ggplot object
save_plot	directly save the plot [boolean]
save_dir	directory to save the plot
save_folder	(optional) folder in directory to save the plot
save_name	name of plot
save_format	format of plot (e.g. tiff, png, pdf, ...)
show_saved_plot	load & display the saved 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

gobject giotto object

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
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
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
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)
```

writeHMRResults	<i>writeHMRResults</i>
-----------------	------------------------

Description

write results from doHMRF to a data.table.

Usage

```
writeHMRResults(
  gobject,
  HMRFoutput,
  k = NULL,
  betas_to_view = NULL,
  print_command = F
)
```

Arguments

gobject	giotto object
HMRFoutput	HMRF output from doHMRF
k	k to write results for
betas_to_view	results from different betas that you want to view
print_command	see the python command

Value

data.table with HMRF results for each b and the selected k

Examples

```
writeHMRResults(gobject)
```

write_giotto_viewer_annotation	<i>write_giotto_viewer_annotation</i>
--------------------------------	---------------------------------------

Description

write out factor-like 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

```
write_giotto_viewer_numeric_annotation  
    write_giotto_viewer_numeric_annotation
```

Description

write out numeric annotation data from a giotto object for the Viewer

Usage

```
write_giotto_viewer_numeric_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

Index

- *Topic **giotto**,
 - giotto-class, [166](#)
 - print.giotto, [233](#)
 - show,giotto-method, [245](#)
- *Topic **giotto**
 - createGiottoObject, [54](#)
- *Topic **object**
 - giotto-class, [166](#)
 - print.giotto, [233](#)
 - show,giotto-method, [245](#)
- adapt_aspect_ratio, [8](#)
- addCellIntMetadata, [8](#)
- addCellMetadata, [9](#), [56](#)
- addCellStatistics, [10](#), [15](#)
- addGeneMetadata, [11](#), [56](#)
- addGeneStatistics, [12](#), [15](#)
- addHMRF, [13](#)
- addNetworkLayout, [13](#)
- addStatistics, [14](#)
- adjustGiottoMatrix, [15](#)
- aes_string2, [16](#)
- all_plots_save_function, [16](#), [23](#), [26](#), [28](#),
[30](#), [31](#), [33](#), [35](#), [77](#), [79](#), [83](#), [85](#), [90](#), [93](#),
[95](#), [98](#), [99](#), [102](#), [105](#), [108](#), [110](#), [141](#),
[143](#), [168](#), [170](#), [173](#), [185–188](#),
[190–192](#), [194–197](#), [201](#), [203](#), [206](#),
[208–211](#), [213](#), [215](#), [217](#), [219](#), [221](#),
[223](#), [225](#), [236](#), [246](#), [247](#), [249–251](#),
[261](#), [263](#), [268](#), [272](#), [274](#), [277](#), [279](#),
[283](#), [288](#), [290](#), [293](#), [295](#), [297](#),
[302–304](#), [307](#), [311](#), [313](#), [315](#), [329](#)
- annotate_spatlocs_with_spatgrid_2D, [19](#)
- annotate_spatlocs_with_spatgrid_3D, [19](#)
- annotateGiotto, [17](#)
- annotateSpatialNetwork, [18](#)
- average_gene_gene_expression_in_groups,
[20](#)
- binSpect, [20](#)
- calculate_distance_and_weight, [25](#)
- calculateHVG, [22](#)
- calculateMetaTable, [24](#)
- calculateMetaTableCells, [24](#)
- cellProximityBarplot, [25](#)
- cellProximityEnrichment, [27](#)
- cellProximityHeatmap, [28](#)
- cellProximityNetwork, [29](#)
- cellProximitySpatPlot, [30](#)
- cellProximitySpatPlot2D, [31](#), [32](#)
- cellProximitySpatPlot3D, [31](#), [34](#)
- cellProximityVisPlot, [36](#)
- cellProximityVisPlot_2D_ggplot, [38](#)
- cellProximityVisPlot_2D_plotly, [40](#)
- cellProximityVisPlot_3D_plotly, [41](#)
- changeGiottoInstructions, [43](#)
- cluster_walktrap, [127](#)
- clusterCells, [44](#)
- clusterSpatialCorGenes, [46](#)
- combCCcom, [47](#), [210–212](#)
- combineCellProximityGenes, [48](#)
- combineCellProximityGenes_per_interaction,
[49](#)
- combineCPG, [49](#)
- combineMetadata, [50](#)
- convert_to_full_spatial_network, [51](#)
- convert_to_reduced_spatial_network, [52](#)
- convertEnsemblToGeneSymbol, [51](#)
- cowplot::save_plot, [166](#)
- create_2d_mesh_grid_line_obj, [67](#)
- create_average_detection_DT, [68](#)
- create_average_DT, [68](#)
- create_cell_type_random_cell_IDs, [69](#)
- create_cluster_matrix, [70](#)
- create_crossSection_object, [70](#)
- create_delaunayNetwork2D, [71](#)
- create_delaunayNetwork3D, [71](#)
- create_delaunayNetwork_deldir, [72](#)
- create_delaunayNetwork_geometry, [72](#)
- create_delaunayNetwork_geometry_3D, [73](#)
- create_delaunayNetwork_RTriangle, [73](#)
- create_dimObject, [74](#)
- create_KNNnetwork_dbSCAN, [74](#)
- create_mesh_grid_lines, [75](#)
- create_spatialNetworkObject, [75](#)
- createCrossSection, [52](#)

- createDelaunayNetwork, 213
- createGiottoInstructions, 53, 55
- createGiottoObject, 54, 319
- createHeatmap_DT, 56
- createMetagenes, 57
- createNearestNetwork, 58
- createSpatialDelaunayNetwork, 60
- createSpatialEnrich, 61, 136
- createSpatialGrid, 62
- createSpatialGrid_2D, 63
- createSpatialGrid_3D, 64
- createSpatialKNNnetwork, 65
- createSpatialNetwork, 66, 87
- crossSectionGenePlot, 76
- crossSectionGenePlot3D, 78
- crossSectionPlot, 80, 83
- crossSectionPlot3D, 83
- decide_cluster_order, 85
- delaunayn, 61
- deldir, 61
- detectSpatialCorGenes, 86
- detectSpatialPatterns, 87
- dimCellPlot, 88
- dimCellPlot2D, 91, 91
- dimGenePlot, 94
- dimGenePlot2D, 96
- dimGenePlot3D, 96, 98, 98
- dimPlot, 100
- dimPlot2D, 102, 103, 206, 208, 215, 217, 221, 223
- dimPlot2D_single, 106
- dimPlot3D, 94, 102, 105, 108, 109
- do_cell_proximity_test, 129
- do_limtest, 129
- do_multi_permuttest_random, 130
- do_page_permutation, 130
- do_permuttest (do_permuttest_random), 131
- do_permuttest_original, 131
- do_permuttest_random, 131
- do_rank_permutation, 132
- do_spatial_grid_averaging, 132
- do_spatial_knn_smoothing, 133
- do_ttest, 134
- do_wilctest (do_ttest), 134
- doHclust, 46, 111
- doHMRf, 112
- doKmeans, 46, 113
- doLeidenCluster, 46, 115, 118, 322
- doLeidenSubCluster, 116
- doLouvainCluster, 46, 118
- doLouvainCluster_community, 46, 119, 119, 123, 124, 322
- doLouvainCluster_multinet, 46, 119, 120, 123, 126, 322
- doLouvainSubCluster, 121
- doLouvainSubCluster_community, 123
- doLouvainSubCluster_multinet, 125
- doRandomWalkCluster, 46, 126
- doSNNCluster, 46, 128
- DT_removeNA, 134
- dt_to_matrix, 135
- estimateCellCellDistance, 135
- exportGiottoViewer, 135
- exprCellCellcom, 137, 184, 185
- extend_vector, 138
- extended_gini_fun, 138
- extractNearestNetwork, 138
- fDataDT, 139
- filter_network, 145
- filterCellProximityGenes, 139
- filterCombinations, 140, 144
- filterCPG, 141
- filterDistributions, 142
- filterGiotto, 144
- find_grid_2D, 159
- find_grid_3D, 159
- find_grid_x, 159
- find_grid_y, 159
- find_grid_z, 160
- find_x_y_ranges, 160
- findCellProximityGenes, 145
- findCellProximityGenes_per_interaction, 147
- findCPG, 147
- findGiniMarkers, 149, 151, 153
- findGiniMarkers_one_vs_all, 150, 154
- findMarkers, 152, 157
- findMarkers_one_vs_all, 153
- findMastMarkers, 153, 155, 156
- findMastMarkers_one_vs_all, 154, 156
- findScranMarkers, 153, 157, 158
- findScranMarkers_one_vs_all, 154, 158
- general_save_function, 17, 160
- get10Xmatrix, 161
- get_cross_section_coordinates, 164
- get_distance, 164
- get_sectionThickness, 165
- getClusterSimilarity, 162
- getDendrogramSplits, 163
- getDistinctColors, 164

- ggplot_save_function, 165
- giotto (giotto-class), 166
- giotto-class, 166
- glouvain_ml, 121
- hclust, 112
- Heatmap, 168
- heatmSpatialCorGenes, 167
- hyperGeometricEnrich, 62, 168
- insertCrossSectionGenePlot3D, 169
- insertCrossSectionSpatPlot3D, 171
- kmeans, 114
- kmeans_binarize, 173
- kNN, 59
- layout_with_drl, 14
- loadHMF, 174
- make_simulated_network, 176
- makeSignMatrixPAGE, 174, 183
- makeSignMatrixRank, 175, 235
- mergeClusters, 176
- my_arowMeans, 178
- my_growMeans, 178
- my_rowMeans, 178
- mygini_fun, 177
- nnDT_to_kNN, 179
- node_clusters, 179
- normalizeGiotto, 180
- normalizeGiottoOld, 181
- PAGEEnrich, 62, 175, 182
- PCA, 240
- pDataDT, 183
- permutationPA, 255
- plot_network_layer_ggplot, 225
- plot_point_layer_ggplot, 226
- plot_point_layer_ggplot_noFILL, 228
- plot_spat_point_layer_ggplot, 229
- plot_spat_point_layer_ggplot_noFILL, 231
- plotCCcomDotplot, 184
- plotCCcomHeatmap, 185
- plotCellProximityGenes, 186
- plotCombineCCcom, 188
- plotCombineCellCellCommunication, 189
- plotCombineCellProximityGenes, 190
- plotCombineCPG, 191
- plotCPG, 193
- plotHeatmap, 194
- plotICG, 196
- plotInteractionChangedGenes, 197
- plotly_axis_scale_2D, 198
- plotly_axis_scale_3D, 198
- plotly_grid, 199
- plotly_network, 200
- plotMetaDataCellsHeatmap, 200, 204
- plotMetaDataHeatmap, 202, 202
- plotPCA, 204
- plotPCA_2D, 206
- plotPCA_3D, 206, 208, 208
- plotRankSpatvsExpr, 210
- plotRecovery, 211
- plotRecovery_sub, 212
- plotStatDelaunayNetwork, 212
- plotTSNE, 213
- plotTSNE_2D, 215
- plotTSNE_3D, 215, 217, 218
- plotUMAP, 219
- plotUMAP_2D, 221
- plotUMAP_3D, 221, 223, 223
- print.giotto, 233
- projection_fun, 234
- rank_binarize, 236
- rankEnrich, 62, 175, 234
- rankSpatialCorGroups, 235
- read_crossSection, 237
- readGiottoInstructions, 236
- removeCellAnnotation, 237
- removeGeneAnnotation, 238
- replaceGiottoInstructions, 238
- reshape_to_data_point, 239
- reshape_to_mesh_grid_obj, 239
- Rtsne, 241
- runPCA, 239
- runtSNE, 240
- runUMAP, 242
- select_expression_values, 244
- select_spatialNetwork, 244
- selectPatternGenes, 243
- set_giotto_python_path, 245
- show, giotto-method, 245
- showClusterDendrogram, 245
- showClusterHeatmap, 246
- showGiottoInstructions, 248
- showPattern, 248
- showPattern2D, 249, 249
- showPattern3D, 250
- showPatternGenes, 251
- showProcessingSteps, 253
- showSpatialCorGenes, 87, 253
- signPCA, 254

- silhouetteRank, [256](#)
- sNN, [59](#)
- sNNclust, [128](#)
- sort_combine_two_DT_columns, [257](#)
- spat_fish_func, [316](#)
- spat_OR_func, [316](#)
- spatCellCellcom, [184](#), [185](#), [257](#)
- spatCellPlot, [258](#)
- spatCellPlot2D, [261](#)
- spatDimCellPlot, [264](#)
- spatDimCellPlot2D, [268](#)
- spatDimGenePlot, [272](#)
- spatDimGenePlot2D, [275](#)
- spatDimGenePlot3D, [275](#), [277](#), [278](#)
- spatDimPlot, [280](#)
- spatDimPlot2D, [284](#), [284](#)
- spatDimPlot3D, [284](#), [288](#), [288](#)
- spatGenePlot, [291](#)
- spatGenePlot2D, [78](#), [293](#), [293](#)
- spatGenePlot3D, [78](#), [293](#), [295](#), [296](#)
- Spatial_AEH, [300](#)
- Spatial_DE, [301](#)
- spatialAEH, [297](#)
- SpatialDE, [298](#)
- spatialDE, [298](#)
- spatNetDistributions, [301](#)
- spatNetDistributionsDistance, [303](#)
- spatNetDistributionsKneighbors, [304](#)
- spatPlot, [305](#)
- spatPlot2D, [308](#), [326](#)
- spatPlot2D_single, [311](#)
- spatPlot3D, [308](#), [311](#), [314](#), [314](#), [327](#)
- specificCellCellcommunicationScores, [317](#)
- split_dendrogram_in_two, [318](#)
- stitchFieldCoordinates, [56](#), [318](#)
- stitchTileCoordinates, [320](#)
- subClusterCells, [320](#)
- subsetGiotto, [322](#)
- subsetGiottoLocs, [323](#)
- transform_2d_mesh_to_3d_mesh, [324](#)
- trendSceek, [324](#)
- trendsceek_test, [324](#)
- triangulate, [61](#), [213](#)
- umap, [243](#)
- viewHMRResults, [325](#)
- viewHMRResults2D, [326](#)
- viewHMRResults3D, [327](#)
- violinPlot, [328](#)
- visDimGenePlot, [329](#)
- visDimGenePlot_2D_ggplot, [331](#)
- visDimGenePlot_3D_plotly, [332](#)
- visDimPlot, [334](#)
- visDimPlot_2D_ggplot, [336](#)
- visDimPlot_2D_plotly, [338](#)
- visDimPlot_3D_plotly, [340](#)
- visForceLayoutPlot, [341](#)
- visGenePlot, [343](#)
- visGenePlot_2D_ggplot, [345](#)
- visGenePlot_3D_plotly, [346](#)
- visPlot, [325](#), [348](#)
- visPlot_2D_ggplot, [350](#)
- visPlot_2D_plotly, [353](#)
- visPlot_3D_plotly, [354](#)
- visSpatDimGenePlot, [356](#)
- visSpatDimGenePlot_2D, [359](#)
- visSpatDimGenePlot_3D, [361](#)
- visSpatDimPlot, [363](#)
- visSpatDimPlot_2D, [365](#)
- visSpatDimPlot_3D, [368](#)
- write_giotto_viewer_annotation, [370](#)
- write_giotto_viewer_dim_reduction, [371](#)
- write_giotto_viewer_numeric_annotation, [372](#)
- writeHMRResults, [370](#)
- zlm, [155](#)