Package 'spatialGE'

May 13, 2025

Title Visualization and Analysis of Spatial Heterogeneity in
Spatially-Resolved Gene Expression

Version 1.2.0

Description Visualization and analysis of spatially resolved transcriptomics data. The 'spatialGE'
R package provides methods for visualizing and analyzing spatially resolved transcriptomics data,
such as 10X Visium, CosMx, or csv/tsv gene expression matrices. It includes tools for spatial
interpolation, autocorrelation analysis, tissue domain detection, gene set enrichment, and
differential expression analysis using spatial mixed models.

License MIT + file LICENSE

Copyright ins/COPYRIGHT

Encoding UTF-8

RoxygenNote 7.3.2

Imports arrow, BiocParallel, concaveman, ComplexHeatmap, data.table,
DelayedArray, DelayedMatrixStats, dynamicTreeCut, dplyr,
EBImage, ggforce, ggplot2, ggpolypath, ggrepel, gstat, GSVA,
hdf5r, jpeg, jsonlite, khroma (>= 1.6.0), magrittr, Matrix,
MASS, methods, parallel, png, RColorBrewer, Rcpp (>= 1.0.7),
readr, readxl, rlang, scales, sctransform, sfsmisc, sf, sp,
spaMM, spdep, stats, stringr, tibble, tidyr, utils, uwot,
wordspace

LinkingTo Rcpp, RcppEigen, RcppProgress

Suggests geoR, ggpubr, janitor, kableExtra, knitr, msigdbr, progress,
rmarkdown, rpart, R.utils, scSpatialSIM, spatstat,
SeuratObject, tidyverse, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation yes

Author Oscar Ospina [aut, cre] (ORCID:
<https://orcid.org/0000-0001-5986-4207>),
Alex Soupir [aut] (ORCID: <https://orcid.org/0000-0003-1251-9179>),
Brooke Fridley [aut] (ORCID: <https://orcid.org/0000-0001-7739-7956>),
Satija Lab [cph] (Copyright holder of code fragments from Seurat
function FindVariableFeatures)

Maintainer Oscar Ospina <oscar.ospina@jhmi.edu>

Repository CRAN

Date/Publication 2025-05-13 08:50:02 UTC

Contents

compare_SThet	*compare_SThet: Compares spatial autocorrelation statistics across samples*

Description

Plots the spatial autocorrelation statistics of genes across samples and colors samples acording to sample metadata.

Usage

```
compare_SThet(
  x = NULL,
  samplemeta = NULL,
  genes = NULL,
  color_by = NULL,
  categorical = TRUE,
  color_pal = "muted",
  ptsize = 1
)
```

Arguments

x	an STlist.
samplemeta	a string indicating the name of the variable in the clinical data frame. If NULL, uses sample names
genes	the name(s) of the gene(s) to plot.
color_by	the variable in x@spatial_meta used to color points in the plot. If NULL, each sample is assigned a different color
categorical	logical indicating whether or not to treat color_by as a categorical variable. Default is TRUE
color_pal	a string of a color palette from khroma or RColorBrewer, or a vector with colors with enough elements to plot categories.
ptsize	a number specifying the size of the points. Passed to the size aesthetic.

Details

This function takes the names of genes and their Moran's I or Geary's C computed for multiple samples and to provide a muti-sample comparison. Samples in the plot can be colored according to sample metadata to explore potential associations between spatial distribution of gene expression and sample-level data.

Value

a list of plots

---

dim,STlist-method   *dim: Prints the dimensions of count arrays within an STList object.*

Description

Returns the number of genes and spots for each array within an STList object

Usage

```
## S4 method for signature 'STlist'
dim(x)
```

Arguments

x	an STList object to show summary from.

Details

This function takes an STList and prints the number of genes (rows) and spots (columns) of each spatial array within that object.

---

distribution_plots	*per_unit_counts: Generates distribution plots of spot/cell meta data or gene expression*

Description

Generates violin plots, boxplots, or density plots of variables in the spatial meta data or of gene expression

Usage

```
distribution_plots(
  x = NULL,
  plot_meta = NULL,
  genes = NULL,
  samples = NULL,
  data_type = "tr",
  color_pal = "okabeito",
  plot_type = "violin",
  ptsize = 0.5,
  ptalpha = 0.5
)
```

Arguments

x	an STlist
plot_meta	vector of variables in x@spatial_meta to plot distributions. If 'total_counts', the function plots the counts per spot/cell. If 'total_genes', the function plots the number of genes per spot/cell are plotted
genes	vector of genes to plot expression distribution. If used in conjunction with plot_meta, the expression values are grouped using that variable
samples	samples to include in the plot. Default (NULL) includes all samples
data_type	one of 'tr' or 'raw', to plot transformed or raw counts

color_pal	a string of a color palette from khroma or RColorBrewer, or a vector with colors
plot_type	one of "violin", "box", or "density" (violin plots, box plots, or density plots respectively). If plot_meta and gene are used together, then density plots are disabled
ptsize	the size of points in the plots
ptalpha	the transparency of points (violin/box plot) or curves (density plots)

Details

The function allows to visualize the distribution of spot/cell total counts, total genes, or expression of specific genes across all samples for comparative purposes. It also allows grouping of gene expression values by categorical variables (e.g., clusters).

Value

a list containing ggplot2 objects

---

filter_data   *filter_data: Filters cells/spots, genes, or samples*

Description

Filtering of spots/cells, genes or samples, as well as count-based filtering

Usage

```
filter_data(
  x = NULL,
  spot_minreads = 0,
  spot_maxreads = NULL,
  spot_mingenes = 0,
  spot_maxgenes = NULL,
  spot_minpct = 0,
  spot_maxpct = NULL,
  gene_minreads = 0,
  gene_maxreads = NULL,
  gene_minspots = 0,
  gene_maxspots = NULL,
  gene_minpct = 0,
  gene_maxpct = NULL,
  samples = NULL,
  rm_tissue = NULL,
  rm_spots = NULL,
  rm_genes = NULL,
  rm_genes_expr = NULL,
  spot_pct_expr = "^MT-"
)
```

Arguments

x	an STlist
spot_minreads	the minimum number of total reads for a spot to be retained
spot_maxreads	the maximum number of total reads for a spot to be retained
spot_mingenes	the minimum number of non-zero counts for a spot to be retained
spot_maxgenes	the maximum number of non-zero counts for a spot to be retained
spot_minpct	the minimum percentage of counts for features defined by spot_pct_expr for a spot to be retained.
spot_maxpct	the maximum percentage of counts for features defined by spot_pct_expr for a spot to be retained.
gene_minreads	the minimum number of total reads for a gene to be retained
gene_maxreads	the maximum number of total reads for a gene to be retained
gene_minspots	he minimum number of spots with non-zero counts for a gene to be retained
gene_maxspots	the maximum number of spots with non-zero counts for a gene to be retained
gene_minpct	the minimum percentage of spots with non-zero counts for a gene to be retained
gene_maxpct	the maximum percentage of spots with non-zero counts for a gene to be retained
samples	samples (as in names(x@counts)) to perform filtering.
rm_tissue	sample (as in names(x@counts)) to remove from STlist. Removes samples in x@counts, x@tr_counts, x@spatial_meta, x@gene_meta, and x@sample_meta
rm_spots	vector of spot/cell IDs to remove. Removes spots/cells in x@counts, x@tr_counts, and x@spatial_meta
rm_genes	vector of gene names to remove from STlist. Removes genes in x@counts, x@tr_counts, and x@gene_meta
rm_genes_expr	a regular expression that matches genes to remove. Removes genes in x@counts, x@tr_counts, and x@gene_meta
spot_pct_expr	a expression to use with spot_minpct and spot_maxpct. By default '^MT-'.

Details

This function provides options to filter elements in an STlist. It can remove cells/spots or genes based on raw counts (x@counts). Users can input an regular expression to query gene names and calculate percentages (for example % mtDNA genes). The function also can filter entire samples. Note that the function removes cells/spots, genes, and/or samples in the raw counts, transformed counts, spatial variables, gene variables, and sample metadata. Also note that the function filters in the following order:

1. Samples (rm_tissue)
2. Spots (rm_spots)
3. Genes (rm_genes)
4. Genes matching rm_genes_expr
5. Min and max counts

Value

an STlist containing the filtered data

Examples

```
# Using included melanoma example (Thrane et al.)
# Download example data set from spatialGE_Data
thrane_tmp = tempdir()
unlink(thrane_tmp, recursive=TRUE)
dir.create(thrane_tmp)
lk='https://github.com/FridleyLab/spatialGE_Data/raw/refs/heads/main/melanoma_thrane.zip?download='
download.file(lk, destfile=paste0(thrane_tmp, '/', 'melanoma_thrane.zip'), mode='wb')
zip_tmp = list.files(thrane_tmp, pattern='melanoma_thrane.zip$', full.names=TRUE)
unzip(zipfile=zip_tmp, exdir=thrane_tmp)
# Generate the file paths to be passed to the STlist function
count_files <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                          full.names=TRUE, pattern='counts')
coord_files <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                          full.names=TRUE, pattern='mapping')
clin_file <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                        full.names=TRUE, pattern='clinical')
# Create STlist
library('spatialGE')
melanoma <- STlist(rnacounts=count_files[c(1,2)],
                   spotcoords=coord_files[c(1,2)],
                   samples=clin_file) # Only first two samples
melanoma <- filter_data(melanoma, spot_minreads=2000)
```

---

gene_interpolation   *gene_interpolation: Spatial interpolation of gene expression*

Description

Performs spatial interpolation ("kriging") of transformed gene counts

Usage

```
gene_interpolation(
  x = NULL,
  genes = "top",
  top_n = 10,
  samples = NULL,
  cores = NULL,
  verbose = TRUE
)
```

Arguments

x	an STlist with transformed RNA counts
genes	a vector of gene names or 'top'. If 'top' (default), interpolation of the 10 genes (`top_n` default) with highest standard deviation in each ST sample is estimated.
top_n	an integer indicating how many top genes to perform interpolation. Default is 10.
samples	the spatial samples for which interpolations will be performed. If NULL (Default), all samples are interpolated.
cores	integer indicating the number of cores to use during parallelization. If NULL, the function uses half of the available cores at a maximum. The parallelization uses `parallel::mclapply` and works only in Unix systems.
verbose	either logical or an integer (0, 1, or 2) to increase verbosity.

Details

This function takes an STlist and a vector of gene names and generates spatial interpolation of gene expression values via "kriging". If genes='top', then the 10 genes (default) with the highest standard deviation for each ST sample are interpolated. The resulting interpolations can be visualized via the `STplot_interpolation` function

Value

x a STlist including spatial interpolations.

Examples

```
# Using included melanoma example (Thrane et al.)
# Download example data set from spatialGE_Data
thrane_tmp = tempdir()
unlink(thrane_tmp, recursive=TRUE)
dir.create(thrane_tmp)
lk='https://github.com/FridleyLab/spatialGE_Data/raw/refs/heads/main/melanoma_thrane.zip?download='
download.file(lk, destfile=paste0(thrane_tmp, '/', 'melanoma_thrane.zip'), mode='wb')
zip_tmp = list.files(thrane_tmp, pattern='melanoma_thrane.zip$', full.names=TRUE)
unzip(zipfile=zip_tmp, exdir=thrane_tmp)
# Generate the file paths to be passed to the STlist function
count_files <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                          full.names=TRUE, pattern='counts')
coord_files <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                          full.names=TRUE, pattern='mapping')
clin_file <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                        full.names=TRUE, pattern='clinical')
# Create STlist
library('spatialGE')
melanoma <- STlist(rnacounts=count_files[c(1,2)],
                   spotcoords=coord_files[c(1,2)],
                   samples=clin_file) # Only first two samples
melanoma <- transform_data(melanoma)
melanoma <- gene_interpolation(melanoma, genes=c('MLANA', 'COL1A1'), samples='ST_mel1_rep2')
```

```
kp = STplot_interpolation(melanoma, genes=c('MLANA', 'COL1A1'))
ggpubr::ggarrange(plotlist=kp)
```

---

get_gene_meta   *get_gene_meta: Extract gene-level metadata and statistics*

Description

Extracts gene-level metadata and spatial statistics (if already computed)

Usage

```
get_gene_meta(x = NULL, sthet_only = FALSE)
```

Arguments

x   an STlist

sthet_only   logical, return only genes with spatial statistics

Details

This function extracts data from the x@gene_meta slot, optionally subsetting only to those genes
for which spatial statistics (Moran's I or Geary's C, see SThet) have been calculated. The output is
a data frame with data from all samples in the STlist

Value

a data frame with gene-level data

Examples

```
# Using included melanoma example (Thrane et al.)
# Download example data set from spatialGE_Data
thrane_tmp = tempdir()
unlink(thrane_tmp, recursive=TRUE)
dir.create(thrane_tmp)
lk='https://github.com/FridleyLab/spatialGE_Data/raw/refs/heads/main/melanoma_thrane.zip?download='
download.file(lk, destfile=paste0(thrane_tmp, '/', 'melanoma_thrane.zip'), mode='wb')
zip_tmp = list.files(thrane_tmp, pattern='melanoma_thrane.zip$', full.names=TRUE)
unzip(zipfile=zip_tmp, exdir=thrane_tmp)
# Generate the file paths to be passed to the STlist function
count_files <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                          full.names=TRUE, pattern='counts')
coord_files <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                          full.names=TRUE, pattern='mapping')
clin_file <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                        full.names=TRUE, pattern='clinical')
```

```
# Create STlist
library('spatialGE')
melanoma <- STlist(rnacounts=count_files[c(1,2)],
                   spotcoords=coord_files[c(1,2)],
                   samples=clin_file) # Only first two samples
melanoma <- transform_data(melanoma, method='log')
melanoma <- SThet(melanoma, genes=c('MLANA', 'TP53'), method='moran')
get_gene_meta(melanoma, sthet_only=TRUE)
```

---

load_images   *load_images: Place tissue images within STlist*

Description

Loads the images from tissues to the appropriate STlist slot.

Usage

```
load_images(x = NULL, images = NULL)
```

Arguments

x   an STlist

images   a string indicating a folder to load images from

Details

This function looks for .PNG or .JPG files within a folder matching the sample names in an exist-
ing STlist. Then, loads the images to the STlist which can be used for plotting along with other
spatialGE plots.

Value

an STlist with images

---

plot_counts	*plot_counts: Generates plots for the distribution of counts*

Description

Generates density plots, violin plots, and/or boxplots for the distribution of count values

Usage

```
plot_counts(
  x = NULL,
  samples = NULL,
  data_type = "tr",
  plot_type = "density",
  color_pal = "okabeito",
  cvalpha = 0.5,
  distrib_subset = 0.5,
  subset_seed = 12345
)
```

Arguments

x	an STlist
samples	samples to include in the plot. Default (NULL) includes all samples
data_type	one of tr or raw, to plot transformed or raw counts
plot_type	one or several of density, violin, and box, to generate density plots, violin plots, and/or boxplots
color_pal	a string of a color palette from khroma or RColorBrewer, or a vector with colors
cvalpha	the transparency of the density plots
distrib_subset	the proportion of spots/cells to plot. Generating these plots can be time consuming due to the large amount of elements to plot. This argument provides control on how many randomly values to show to speed plotting
subset_seed	related to distrib_subset. Sets the seed number to ensure the same subset of values is selected for plotting

Details

The function allows to visualize the distribution counts across all genes and spots in the STlist. The user can select between density plots, violin plots, or box plots as visualization options. Useful for assessment of the effect of filtering and data transformations and to assess zero-inflation. To plot counts or genes per spot/cell, the function distribution_plots should be used instead.

Value

a list of ggplot objects

Examples

```
# Using included melanoma example (Thrane et al.)
# Download example data set from spatialGE_Data
thrane_tmp = tempdir()
unlink(thrane_tmp, recursive=TRUE)
dir.create(thrane_tmp)
lk='https://github.com/FridleyLab/spatialGE_Data/raw/refs/heads/main/melanoma_thrane.zip?download='
download.file(lk, destfile=paste0(thrane_tmp, '/', 'melanoma_thrane.zip'), mode='wb')
zip_tmp = list.files(thrane_tmp, pattern='melanoma_thrane.zip$', full.names=TRUE)
unzip(zipfile=zip_tmp, exdir=thrane_tmp)
# Generate the file paths to be passed to the STlist function
count_files <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                          full.names=TRUE, pattern='counts')
coord_files <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                          full.names=TRUE, pattern='mapping')
clin_file <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                        full.names=TRUE, pattern='clinical')
# Create STlist
library('spatialGE')
melanoma <- STlist(rnacounts=count_files[c(1,2)],
                   spotcoords=coord_files[c(1,2)],
                   samples=clin_file) # Only first two samples
cp <- plot_counts(melanoma, data_type='raw', plot_type=c('violin', 'box'))
ggpubr::ggarrange(plotlist=cp)
```

---

plot_image	*plot_image: Generate a ggplot object of the tissue image*

Description

Creates ggplot objects of the tissue images when available within the STlist

Usage

```
plot_image(x = NULL, samples = NULL)
```

Arguments

x	an STlist
samples	a vector of numbers indicating the ST samples to plot, or their sample names. If vector of numbers, it follow the order of names(x@counts). If NULL, the function plots all samples

Details

If the STlist contains tissue images in the @misc slot, the plot_image function can be used to generate ggplot objects. These ggplot objects can be plotted next to quilt plots (STplot function) for comparative analysis.

Value

a list of plots

| pseudobulk_dim_plot | *pseudobulk_dim_plot: Plot PCA of pseudobulk samples* |

Description

Generates a PCA plot after computation of "pseudobulk" counts

Usage

```
pseudobulk_dim_plot(
  x = NULL,
  color_pal = "muted",
  plot_meta = NULL,
  dim = "pca",
  pcx = 1,
  pcy = 2,
  ptsize = 5
)
```

Arguments

x	an STlist with pseudobulk PCA results in the @misc slot (generated by pseudobulk_samples)
color_pal	a string of a color palette from khroma or RColorBrewer, or a vector of color names or HEX values. Each color represents a category in the variable specified in plot_meta
plot_meta	a string indicating the name of the variable in the sample metadata to color points in the PCA plot
dim	one of umap or pca. The dimension reduction to plot
pcx	integer indicating the principal component to plot in the x axis
pcy	integer indicating the principal component to plot in the y axis
ptsize	the size of the points in the PCA plot. Passed to the size aesthetic from ggplot2

Details

Generates a Principal Components Analysis plot to help in initial data exploration of differences among samples. The points in the plot represent "pseudobulk" samples. This function follows after usage of pseudobulk_samples.

Value

a ggplot object

Examples

```
# Using included melanoma example (Thrane et al.)
# Download example data set from spatialGE_Data
thrane_tmp = tempdir()
unlink(thrane_tmp, recursive=TRUE)
dir.create(thrane_tmp)
lk='https://github.com/FridleyLab/spatialGE_Data/raw/refs/heads/main/melanoma_thrane.zip?download='
download.file(lk, destfile=paste0(thrane_tmp, '/', 'melanoma_thrane.zip'), mode='wb')
zip_tmp = list.files(thrane_tmp, pattern='melanoma_thrane.zip$', full.names=TRUE)
unzip(zipfile=zip_tmp, exdir=thrane_tmp)
# Generate the file paths to be passed to the STlist function
count_files <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                          full.names=TRUE, pattern='counts')
coord_files <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                          full.names=TRUE, pattern='mapping')
clin_file <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                        full.names=TRUE, pattern='clinical')
# Create STlist
library('spatialGE')
melanoma <- STlist(rnacounts=count_files,
                   spotcoords=coord_files,
                   samples=clin_file, cores=2)
melanoma <- pseudobulk_samples(melanoma)
pseudobulk_dim_plot(melanoma, plot_meta='patient')
```

---

pseudobulk_heatmap   *pseudobulk_heatmap: Heatmap of pseudobulk samples*

Description

Generates a heatmap plot after computation of "pseudobulk" counts

Usage

```
pseudobulk_heatmap(
  x = NULL,
  color_pal = "muted",
  plot_meta = NULL,
  hm_display_genes = 30
)
```

Arguments

x   an STlist with pseudobulk counts in the @misc slot (generated by pseudobulk_samples)

color_pal   a string of a color palette from khroma or RColorBrewer, or a vector of color
                names or HEX values. Each color represents a category in the variable specified
                in plot_meta

plot_meta   a string indicating the name of the variable in the sample metadata to annotate
             heatmap columns

hm_display_genes

             number of genes to display in heatmap, selected based on decreasing order of
             standard deviation across samples

Details

Generates a heatmap of transformed "pseudobulk" counts to help in initial data exploration of differences among samples. Each column in the heatmap represents a "pseudobulk" sample. Rows are genes, with the number of genes displayed controlled by the hm_display_genes argument. This function follows after usage of pseudobulk_samples.

Value

a ggplot object

Examples

```
# Using included melanoma example (Thrane et al.)
# Download example data set from spatialGE_Data
thrane_tmp = tempdir()
unlink(thrane_tmp, recursive=TRUE)
dir.create(thrane_tmp)
lk='https://github.com/FridleyLab/spatialGE_Data/raw/refs/heads/main/melanoma_thrane.zip?download='
download.file(lk, destfile=paste0(thrane_tmp, '/', 'melanoma_thrane.zip'), mode='wb')
zip_tmp = list.files(thrane_tmp, pattern='melanoma_thrane.zip$', full.names=TRUE)
unzip(zipfile=zip_tmp, exdir=thrane_tmp)
# Generate the file paths to be passed to the STlist function
count_files <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                          full.names=TRUE, pattern='counts')
coord_files <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                          full.names=TRUE, pattern='mapping')
clin_file <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                        full.names=TRUE, pattern='clinical')
# Create STlist
library('spatialGE')
melanoma <- STlist(rnacounts=count_files,
                   spotcoords=coord_files,
                   samples=clin_file, cores=2)
melanoma <- pseudobulk_samples(melanoma)
hm <- pseudobulk_heatmap(melanoma, plot_meta='BRAF_status', hm_display_genes=30)
```

---

pseudobulk_samples   *pseudobulk_samples: Aggregates counts into "pseudo bulk" samples*

Description

Aggregates spot/cell counts into "pseudo bulk" samples for data exploration

Usage

```
pseudobulk_samples(x = NULL, max_var_genes = 5000, calc_umap = FALSE)
```

Arguments

x	an STlist.
max_var_genes	number of most variable genes (standard deviation) to use in pseudobulk analysis
calc_umap	logical, whether to calculate UMAP embeddings in addition to PCs

Details

This function takes an STlist and aggregates the spot/cell counts into "pseudo bulk" counts by summing all counts from all cell/spots for each gene. Then performs Principal Component Analysis (PCA) to explore non-spatial sample-to-sample variation

Value

an STlist with appended pseudobulk counts and PCA coordinates

Examples

```
# Using included melanoma example (Thrane et al.)
# Download example data set from spatialGE_Data
thrane_tmp = tempdir()
unlink(thrane_tmp, recursive=TRUE)
dir.create(thrane_tmp)
lk='https://github.com/FridleyLab/spatialGE_Data/raw/refs/heads/main/melanoma_thrane.zip?download='
download.file(lk, destfile=paste0(thrane_tmp, '/', 'melanoma_thrane.zip'), mode='wb')
zip_tmp = list.files(thrane_tmp, pattern='melanoma_thrane.zip$', full.names=TRUE)
unzip(zipfile=zip_tmp, exdir=thrane_tmp)
# Generate the file paths to be passed to the STlist function
count_files <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                          full.names=TRUE, pattern='counts')
coord_files <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                          full.names=TRUE, pattern='mapping')
clin_file <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                        full.names=TRUE, pattern='clinical')
# Create STlist
library('spatialGE')
melanoma <- STlist(rnacounts=count_files,
                   spotcoords=coord_files,
                   samples=clin_file, cores=2)
melanoma <- pseudobulk_samples(melanoma)
pseudobulk_dim_plot(melanoma)
```

show,STlist-method   *show: Prints overview of STList oject.*

Description

Prints overview/summary of STList oject.

Usage

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

Arguments

object   an STList object to show summary from.

Details

This function takes an STList and prints a the number of spatial arrays in that object and other information about the object.

spatial_metadata   *spatial_metadata: Prints the names of the available spot/cell annotations*

Description

returns the names of the annotations in the x@spatial_meta slot.

Usage

```
spatial_metadata(x)
```

Arguments

x   an STList object

Value

a list of character vectors containing the column names of x@spatial_meta

---

STclust   *STclust: Detect clusters of spots/cells*

Description

Perform unsupervised spatially-informed clustering on the spots/cells of a ST sample

Usage

```
STclust(
  x = NULL,
  samples = NULL,
  ws = 0.025,
  dist_metric = "euclidean",
  linkage = "ward.D2",
  ks = "dtc",
  topgenes = 2000,
  deepSplit = FALSE,
  cores = NULL,
  verbose = TRUE
)
```

Arguments

x	an STlist with normalized expression data
samples	a vector with strings or a vector with integers indicating the samples to run STclust
ws	a double (0-1) indicating the weight to be applied to spatial distances. Defaults to 0.025
dist_metric	the distance metric to be used. Defaults to 'euclidean'. Other options are the same as in `wordspace::dist.matrix`
linkage	the linkage method applied to hierarchical clustering. Passed to `hclust` and defaults to 'ward.D'
ks	the range of k values to assess. Defaults to `dtc`, meaning `cutreeDynamic` is applied
topgenes	the number of genes with highest spot-to-spot expression variation. The variance is calculated via `Seurat::FindVariableFeatures`.
deepSplit	a logical or integer (1-4), to be passed to `cutreeDynamic` and control cluster resolution
cores	an integer indicating the number of cores to use in parallelization (Unix only)
verbose	either logical or an integer (0, 1, or 2) to increase verbosity

Details

The function takes an STlist and calculates euclidean distances between cells or spots based on the x,y spatial locations, and the expression of the top variable genes (`Seurat::FindVariableFeatures`). The resulting distances are weighted by applying 1-ws to the gene expression distances and ws to the spatial distances. Hierarchical clustering is performed on the sum of the weighted distance matrices. The `STclust` method allows for identification of tissue niches/domains that are spatially cohesive.

Value

an STlist with cluster assignments

Examples

```
# Using included melanoma example (Thrane et al.)
# Download example data set from spatialGE_Data
thrane_tmp = tempdir()
unlink(thrane_tmp, recursive=TRUE)
dir.create(thrane_tmp)
lk='https://github.com/FridleyLab/spatialGE_Data/raw/refs/heads/main/melanoma_thrane.zip?download='
download.file(lk, destfile=paste0(thrane_tmp, '/', 'melanoma_thrane.zip'), mode='wb')
zip_tmp = list.files(thrane_tmp, pattern='melanoma_thrane.zip$', full.names=TRUE)
unzip(zipfile=zip_tmp, exdir=thrane_tmp)
# Generate the file paths to be passed to the STlist function
count_files <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                          full.names=TRUE, pattern='counts')
coord_files <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                          full.names=TRUE, pattern='mapping')
clin_file <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                          full.names=TRUE, pattern='clinical')
# Create STlist
library('spatialGE')
melanoma <- STlist(rnacounts=count_files[c(1,3)], spotcoords=coord_files[c(1,3)], samples=clin_file)
melanoma <- transform_data(melanoma, method='log')
melanoma <- STclust(melanoma, ws=c(0, 0.025))
STplot(melanoma, ws=0.025, samples='ST_mel1_rep2', ptsize=1)
```

STdiff	*STdiff: Differential gene expression analysis for spatial transcriptomics data*

Description

Tests for differentially expressed genes using linear models with or without spatial covariance structures

Usage

```
STdiff(
  x = NULL,
  samples = NULL,
  annot = NULL,
  w = NULL,
  k = NULL,
  deepSplit = NULL,
  topgenes = 5000,
  pval_thr = 0.05,
  pval_adj = "fdr",
  test_type = "mm",
  sp_topgenes = 0.2,
  clusters = NULL,
  pairwise = FALSE,
  verbose = 1L,
  cores = NULL
)
```

Arguments

x	an STlist
samples	an integer indicating the spatial samples to be included in the DE tests. Numbers follow the order in names(x@counts). Sample names are also allowed. If NULL, performs tests on all samples
annot	a column name in x@spatial_meta containing the groups/clusters to be tested. Required if k and w are empty.
w	the spatial weight used in STclust. Required if annot is empty.
k	the k value used in STclust, or dtc for dynamicTreeCut clusters. Required if annot is empty.
deepSplit	the deepSplit value if used in STclust. Required if k='dtc'.
topgenes	an integer indicating the top variable genes to select from each sample based on variance (default=5000). If NULL, all genes are selected.
pval_thr	cut-off of adjusted p-values to define differentially expressed genes from non-spatial linear models. A proportion of genes (sp_topgenes) under this cut-off will be applied the spatial models. Default=0.05
pval_adj	Method to adjust p-values. Defaults to FDR. Other options as available from p.adjust
test_type	one of mm, t_test, or wilcoxon. Specifies the type of test performed.
sp_topgenes	Proportion of differentially expressed genes from non-spatial linear models (and controlled by pval_thr) to use in differential gene expression analysis with spatial linear models. If 0 (zero), no spatial models are fit. Default=0.2
clusters	cluster name(s) to test DE genes, as opposed to all clusters.

pairwise	whether or not to carry tests on a pairwise manner. The default is `pairwise=F`, meaning that DE genes are tested by comparing each cluster to the rest of the pooled cell/spots.
verbose	either logical or an integer (0, 1, or 2) to increase verbosity
cores	Number of cores to use in parallelization. If `NULL`, the number of cores to use is detected automatically

Details

The method tests for differentially expressed genes between groups of spots/cells (e.g., clusters) in a spatial transcriptomics sample. Specifically, the function tests for genes with significantly higher or lower gene expression in one group of spots/cells with respect to the rest of spots/cells in the sample. The method first runs non-spatial linear models on the genes to detect differentially expressed genes. Then spatial linear models with exponential covariance structure are fit on a subset of genes detected as differentially expressed by the non-linear models (sp_topgenes). If running on clusters detected via STclust, the user can specify the assignments using the same parameters (w, k, deepSplit). Otherwise, the assignments are specified by indicating one of the column names in x@spatial_meta. The function uses spaMM::fitme and is computationally expensive even on HPC environments. To run the STdiff using the non-spatial approach (faster), set sp_topgenes=0.

Value

a list with one data frame per sample with results of differential gene expression analysis

---

STdiff_volcano	*STdiff_volcano: Generates volcano plots from STdiff results*

Description

Generates volcano plots of differential expression results from STdiff

Usage

```
STdiff_volcano(
  x = NULL,
  samples = NULL,
  clusters = NULL,
  pval_thr = 0.05,
  color_pal = NULL
)
```

Arguments

x	the output of `STdiff`
samples	samples to create plots
clusters	names of the clusters to generate comparisons
pval_thr	the p-value threshold to color genes with differential expression
color_pal	the palette to color genes by significance

Details

The function generated volcano plots (p-value vs. log-fold change) for genes tested with STdiff. Colors can be customized to show significance from spatial and non-spatial models

Value

a list of ggplot objects

STenrich	*STenrich*

Description

Test for spatial enrichment of gene expression sets in ST data sets

Usage

```
STenrich(
  x = NULL,
  samples = NULL,
  gene_sets = NULL,
  score_type = "avg",
  reps = 1000,
  annot = NULL,
  domain = NULL,
  num_sds = 1,
  min_units = 20,
  min_genes = 5,
  pval_adj_method = "BH",
  seed = 12345,
  cores = NULL,
  verbose = TRUE
)
```

Arguments

x	an STlist with transformed gene expression
samples	a vector with sample names or indexes to run analysis
gene_sets	a named list of gene sets to test. The names of the list should identify the gene sets to be tested
score_type	Controls how gene set expression is calculated. The options are the average expression among genes in a set ('avg'), or a GSEA score ('gsva'). The default is 'avg'
reps	the number of random samples to be extracted. Default is 1000 replicates
annot	name of the annotation within x@spatial_meta containing the spot/cell categories. Needs to be used in conjunction with domain

domain	the domain to restrict the analysis. Must exist within the spot/cell categories included in the selected annotation (i.e., annot)
num_sds	the number of standard deviations to set the minimum gene set expression threshold. Default is one (1) standard deviation
min_units	Minimum number of spots with high expression of a pathway for that gene set to be considered in the analysis. Defaults to 20 spots or cells
min_genes	the minimum number of genes of a gene set present in the data set for that gene set to be included. Default is 5 genes
pval_adj_method	
	the method for multiple comparison adjustment of p-values. Options are the same as that of p.adjust. Default is 'BH'
seed	the seed number for the selection of random samples. Default is 12345
cores	the number of cores used during parallelization. If NULL (default), the number of cores is defined automatically
verbose	either logical or an integer (0, 1, or 2) to increase verbosity

Details

The function performs a randomization test to assess if the sum of distances between cells/spots with high expression of a gene set is lower than the sum of distances among randomly selected cells/spots. The cells/spots are considered as having high gene set expression if the average expression of genes in a set is higher than the average expression plus num_sds times the standard deviation. Control over the size of regions with high expression is provided by setting the minimum number of cells/spots (min_units). This method is a modification of the method devised by Hunter et al. 2021 (zebrafish melanoma study).

Value

a list of data frames with the results of the test

STgradient	*STgradient: Tests of gene expression spatial gradients*

Description

Calculates Spearman's coefficients to detect genes showing expression spatial gradients

Usage

```
STgradient(
  x = NULL,
  samples = NULL,
  topgenes = 2000,
  annot = NULL,
  ref = NULL,
```

```
    exclude = NULL,
    out_rm = FALSE,
    limit = NULL,
    distsumm = "min",
    min_nb = 3,
    robust = TRUE,
    nb_dist_thr = NULL,
    log_dist = FALSE,
    cores = NULL,
    verbose = TRUE
)
```

Arguments

x	an STlist with transformed gene expression
samples	the samples on which the test should be executed
topgenes	the number of high-variance genes to be tested. These genes are selected in descending order of variance as caclulated using Seurat's vst method
annot	the name of a column in @spatial_meta containing the tissue domain assignments for each spot or cell. These assignments can be generated using the STclust function
ref	one of the tissue domains in the column specified in annot, corresponding to the "reference" cluster or domain. Spearman's correlations will be calculated using spots assigned to domains other than this reference domain (or domains specified in exclude).
exclude	optional, a cluster/domain to exclude from the analysis
out_rm	logical (optional), remove gene expression outliers defined by the interquartile method. This option is only valid when robust=F
limit	limite the analysis to spots/cells with distances to ref shorther than the value specified here. Useful when gradients might occur at smaller scales or when the domain in ref is scattered through the tissue. Caution must be used due to difficult interpretation of imposed limits. It is suggested to run analysis without restricted distances in addition for comparison.
distsumm	the distance summary metric to use in correlations. One of min or avg
min_nb	the minimum number of immediate neighbors a spot or cell has to have in order to be included in the analysis. This parameter seeks to reduce the effect of isolated ref spots on the correlation
robust	logical, whether to use robust regression (MASS and sfsmisc packages)
nb_dist_thr	a numeric vector of length two indicating the tolerance interval to assign spots/cells to neighborhoods. The wider the range of the interval, the more likely distinct neighbors to be considered. If NULL, c(0.75, 1.25) and c(0.25, 3) is assigned for Visium and CosMx respectively.
log_dist	logical, whether to apply the natural logarithm to the spot/cell distances. It applies to all distances a constant (1e-200) to avoid log(0)

cores	the number of cores used during parallelization. If NULL (default), the number of cores is defined automatically
verbose	logical, whether to print text to console

Details

The `STgradient` function fits linear models and calculates Spearman coefficients between the expression of a gene and the minimum or average distance of spots or cells to a reference tissue domain. In other wordsm the `STgradient` function can be used to investigate if a gene is expressed higher in spots/cells closer to a specific reference tissue domain, compared to spots/cells farther from the reference domain (or viceversa as indicated by the Spearman's cofficient).

Value

a list of data frames with the results of the test

SThet	*SThet: Computes global spatial autocorrelation statistics on gene expression*

Description

Computes the global spatial autocorrelation statistics Moran's I and/or Geary's C for a set of genes

Usage

```
SThet(
  x = NULL,
  genes = NULL,
  samples = NULL,
  method = "moran",
  k = NULL,
  overwrite = TRUE,
  cores = NULL,
  verbose = TRUE
)
```

Arguments

x	an STlist
genes	a vector of gene names to compute statistics
samples	the samples to compute statistics
method	The spatial statistic(s) to estimate. It can be set to 'moran', 'geary' or both. Default is 'moran'

k	the number of neighbors to estimate weights. By default NULL, meaning that spatial weights will be estimated from Euclidean distances. If an positive integer is entered, then the faster k nearest-neighbors approach is used. Please keep in mind that estimates are not as accurate as when using the default distance-based method.
overwrite	logical indicating if previous statistics should be overwritten. Default to FALSE (do not overwrite)
cores	integer indicating the number of cores to use during parallelization. If NULL, the function uses half of the available cores at a maximum. The parallelization uses `parallel::mclapply` and works only in Unix systems
verbose	logical, whether to print text to console

Details

The function computes global spatial autocorrelation statistics (Moran's I and/or Geary's C) for the requested genes and samples. Then computation uses the package spdep. The calculated statistics are stored in the STlist, which can be accessed with the `get_gene_meta` function. For visual comparative analysis, the function `compare_SThet` can be used afterwards.

Value

an STlist containing spatial statistics

Examples

```
# Using included melanoma example (Thrane et al.)
# Download example data set from spatialGE_Data
thrane_tmp = tempdir()
unlink(thrane_tmp, recursive=TRUE)
dir.create(thrane_tmp)
lk='https://github.com/FridleyLab/spatialGE_Data/raw/refs/heads/main/melanoma_thrane.zip?download='
download.file(lk, destfile=paste0(thrane_tmp, '/', 'melanoma_thrane.zip'), mode='wb')
zip_tmp = list.files(thrane_tmp, pattern='melanoma_thrane.zip$', full.names=TRUE)
unzip(zipfile=zip_tmp, exdir=thrane_tmp)
# Generate the file paths to be passed to the STlist function
count_files <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                          full.names=TRUE, pattern='counts')
coord_files <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                          full.names=TRUE, pattern='mapping')
clin_file <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                        full.names=TRUE, pattern='clinical')
# Create STlist
library('spatialGE')
melanoma <- STlist(rnacounts=count_files[c(1,2)],
                   spotcoords=coord_files[c(1,2)],
                   samples=clin_file) # Only first two samples
melanoma <- transform_data(melanoma, method='log')
melanoma <- SThet(melanoma, genes=c('MLANA', 'TP53'), method='moran')
get_gene_meta(melanoma, sthet_only=TRUE)
```

STlist   *STlist: Creation of STlist objects for spatial transcriptomics analysis*

Description

Creates an STlist object from one or multiple spatial transcriptomic samples.

Usage

```
STlist(
  rnacounts = NULL,
  spotcoords = NULL,
  samples = NULL,
  cores = NULL,
  verbose = TRUE
)
```

Arguments

rnacounts   the count data which can be provided in one of these formats:

- File paths to comma- or tab-delimited files containing raw gene counts, one file for each spatial sample. The first column contains gene names and subsequent columns contain the expression data for each cell/spot. Duplicate gene names will be modified using make.unique. Requires spotcoords and samples.

- File paths to Visium output directories (one per spatial sample). The directory must follow the structure resulting from spaceranger count. The directory contains the .h5 and spatial sub-directory. If no .h5 file is available, sparse matrices (MEX) from spaceranger count. In that case a second sub-directory called filtered_feature_bc_matrix should contain contain the barcodes.tsv.gz, features.tsv.gz, and matrix.mtx.gz files. The spatial sub-directory minimally contains the coordinates (tissue_positions_list.csv), and optionally the high resolution PNG image and accompanying scaling factors (scalefactors_json.json). Requires samples. #'

- File paths to Xenium output directories (one per spatial sample). The directory must follow the structure resulting from the xeniumranger pipeline. The directory contains the .h5 or sparse matrices (MEX). In that case a second sub-directory called cell_feature_matrix should contain contain the barcodes.tsv.gz, features.tsv.gz, and matrix.mtx.gz files. The coordinates must be available in the cells.parquet. Requires samples.

- The exprMat file for each slide of a CosMx-SMI output. The file must contain the "fov" and "cell_ID" columns. The STlist function will separate data from each FOV, since analysis in spatialGE is conducted at the FOV level. Requires samples and spotcoords.

- Seurat object (V4). A Seurat V4 object produced via Seurat::Load10X_Spatial. Multiple samples are allowed as long as they are stored as "slices" in the

Seurat object. Does not require `samples` as sample names are taken from `names(seurat_obj@images)`

- A named list of data frames with raw gene counts (one data frame per spatial sample). Requires `spotcoords`. Argument `samples` only needed when a file path to sample metadata is provided.

spotcoords   the cell/spot coordinates. Not required if inputs are Visium or Xenium (spaceranger or xeniumranger outputs).

- File paths to comma- or tab-delimited files containing cell/spot coordinates, one for each spatial sample. The files must contain three columns: cell/spot IDs, Y positions, and X positions. The cell/spot IDs must match the column names for each cells/spots (columns) in the gene count files. Requires `samples` and `rnacounts`.
- The `metadata` file for each slide of a CosMx-SMI output. The file must contain the "fov", "cell_ID", "CenterX_local_px", and "CenterY_local_px" columns. The `STlist` function will separate data from each FOV, since analysis in spatialGE is conducted at the FOV level. Requires `samples` and `rnacounts`.
- A named list of data frames with cell/spot coordinates. The list names must match list names of the gene counts list

samples   the sample names/IDs and (optionally) metadata associated with each spatial sample. The following options are available for `samples`:

- A vector with sample names, which will be used to match gene the counts and cell/spot coordinates file paths. A sample name must not match file paths for two different samples. For example, instead of using "tissue1" and "tissue12", use "tissue01" and "tissue12".
- A path to a file containing a table with metadata. This file is a comma- or tab-separated table with one sample per row and sample names/IDs in the first column. Subsequent columns may contain variables associated with each spatial sample

cores   integer indicating the number of cores to use during parallelization. If NULL, the function uses half of the available cores at a maximum. The parallelization uses `parallel::mclapply` and works only in Unix systems

verbose   logical, whether to print text to console

Details

Objects of the S4 class STlist are the starting point of analyses in spatialGE. The STlist contains data from one or multiple samples (i.e., tissue slices), and results from most spatialGE's functions are stored within the object.

- Raw gene counts and spatial coordinates. Gene count data have genes in rows and sampling units (e.g., cells, spots) in columns. Spatial coordinates have sampling units in rows and three columns: sample unit IDs, Y position, and X position.
- Visium outputs from *Space Ranger*. The Visium directory must have the directory structure resulting from `spaceranger count`, with either a count matrix represented in MEX files or a h5 file. The directory should also contain a `spatial` sub-directory, with the spatial coordinates

(tissue_positions_list.csv), and optionally the high resolution tissue image and scaling factor file scalefactors_json.json.

- Xenium outputs from *Xenium Ranger*. The Xenium directory must have the directory structure resulting from the xeniumranger pipeline, with either a cell-feature matrix represented in MEX files or a h5 file. The directory should also contain a parquet file, with the spatial coordinates (cells.parquet).

- CosMx-SMI outputs. Two files are required to process SMI outputs: The exprMat and metadata files. Both files must contain the "fov" and "cell_ID" columns. In addition, the metadata files must contain the "CenterX_local_px" and "CenterY_local_px" columns.

- Seurat object (V4). A Seurat V4 object produced via Seurat::Load10X_Spatial.

Optionally, the user can input a path to a file containing a table of sample-level metadata (e.g., clinical outcomes, tissue type, age). This sample metadata file should contain sample IDs in the first column partially matching the file names of the count/coordinate file paths or Visium directories. *Note:* The sample ID of a given sample cannot be a substring of the sample ID of another sample. For example, instead of using "tissue1" and "tissue12", use "tissue01" and "tissue12".

The function uses parallelization if run in a Unix system. Windows users will experience longer times depending on the number of samples.

Value

an STlist object containing the counts and coordinates, and optionally the sample metadata, which can be used for downstream analysis with spatialGE

Examples

```
# Using included melanoma example (Thrane et al.)
# Download example data set from spatialGE_Data
thrane_tmp = tempdir()
unlink(thrane_tmp, recursive=TRUE)
dir.create(thrane_tmp)
lk='https://github.com/FridleyLab/spatialGE_Data/raw/refs/heads/main/melanoma_thrane.zip?download='
download.file(lk, destfile=paste0(thrane_tmp, '/', 'melanoma_thrane.zip'), mode='wb')
zip_tmp = list.files(thrane_tmp, pattern='melanoma_thrane.zip$', full.names=TRUE)
unzip(zipfile=zip_tmp, exdir=thrane_tmp)
# Generate the file paths to be passed to the STlist function
count_files <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                          full.names=TRUE, pattern='counts')
coord_files <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                          full.names=TRUE, pattern='mapping')
clin_file <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                        full.names=TRUE, pattern='clinical')
# Create STlist
library('spatialGE')
melanoma <- STlist(rnacounts=count_files[c(1,3)],
                   spotcoords=coord_files[c(1,3)],
                   samples=clin_file) # Only first three samples
melanoma
```

STlist-class   *Definition of an STlist object class.*

Description

Definition of an STlist object class.

Slots

counts per spot RNA counts

spatial_meta per spot x,y coordinates

gene_meta per gene statistics (e.g., average expression, variance, Moran's I)

sample_meta dataframe with metadata per sample

tr_counts transfromed per spot counts

gene_krige results from kriging on gene expression

misc Parameters and images from ST data

STplot   *STplot: Plots of gene expression, cluster memberships, and metadata*
                                in spatial context

Description

Generates a plot of the location of spots/cells within an spatial sample, and colors them according
to gene expression levels or spot/cell-level metadata

Usage

```
STplot(
  x,
  samples = NULL,
  genes = NULL,
  plot_meta = NULL,
  ks = "dtc",
  ws = NULL,
  deepSplit = NULL,
  color_pal = NULL,
  data_type = "tr",
  ptsize = NULL,
  txsize = NULL
)
```

Arguments

x	an STlist
samples	a vector of numbers indicating the ST samples to plot, or their sample names. If vector of numbers, it follow the order of samples in names(x@counts). If NULL, the function plots all samples
genes	a vector of gene names or a named list of gene sets. In the latter case, the averaged expression of genes within the sets is plotted
plot_meta	a column name in x@spatial_meta to plot
ks	the k values to plot or 'dtc' to plot results from dynamicTreeCut clustering solutions. Requires previous analysis with STclust
ws	the spatial weights to plot samples if STclust was used
deepSplit	a logical or positive number indicating the deepSplit, if samples were analyzed with STclust
color_pal	a string of a color palette from khroma or RColorBrewer, or a vector with enough color names or HEX values
data_type	one of 'tr' or 'raw', to plot transformed or raw counts respectively
ptsize	a number specifying the size of the points. Passed to the size
txsize	a number controlling the size of the text in the plot title and legend title. Passed to the element_text aesthetic.

Details

The function takes an STlist and plots the cells or spots in their spatial context. The users can color the spots/cells according to the expression of selected genes, cluster memberships, or any spot/cell level metadata included in x@spatial_meta. The function also can average expression of gene sets.

Value

a list of plots

Examples

```
# Using included melanoma example (Thrane et al.)
# Download example data set from spatialGE_Data
thrane_tmp = tempdir()
unlink(thrane_tmp, recursive=TRUE)
dir.create(thrane_tmp)
lk='https://github.com/FridleyLab/spatialGE_Data/raw/refs/heads/main/melanoma_thrane.zip?download='
download.file(lk, destfile=paste0(thrane_tmp, '/', 'melanoma_thrane.zip'), mode='wb')
zip_tmp = list.files(thrane_tmp, pattern='melanoma_thrane.zip$', full.names=TRUE)
unzip(zipfile=zip_tmp, exdir=thrane_tmp)
# Generate the file paths to be passed to the STlist function
count_files <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                          full.names=TRUE, pattern='counts')
coord_files <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                          full.names=TRUE, pattern='mapping')
clin_file <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
```

```
                              full.names=TRUE, pattern='clinical')
# Create STlist
library('spatialGE')
melanoma <- STlist(rnacounts=count_files[c(1,2)],
                   spotcoords=coord_files[c(1,2)],
                   samples=clin_file) # Only first two samples
melanoma <- transform_data(melanoma)
STplot(melanoma, gene='MLANA', samples='ST_mel1_rep2', ptsize=1)
```

STplot_interpolation   *STplot_interpolation: Visualize gene expression surfaces*

Description

Produces a gene expression surface from kriging interpolation of ST data.

Usage

```
STplot_interpolation(
  x = NULL,
  genes = NULL,
  top_n = 10,
  samples = NULL,
  color_pal = "BuRd"
)
```

Arguments

x	an STlist containing results from `gene_krige` for the genes selected.
genes	a vector of gene names (one or several) to plot. If 'top', the 10 genes with highest standard deviation from each spatial sample are plotted.
top_n	an integer indicating how many top genes to perform kriging. Default is 10.
samples	a vector indicating the spatial samples to plot. If vector of numbers, it follows the order of `names(x@counts)`. If NULL, the function plots all samples
color_pal	a color scheme from `khroma` or `RColorBrewer`.

Details

This function produces a gene expression surface plot via kriging for one or several genes and spatial samples

Value

a list of plots

Examples

```
# Using included melanoma example (Thrane et al.)
# Download example data set from spatialGE_Data
thrane_tmp = tempdir()
unlink(thrane_tmp, recursive=TRUE)
dir.create(thrane_tmp)
lk='https://github.com/FridleyLab/spatialGE_Data/raw/refs/heads/main/melanoma_thrane.zip?download='
download.file(lk, destfile=paste0(thrane_tmp, '/', 'melanoma_thrane.zip'), mode='wb')
zip_tmp = list.files(thrane_tmp, pattern='melanoma_thrane.zip$', full.names=TRUE)
unzip(zipfile=zip_tmp, exdir=thrane_tmp)
# Generate the file paths to be passed to the STlist function
count_files <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                          full.names=TRUE, pattern='counts')
coord_files <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                          full.names=TRUE, pattern='mapping')
clin_file <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                        full.names=TRUE, pattern='clinical')
# Create STlist
library('spatialGE')
melanoma <- STlist(rnacounts=count_files[c(1,2)],
                   spotcoords=coord_files[c(1,2)],
                   samples=clin_file) # Only first two samples
melanoma <- transform_data(melanoma)
melanoma <- gene_interpolation(melanoma, genes=c('MLANA', 'COL1A1'), samples='ST_mel1_rep2')
kp = STplot_interpolation(melanoma, genes=c('MLANA', 'COL1A1'), samples='ST_mel1_rep2')
ggpubr::ggarrange(plotlist=kp)
```

---

summarize_STlist	*summarize_STlist: Generates a data frame with summary statistics*

Description

Produces a data frame with counts per gene and counts per ROI/spot/cell

Usage

```
summarize_STlist(x = NULL)
```

Arguments

x   an STlist

Details

The function creates a table with counts per gene and counts per region of interest (ROI), spot, or cell in the samples stored in the STlist

Value

a data frame

Examples

```
# Using included melanoma example (Thrane et al.)
# Download example data set from spatialGE_Data
thrane_tmp = tempdir()
unlink(thrane_tmp, recursive=TRUE)
dir.create(thrane_tmp)
lk='https://github.com/FridleyLab/spatialGE_Data/raw/refs/heads/main/melanoma_thrane.zip?download='
download.file(lk, destfile=paste0(thrane_tmp, '/', 'melanoma_thrane.zip'), mode='wb')
zip_tmp = list.files(thrane_tmp, pattern='melanoma_thrane.zip$', full.names=TRUE)
unzip(zipfile=zip_tmp, exdir=thrane_tmp)
# Generate the file paths to be passed to the STlist function
count_files <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                          full.names=TRUE, pattern='counts')
coord_files <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                          full.names=TRUE, pattern='mapping')
clin_file <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                        full.names=TRUE, pattern='clinical')
# Create STlist
library('spatialGE')
melanoma <- STlist(rnacounts=count_files[c(1,2)],
                   spotcoords=coord_files[c(1,2)],
                   samples=clin_file) # Only first two samples
summarize_STlist(melanoma)
```

---

summary,STlist-method   *summary: Prints overview of STList oject.*

Description

Prints overview/summary of STList oject.

Usage

```
## S4 method for signature 'STlist'
summary(object)
```

Arguments

object   an STList object to show summary from.

Details

This function takes an STList and prints a the number of spatial arrays in that object and other information about the object.

| tissue_names | *tissue_names: Prints the names of the tissue samples in the STlist* |

Description

returns a character vector with the names of tissue samples in the STlist.

Usage

```
tissue_names(x)
```

Arguments

x   an STList object

Value

a character vector with the sample names in the STlist object

---

| transform_data | *transform_data: Transformation of spatial transcriptomics data* |

Description

Applies data transformation methods to spatial transcriptomics samples within an STlist

Usage

```
transform_data(
  x = NULL,
  method = "log",
  scale_f = 10000,
  sct_n_regr_genes = 3000,
  sct_min_cells = 5,
  cores = NULL
)
```

Arguments

x	an STlist with raw count matrices.
method	one of `log` or `sct`. If `log`, log-normalization is performed. If `sct`, then the SCTransform method is applied by calling `sctransform::vst`
scale_f	the scale factor used in logarithmic transformation

sct_n_regr_genes

> the number of genes to be used in the regression model during SCTransform. The function sctransform::vst makes a random gene selection based on this number

sct_min_cells The minimum number of spots/cells to be used in the regression model fit by sctransform::vst

cores integer indicating the number of cores to use during parallelization. If NULL, the function uses half of the available cores at a maximum. The parallelization uses parallel::mclapply and works only in Unix systems.

Details

This function takes an STlist with raw counts and performs data transformation. The user has the option to select between log transformation after library size normalization (method='log'), or SCTransform (method='sct'). In the case of logarithmic transformation, a scaling factor (10^4 by default) is applied. The function uses parallelization using "forking" (not available in Windows OS). Note that the method sct returns a matrix with less genes as filtering is done for low expression genes.

Value

x an updated STlist with transformed counts.

Examples

```
# Using included melanoma example (Thrane et al.)
# Download example data set from spatialGE_Data
thrane_tmp = tempdir()
unlink(thrane_tmp, recursive=TRUE)
dir.create(thrane_tmp)
lk='https://github.com/FridleyLab/spatialGE_Data/raw/refs/heads/main/melanoma_thrane.zip?download='
download.file(lk, destfile=paste0(thrane_tmp, '/', 'melanoma_thrane.zip'), mode='wb')
zip_tmp = list.files(thrane_tmp, pattern='melanoma_thrane.zip$', full.names=TRUE)
unzip(zipfile=zip_tmp, exdir=thrane_tmp)
# Generate the file paths to be passed to the STlist function
count_files <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                          full.names=TRUE, pattern='counts')
coord_files <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                          full.names=TRUE, pattern='mapping')
clin_file <- list.files(paste0(thrane_tmp, '/melanoma_thrane'),
                        full.names=TRUE, pattern='clinical')
# Create STlist
library('spatialGE')
melanoma <- STlist(rnacounts=count_files[c(1,2)],
                   spotcoords=coord_files[c(1,2)],
                   samples=clin_file) # Only first two samples
melanoma <- transform_data(melanoma, method='log', cores=2)
```

Index