Package ‘Voyager’

April 4, 2024

Type Package

Title From geospatial to spatial omics

Version 1.4.0

Description SpatialFeatureExperiment (SFE) is a new S4 class for working with spatial single-cell genomics data. The voyager package implements basic exploratory spatial data analysis (ESDA) methods for SFE. Univariate methods include univariate global spatial ESDA methods such as Moran’s I, permutation testing for Moran’s I, and correlograms. Bivariate methods include Lee’s L and cross variogram. Multivariate methods include MULTISPATI PCA and multivariate local Geary’s C recently developed by Anselin. The Voyager package also implements plotting functions to plot SFE data and ESDA results.

Imports BiocParallel, bluster, ggnewscale, ggplot2 (>= 3.4.0), grDevices, grid, lifecycle, Matrix, matrixStats, methods, patchwork, rlang, RSpectra, S4Vectors, scales, scico, sf, SingleCellExperiment, SpatialExperiment, SpatialFeatureExperiment (>= 1.2.1), spdep, stats, SummarizedExperiment, terra, utils

Suggests automap, BiocSingular, BiocStyle, cowplot, ExperimentHub, ggh4x, gstat, hexbin, knitr, pheatmap, rhdf5, rmarkdown, scater, scattermore, scran, sfarrow, SFEData, testthat (>= 3.0.0), vdiffr, vroom

License Artistic-2.0

Encoding UTF-8

RoxygenNote 7.2.3

Config/testthat/edition 3

Depends R (>= 4.2.0)

biocViews GeneExpression, Spatial, Transcriptomics, Visualization

VignetteBuilder knitr

URL https://github.com/pachterlab/voyager

BugReports https://github.com/pachterlab/voyager/issues
Collate 'AllGenerics.R' 'res2df.R' 'SFEMethod-class.R'
'SFEMethod-bivariate.R' 'SFEMethod-multivariate.R'
'SFEMethod-spdep.R' 'bivariate.R' 'featureData.R'
geom_spi.R' 'gstat.R' 'listSFEMethods.R' 'multivariate.R'
'plot-non-spatial.R' 'plot-univar-downstream.R' 'plot.R'
'plotLocalResult.R' 'plotSpatialReducedDim.R' 'spatial-misc.R'
'univariate-downstream.R' 'univariate-internal.R'
'univariate.R' 'utils.R'
git_url https://git.bioconductor.org/packages/Voyager
git_branch RELEASE_3_18
git_last_commit 27aa232
git_last_commit_date 2023-10-24
Repository Bioconductor 3.18
Date/Publication 2024-04-03
Author Lambda Moses [aut, cre] (<https://orcid.org/0000-0002-7092-9427>),
Kayla Jackson [aut] (<https://orcid.org/0000-0001-6483-0108>),
Laura Luebbert [aut] (<https://orcid.org/0000-0003-1379-2927>),
Lior Pachter [aut, rev] (<https://orcid.org/0000-0002-9164-6231>)
Maintainer Lambda Moses <dlu2@caltech.edu>

R topics documented:

- calculateBivariate ........................................ 3
- calculateMultivariate ...................................... 6
- calculateUnivariate ........................................ 9
- clusterCorrelograms ....................................... 18
- clusterMoranPlot .......................................... 19
- clusterVariograms ......................................... 20
- colFeatureData ............................................ 22
- ditto_colors ................................................ 23
- ElbowPlot .................................................... 23
- getDivergeRange .......................................... 24
- getParams .................................................... 25
- listSFEMethods ............................................. 26
- listw2sparse ............................................... 27
- moranBounds ............................................... 27
- moranPlot ................................................... 28
- multispati_rsp ............................................. 30
- multi_listw2sparse ....................................... 32
- plotCellBin2D .............................................. 32
- plotColDataFreqpoly ..................................... 33
- plotColDataHistogram ..................................... 35
- plotColGraph ............................................... 37
- plotCorrelogram .......................................... 39
- plotCrossVariogram ....................................... 41
calculateBivariate

Bivariate spatial statistics

Description

These functions perform bivariate spatial analysis. In this version, the bivariate global method supported are `lee`, `lee.mc`, and `lee.test` from `spdep`, and cross variograms from `gstat` (use `cross_variogram` and `cross_variogram_map` for type argument, see `variogram-internal`). Global Lee statistic is computed by my own implementation that is much faster than that in `spdep`. Bivariate local methods supported are `lee` (use `locallee` for type argument) and `localmoran_bv` a bivariate version of Local Moran in `spdep`.

Usage

```r
# S4 method for signature 'ANY'
calculateBivariate(
  x,
  y = NULL,
  type,
  listw = NULL,
  coords_df = NULL,
  BPPARAM = SerialParam(),
  zero.policy = NULL,
  returnDF = TRUE,
  p.adjust.method = "BH",
  name = NULL,
  ...
)
```

```r
# S4 method for signature 'SpatialFeatureExperiment'
calculateBivariate(
  x,
  type,
  feature1,
  ...
)
```
calculateBivariate()

runBivariate(
    x,
    type,
    feature1,
    feature2 = NULL,
    colGraphName = 1L,
    colGeometryName = 1L,
    sample_id = "all",
    exprs_values = "logcounts",
    BPPARAM = SerialParam(),
    zero.policy = NULL,
    returnDF = TRUE,
    p.adjust.method = "BH",
    swap_rownames = NULL,
    name = NULL,
    ...
)

Arguments

x A numeric matrix whose rows are features/genes, or a numeric vector (then y must be specified), or a SpatialFeatureExperiment (SFE) object with such a matrix in an assay.

y A numeric matrix whose rows are features/genes, or a numeric vector. Bivariate statics will be computed for all pairwise combinations of row names of x and row names of y, except in cross variogram where combinations within x and y are also computed.

type An SFEMethod object, or a string matching the name of an SFEMethod object. The methods mentioned above correspond to SFEMethod objects already implemented in the Voyager package. Use listSFEMethods to see which methods are available. You can implement new SFEMethod objects to apply Voyager functions to other spatial analysis methods. This is in part inspired by the caret, parsnip, and BiocSingular packages.

listw Weighted neighborhood graph as a spdep listw object. Not used when the
method specified in type does not use a spatial neighborhood graph, such as the variogram.

calculatedBivariate

coords_df A sf data frame specifying location of each cell. Not used when the method specified in type uses a spatial neighborhood graph. Must be specified otherwise.

BPPARAM A BiocParallelParam object specifying whether and how computing the metric for numerous genes shall be parallelized.

zero.policy default NULL, use global option value; if TRUE assign zero to the lagged value of zones without neighbours, if FALSE assign NA

returnDF Logical, when the results are not added to a SFE object, whether the results should be formatted as a DataFrame.

p.adjust.method Method to correct for multiple testing, passed to p.adjustSP. Methods allowed are in p.adjust.methods.

name Name to use to store the results, defaults to the name in the SFEMethod object passed to argument type. Can be set to distinguish between results from the same method but with different parameters.

... Other arguments passed to S4 method (for convenience wrappers like calculateMoransI) or method used to compute metrics as specified by the argument type (as in more general functions like calculateUnivariate). See documentation of functions with the same name as specified in type in the spdep package for the method specific arguments. For variograms, see .variogram.

feature1 ID or symbol of the first genes in SFE object, for the argument x. Mandatory if length of feature1 is 1.

feature2 ID or symbol of the second genes in SFE object, for the argument x. Mandatory if length of feature1 is 1.

colGraphName Name of the listw graph in the SFE object that corresponds to entities represented by columns of the gene count matrix. Use colGraphNames to look up names of the available graphs for cells/spots. Note that for multiple sample_ids, it is assumed that all of them have a graph of this same name.

colGeometryName Name of a colGeometry sf data frame whose numeric columns of interest are to be used to compute the metric. Use colGeometryNames to look up names of the sf data frames associated with cells/spots. In the SFE method of calculateUnivariate, this is to specify location of cells for methods that don’t take a spatial neighborhood graph such as the variogram. If the geometry is not of type POINT, then spatialCoords(x) is used instead.

sample_id Sample(s) in the SFE object whose cells/spots to use. Can be "all" to compute metric for all samples; the metric is computed separately for each sample.

exprs_values Integer scalar or string indicating which assay of x contains the expression values.

swap_rownames Column name of rowData(object) to be used to identify features instead of rownames(object) when labeling plot elements. If not found in rowData, then rownames of the gene count matrix will be used.

overwrite Logical, whether to overwrite existing results with the same name. Defaults to FALSE.
calculateMultivariate

DESCRIPTION

These functions perform multivariate spatial data analysis, usually spatially informed dimension reduction.

VALUE

The calculateBivariate function returns a correlation matrix for global Lee, and the results for each pair of genes for other methods. Global results are not stored in the SFE object. Some methods return one result for each pair of genes, while some return pairwise results for more than 2 genes jointly. Local results are stored in the localResults field in the SFE object, with name the concatenation of the two gene names separated by two underscores (___).

EXAMPLES

library(SFEData)
library(scater)
library(scran)
library(SpatialFeatureExperiment)
library(SpatialExperiment)
sfe <- McKellarMuscleData()
sfe <- sfe[,sfe$in_tissue]
sfe <- logNormCounts(sfe)
gs <- modelGeneVar(sfe)
hvgs <- getTopHVGs(gs, fdr.threshold = 0.01)
g <- colGraph(sfe, "visium") <- findVisiumGraph(sfe)

# Matrix method
mat <- logcounts(sfe)[hvgs[1:5],]
df <- df2sf(spatialCoords(sfe), spatialCoordsNames(sfe))
out <- calculateBivariate(mat, type = "lee", listw = g)
out <- calculateBivariate(mat, type = "cross_v Variogram", coords_df = df)

# SFE method
out <- calculateBivariate(sfe, type = "lee",
 feature1 = c("Myh1", "Myh2", "Csrp3"), swap_rownames = "symbol")
out2 <- calculateBivariate(sfe, type = "lee.test", feature1 = "Myh1",
 feature2 = "Myh2", swap_rownames = "symbol")
sfe <- runBivariate(sfe, type = "locallee", feature1 = "Myh1",
 feature2 = "Myh2", swap_rownames = "symbol")

---

calculateMultivariate  Multivariate spatial data analysis

Usage

# S4 method for signature 'ANY,SFEMethod'
calculateMultivariate(
x,
  type,
calculateMultivariate

```r
listw = NULL,
transposed = FALSE,
zero.policy = TRUE,
p.adjust.method = "BH",
...
)

## S4 method for signature 'ANY,character'
calculateMultivariate(x, type, listw = NULL, transposed = FALSE, ...)

## S4 method for signature 'SpatialFeatureExperiment,ANY'
calculateMultivariate(
  x,
  type,
  colGraphName = 1L,
  subset_row = NULL,
  exprs_values = "logcounts",
  sample_action = c("joint", "separate"),
  BPPARAM = SerialParam(),
  ...
)

runMultivariate(
  x,
  type,
  colGraphName = 1L,
  subset_row = NULL,
  exprs_values = "logcounts",
  sample_action = c("joint", "separate"),
  BPPARAM = SerialParam(),
  name = NULL,
  dest = c("reducedDim", "colData"),
  ...
)
```

Arguments

x A numeric matrix whose rows are features/genes, or a SpatialFeatureExperiment (SFE) object with such a matrix in an assay.

type An SFEMethod object, or a string matching the name of an SFEMethod object. The methods mentioned above correspond to SFEMethod objects already implemented in the Voyager package. Use listSFEMethods to see which methods are available. You can implement new SFEMethod objects to apply Voyager functions to other spatial analysis methods. This is in part inspired by the caret, parsnip, and BiocSingular packages.

listw Weighted neighborhood graph as a spdep listw object. Not used when the method specified in type does not use a spatial neighborhood graph, such as the variogram.
transposed Logical, whether the matrix has genes in columns and cells in rows.

zero.policy default NULL, use global option value; if TRUE assign zero to the lagged value of zones without neighbours, if FALSE assign NA

p.adjust.method Method to correct for multiple testing, passed to \texttt{p.adjustSP}. Methods allowed are in \texttt{p.adjust.methods}.

... Extra arguments passed to the specific multivariate method. For example, see \texttt{multispati_rsp} for arguments for MULTISPATI PCA. See \texttt{localC} for arguments for "localC_multi" and "localC_perm_multi".

colGraphName Name of the listw graph in the SFE object that corresponds to entities represented by columns of the gene count matrix. Use \texttt{colGraphNames} to look up names of the available graphs for cells/spots. Note that for multiple sample\_ids, it is assumed that all of them have a graph of this same name.

subset_row Vector specifying the subset of features to use for dimensionality reduction. This can be a character vector of row names, an integer vector of row indices or a logical vector.

exprs\_values Integer scalar or string indicating which assay of x contains the expression values.

sample_action Character, either "joint" or "separate". Spatial methods depend on the spatial coordinates and/or spatial neighborhood graph, which is why SpatialExperiment uses sample\_id to keep coordinates from different samples separate. Some spatial methods can be sensibly run jointly for multiple samples. In this case, "joint" will run the method jointly for all samples, and "separate" will run the method separately for each sample and concatenate the results.

BPARAM A \texttt{BiocParallelParam} object specifying whether and how computing the metric for numerous genes shall be parallelized. This is to parallelize computation across multiple samples when there are a large number of samples. Be cautious if using an optimized BLAS for matrix operations that supports multithreading.

name Name to use to store the results, defaults to the name in the SFEMethod object passed to argument type. Can be set to distinguish between results from the same method but with different parameters.

dest Character, either "reducedDim" or "colData". If the output of the multivariate method is a matrix or array, as in spatially informed dimension reduction, then the only option is "reducedDim", so the results will be stored in \texttt{reducedDim} of the SFE object. If the output is a vector, as in the multivariate version of \texttt{localC}, then it will be sorted in \texttt{colData}. Data frame output, such as from \texttt{localC_perm}, can be stored in either \texttt{reducedDim} or \texttt{colData}.

\textbf{Details}

For the argument \texttt{type}, this package supports "multispati" for MULTISPATI PCA, "localC\_multi" for a multivariate generalization of Geary’s C, "localC\_perm\_multi" for the multivariate Geary’s C with permutation testing, and "gwpca" for geographically weighted PCA.
calculateUnivariate

Value

In `calculateMultivariate`, a matrix for cell embeddings whose attributes include loadings and
eigenvalues if relevant, ready to be added to the SFE object with `reducedDim` setter. For `run*`, a
`SpatialFeatureExperiment` object with the results added. See Details for where the results are
stored.

References

of Wartenberg’s multivariate spatial correlation. Journal of vegetation science, 19, 45-56.


Examples

```r
# example code
library(SFEData)
library(scater)
library(scran)
sfe <- McKellarMuscleData()
sfe <- logNormCounts(sfe)
gvs <- modelGeneVar(sfe)
hvgs <- getTopHVGS(gvs, fdr.threshold = 0.05)
colGraph(sfe, "visium") <- findVisiumGraph(sfe)
sfe <- runMultivariate(sfe, "multispati", subset_row = hvgs)
```

Description

These functions compute univariate spatial statistics, both global and local, on matrices, data frames,
and SFE objects. For SFE objects, the statistics can be computed for numeric columns of `colData`,
`colGeometries`, and `annotGeometries`, and the results are stored within the SFE object. `calculateMoransI`
and `runMoransI` are convenience wrappers for `calculateUnivariate` and `runUnivariate` respec-
tively.

Usage

```r
## S4 method for signature 'ANY,SFEMethod'
calculateUnivariate(
  x,
  type,
  listw = NULL,
  coords_df = NULL,
  BPPARAM = SerialParam(),
  zero.policy = NULL,
```
calculateUnivariate

```
returnDF = TRUE,
p.adjust.method = "BH",
name = NULL,
...
)

## S4 method for signature 'ANY,character'
calculateUnivariate(
  x,
  type,
  listw = NULL,
  coords_df = NULL,
  BPPARAM = SerialParam(),
  zero.policy = NULL,
  returnDF = TRUE,
  p.adjust.method = "BH",
  name = NULL,
  ...
)

## S4 method for signature 'SpatialFeatureExperiment,ANY'
calculateUnivariate(
  x,
  type,
  features = NULL,
  colGraphName = 1L,
  colGeometryName = 1L,
  sample_id = "all",
  exprs_values = "logcounts",
  BPPARAM = SerialParam(),
  zero.policy = NULL,
  returnDF = TRUE,
  include_self = FALSE,
  p.adjust.method = "BH",
  swap_rownames = NULL,
  name = NULL,
  ...
)

## S4 method for signature 'ANY'
calculateMoransI(
  x,
  ...
  BPPARAM = SerialParam(),
  zero.policy = NULL,
  name = "moran"
)
```
## S4 method for signature 'SpatialFeatureExperiment'
calculateMoransI(
  x,
  features = NULL,
  colGraphName = 1L,
  colGeometryName = 1L,
  sample_id = "all",
  exprs_values = "logcounts",
  BPPARAM = SerialParam(),
  zero.policy = NULL,
  returnDF = TRUE,
  include_self = FALSE,
  p.adjust.method = "BH",
  swap_rownames = NULL,
  name = NULL,
  ...
)

colDataUnivariate(
  x,
  type,
  features,
  colGraphName = 1L,
  sample_id = "all",
  BPPARAM = SerialParam(),
  zero.policy = NULL,
  include_self = FALSE,
  p.adjust.method = "BH",
  name = NULL,
  ...
)

colDataMoransI(
  x,
  features,
  colGraphName = 1L,
  sample_id = "all",
  BPPARAM = SerialParam(),
  zero.policy = NULL,
  include_self = FALSE,
  p.adjust.method = "BH",
  name = NULL,
  ...
)

colGeometryUnivariate(
  x,
  type,
features,  
colGeometryName = 1L,  
colGraphName = 1L,  
sample_id = "all",  
BPPARAM = SerialParam(),  
zero.policy = NULL,  
include_self = FALSE,  
p.adjust.method = "BH",  
name = NULL,
...  
)

colGeometryMoransI(
  x,  
  features,  
colGeometryName = 1L,  
colGraphName = 1L,  
sample_id = "all",  
BPPARAM = SerialParam(),  
zero.policy = NULL,  
include_self = FALSE,  
p.adjust.method = "BH",  
name = NULL,
...  
)

annotGeometryUnivariate(
  x,  
type,  
features,  
annotGeometryName = 1L,  
annotGraphName = 1L,  
sample_id = "all",  
BPPARAM = SerialParam(),  
zero.policy = NULL,  
include_self = FALSE,  
p.adjust.method = "BH",  
name = NULL,
...  
)

annotGeometryMoransI(
  x,  
features,  
annotGeometryName = 1L,  
annotGraphName = 1L,  
sample_id = "all",  
BPPARAM = SerialParam(),  
zero.policy = NULL,  
include_self = FALSE,  
p.adjust.method = "BH",  
name = NULL,  
...  
)
calculateUnivariate

runUnivariate{
  x,
  type,
  features = NULL,
  colGraphName = 1L,
  colGeometryName = 1L,
  sample_id = "all",
  exprs_values = "logcounts",
  BPPARAM = SerialParam(),
  swap_rownames = NULL,
  zero.policy = NULL,
  include_self = FALSE,
  p.adjust.method = "BH",
  name = NULL,
  overwrite = FALSE,
  ...
}

runMoransI{
  x,
  features = NULL,
  colGraphName = 1L,
  colGeometryName = 1L,
  sample_id = "all",
  exprs_values = "logcounts",
  BPPARAM = SerialParam(),
  swap_rownames = NULL,
  zero.policy = NULL,
  include_self = FALSE,
  p.adjust.method = "BH",
  name = NULL,
  ...
}

reducedDimUnivariate{
  x,
  type,
  dimred = 1L,
  components = 1L,
  colGraphName = 1L,
  sample_id = "all",
  ...
}
BPPARAM = SerialParam(),
zero.policy = NULL,
include_self = FALSE,
p.adjust.method = "BH",
name = NULL,
...
)

reducedDimMoransI(
  x,
  dimred = 1L,
  components = 1L,
  colGraphName = 1L,
  sample_id = "all",
  BPPARAM = SerialParam(),
  zero.policy = NULL,
  include_self = FALSE,
p.adjust.method = "BH",
  name = NULL,
  ...
)

Arguments

x               A numeric matrix whose rows are features/fgenes, or a SpatialFeatureExperiment (SFE) object with such a matrix in an assay.
type            An SFEMethod object, or a string matching the name of an SFEMethod object. The methods mentioned above correspond to SFEMethod objects already implemented in the Voyager package. Use listSFEMethods to see which methods are available. You can implement new SFEMethod objects to apply Voyager functions to other spatial analysis methods. This is in part inspired by the caret, parsnip, and BiocSingular packages.
listw           Weighted neighborhood graph as a spdep listw object. Not used when the method specified in type does not use a spatial neighborhood graph, such as the variogram.
coords_df       A sf data frame specifying location of each cell. Not used when the method specified in type uses a spatial neighborhood graph. Must be specified otherwise.
BPPARAM         A BiocParallelParam object specifying whether and how computing the metric for numerous genes shall be parallelized.
zero.policy     default NULL, use global option value; if TRUE assign zero to the lagged value of zones without neighbours, if FALSE assign NA
returnDF        Logical, when the results are not added to a SFE object, whether the results should be formatted as a DataFrame.
p.adjust.method Method to correct for multiple testing, passed to p.adjustSP. Methods allowed are in p.adjust.methods.
name
Name to use to store the results, defaults to the name in the SFEMethod object passed to argument type. Can be set to distinguish between results from the same method but with different parameters.

... Other arguments passed to S4 method (for convenience wrappers like calculateMoransI) or method used to compute metrics as specified by the argument type (as in more general functions like calculateUnivariate). See documentation of functions with the same name as specified in type in the spdep package for the method specific arguments. For variograms, see .variogram.

features Genes (calculate* SFE method and run*) or numeric columns of colData(x) (colData*) or any colGeometry (colGeometry*) or annotGeometry (annotGeometry*) for which the univariate metric is to be computed. Default to NULL. When NULL, then the metric is computed for all genes with the values in the assay specified in the argument exprs_values. This can be parallelized with the argument BPPARAM. For genes, if the row names of the SFE object are Ensembl IDs, then the gene symbol can be used and converted to IDs behind the scene with a column in rowData can be specified in swap_rownames. However, if one symbol matches multiple IDs, a warning will be given and the first match will be used. Internally, the results are always stored by the Ensembl ID rather than symbol.

colGraphName Name of the listw graph in the SFE object that corresponds to entities represented by columns of the gene count matrix. Use colGraphNames to look up names of the available graphs for cells/spots. Note that for multiple sample_ids, it is assumed that all of them have a graph of this same name.

colGeometryName Name of a colGeometry sf data frame whose numeric columns of interest are to be used to compute the metric. Use colGeometryNames to look up names of the sf data frames associated with cells/spots. In the SFE method of calculateUnivariate, this is to specify location of cells for methods that don’t take a spatial neighborhood graph such as the variogram. If the geometry is not of type POINT, then spatialCoords(x) is used instead.

sample_id Sample(s) in the SFE object whose cells/spots to use. Can be "all" to compute metric for all samples; the metric is computed separately for each sample.

exprs_values Integer scalar or string indicating which assay of x contains the expression values.

include_self Logical, whether the spatial neighborhood graph should include edges from each location to itself. This is for Getis-Ord Gi* as in localG and localG_perm, not to be used for any other method.

swap_rownames Column name of rowData(object) to be used to identify features instead of rownames(object) when labeling plot elements. If not found in rowData, then rownames of the gene count matrix will be used.

annotGeometryName Name of a annotGeometry sf data frame whose numeric columns of interest are to be used to compute the metric. Use annotGeometryNames to look up names of the sf data frames associated with annotations.

annotGraphName Name of the listw graph in the SFE object that corresponds to the annotGeometry of interest. Use annotGraphNames to look up names of available annotation graphs.
overwrite Logical, whether to overwrite existing results with the same name. Defaults to FALSE.
dimred Name of a dimension reduction, can be seen in reducedDimNames.
components Numeric vector of which components in the dimension reduction to compute spatial statistics on.

Details

Most univariate methods in the package spdep are supported here. These methods are global, meaning returning one result for all spatial locations in the dataset: moran, geary, moran.mc, geary.mc, moran.test, geary.test, globalG.test, sp.correlogram. The variogram and variogram map from the gstat package are also supported.

The following methods are local, meaning each location has its own results: moran.plot, localmoran, localmoran_perm, localC, localC_perm, localG, localG_perm, LOSH, LOSH.mc, LOSH.cs. The GWmodel::gwss method will be supported soon, but is not supported yet.

Global results for genes are stored in rowData. For colGeometry and annotGeometry, the results are added to an attribute of the data frame called featureData, which is a DataFrame analogous to rowData for the gene count matrix, and can be accessed with the geometryFeatureData function. New column names in featureData would follow the same rules as in rowData. For colData, the results can be accessed with the colFeatureData function.

Local results are stored in the field localResults field of the SFE object, which can be accessed with localResults or localResult. If the results have p-values, then -log10 p and adjusted -log10 p are added. Note that in the multiple testing correction, p.adjustSP is used.

When the results are stored in the SFE object, parameters used to compute the results as well as to construct the spatial neighborhood graph are also added. For localResults, the parameters are added to the metadata field params of the localResults sorted by name, which defaults to the name in the SFEMethod object as specified in the type argument. For global methods, parameters for results for genes are in the metadata of rowData(x), organized by name (metadata(rowData(x))$params[[name]]). For colData, the global method parameters are stored in metadata of colData in the field params (metadata(colData(x))$params[[name]]). For geometries, the global method parameters are in an attribute named "params" of the corresponding sf data frame (attr(df, "params")[[name]]).

Value

In calculateUnivariate, if returnDF = TRUE, then a DataFrame, otherwise a list each element of which is the results for each feature. For run*, a SpatialFeatureExperiment object with the results added. See Details for where the results are stored.

References

Examples

library(SpatialFeatureExperiment)
library(SingleCellExperiment)
library(SFEData)
sfe <- McKellarMuscleData("small")
colGraph(sfe, "visium") <- findVisiumGraph(sfe)
features_use <- rownames(sfe)[1:5]

# Moran's I
moran_results <- calculateMoransI(sfe,
features = features_use,
colGraphName = "visium",
exprs_values = "counts"
)

# This does not advocate for computing Moran's I on raw counts.
# Just an example for function usage.
sfe <- runMoransI(sfe,
features = features_use, colGraphName = "visium",
exprs_values = "counts"
)

# Look at the results
head(rowData(sfe))

# Local Moran's I
sfe <- runUnivariate(sfe,
type = "localmoran", features = features_use,
colGraphName = "visium", exprs_values = "counts"
)
head(localResult(sfe, "localmoran", features_use[1]))

# For colData
sfe <- colDataUnivariate(sfe,
type = "localmoran", features = "nCounts",
colGraphName = "visium"
)
head(localResult(sfe, "localmoran", "nCounts"))

# For annotGeometries
annotGraph(sfe, "myofiber_tri2nb") <-
findSpatialNeighbors(sfe,
type = "myofiber_simplified", MARGIN = 3L,
method = "tri2nb", dist_type = "idw",
zero.policy = TRUE
)
sfe <- annotGeometryUnivariate(sfe,
type = "localG", features = "area",
annotGraphName = "myofiber_tri2nb",
annotGeometryName = "myofiber_simplified",
zero.policy = TRUE
)
Using clusterCorrelograms, you can find clusters of correlogram patterns in length scales of spatial autocorrelation. All the correlograms clustered must be computed with the same method and have the same number of lags. Correlograms are clustered jointly across samples.

### Usage

```r
clusterCorrelograms(
  sfe, 
  features, 
  BLUSPARAM, 
  sample_id = "all", 
  method = "I", 
  colGeometryName = NULL, 
  annotGeometryName = NULL, 
  reducedDimName = NULL, 
  swap_rownames = NULL, 
  name = "sp.correlogram"
)
```

### Arguments

- **sfe**: A `SpatialFeatureExperiment` object with correlograms computed for features of interest.
- **features**: Features whose correlograms to cluster.
- **BLUSPARAM**: A `BlusterParam` object specifying the algorithm to use.
- **sample_id**: Sample(s) in the SFE object whose cells/spots to use. Can be "all" to compute metric for all samples; the metric is computed separately for each sample.
- **method**: "corr" for correlation, "I" for Moran’s I, "C" for Geary’s C
- **colGeometryName**: Name of colGeometry from which to look for features.
- **annotGeometryName**: Name of annotGeometry from which to look for features.
- **reducedDimName**: Name of a dimension reduction, can be seen in `reducedDimNames`. `colGeometryName` and `annotGeometryName` have precedence over `reducedDimName`.
- **swap_rownames**: Column name of `rowData(object)` to be used to identify features instead of `rownames(object)` when labeling plot elements. If not found in `rowData`, then `rownames` of the gene count matrix will be used.
- **name**: Name under which the correlogram results are stored, which is by default "sp.correlogram".

### Description

Cluster the correlograms to find patterns in length scales of spatial autocorrelation. All the correlograms clustered must be computed with the same method and have the same number of lags. Correlograms are clustered jointly across samples.
clusterMoranPlot

Value

A data frame with 3 columns: feature for the features, cluster a factor for cluster membership of the features within each sample, and sample_id for the sample.

Examples

library(SpatialFeatureExperiment)
library(SFEData)
library(bluster)
sfe <- McKellarMuscleData("small")
colGraph(sfe, "visium") <- findVisiumGraph(sfe)
inds <- c(1, 3, 4, 5)
sfe <- runUnivariate(sfe,
    type = "sp.correlogram",
    features = rownames(sfe)[inds],
    exprs_values = "counts", order = 5
)
clust <- clusterCorrelograms(sfe,
    features = rownames(sfe)[inds],
    BLUSPARAM = KmeansParam(2)
)

Description

The Moran plot plots the value at each location on the x axis, and the average of the neighbors of each locations on the y axis. Sometimes clusters can be seen on the Moran plot, indicating different types of neighborhoods.

Usage

clusterMoranPlot(
    sfe,
    features,
    BLUSPARAM,
    sample_id = "all",
    colGeometryName = NULL,
    annotGeometryName = NULL,
    swap_rownames = NULL
)

Arguments

sfe A SpatialFeatureExperiment object with Moran plot computed for the feature of interest. If the Moran plot for that feature has not been computed for that feature in this sample_id, it will be calculated and stored in rowData. See calculateUnivariate.
features        Features whose Moran plot are to be cluster. Features whose Moran plots have not been computed will be skipped, with a warning.

BLUSPARAM      A BlusterParam object specifying the algorithm to use.

sample_id      Sample(s) in the SFE object whose cells/spots to use. Can be "all" to compute metric for all samples; the metric is computed separately for each sample.

colGeometryName Name of colGeometry from which to look for features.

annotGeometryName Name of annotGeometry from which to look for features.

swap_rownames  Column name of rowData(object) to be used to identify features instead of rownames(object) when labeling plot elements. If not found in rowData, then rownames of the gene count matrix will be used.

Value
A data frame each column of which is a factor for cluster membership of each feature. The column names are the features.

Examples

```r
library(SpatialFeatureExperiment)
library(SingleCellExperiment)
library(SFEData)
library(bluster)
sfe <- McKellarMuscleData("small")
colGraph(sfe, "visium") <- findVisiumGraph(sfe)
# Compute moran plot
sfe <- runUnivariate(sfe,
    type = "moran.plot", features = rownames(sfe)[1],
    exprs_values = "counts"
)
clusts <- clusterMoranPlot(sfe, rownames(sfe)[1],
    BLUSPARAM = KmeansParam(2)
)
```

clusterVariograms  Cluster variograms of multiple features

Description
This function clusters variograms of features across samples to find patterns in decays in spatial autocorrelation. The fitted variograms are clustered as different samples can have different distance bins.
Usage

```r
clusterVariograms(
  sfe,  
  features, 
  BLUSPARAM, 
  n = 20, 
  sample_id = "all", 
  colGeometryName = NULL, 
  annotGeometryName = NULL, 
  reducedDimName = NULL, 
  swap_rownames = NULL, 
  name = "variogram"
)
```

Arguments

- **sfe**: A SpatialFeatureExperiment object with correlograms computed for features of interest.
- **features**: Features whose correlograms to cluster.
- **BLUSPARAM**: A BlusterParam object specifying the algorithm to use.
- **n**: Number of points on the fitted variogram line.
- **sample_id**: Sample(s) in the SFE object whose cells/spots to use. Can be "all" to compute metric for all samples; the metric is computed separately for each sample.
- **colGeometryName**: Name of colGeometry from which to look for features.
- **annotGeometryName**: Name of annotGeometry from which to look for features. If not found in colGeometryName and annotGeometryName have precedence over reducedDimName.
- **reducedDimName**: Name of a dimension reduction, can be seen in `reducedDimNames`. colGeometryName and annotGeometryName have precedence over reducedDimName.
- **swap_rownames**: Column name of `rowData(object)` to be used to identify features instead of `rownames(object)` when labeling plot elements. If not found in rowData, then rownames of the gene count matrix will be used.
- **name**: Name under which the correlogram results are stored, which is by default "sp.correlogram".

Value

A data frame with 3 columns: `feature` for the features, `cluster` a factor for cluster membership of the features within each sample, and `sample_id` for the sample.

Examples

```r
library(SFEData)
library(scater)
library(bluster)
library(Matrix)
sfe <- McKellarMuscleData()
```
sfe <- logNormCounts(sfe)
# Just the highly expressed genes
gs <- order(Matrix::rowSums(counts(sfe)), decreasing = TRUE)[1:10]
genere <- rownames(sfe)[gs]
sfe <- runUnivariate(sfe, "variogram", features = genes)
clusts <- clusterVariograms(sfe, genes, BLUSPARAM = HclustParam(),
   swap_rownames = "symbol")

# Plot the clustering
plotVariogram(sfe, genes, color_by = clusts, group = "feature",
   use_lty = FALSE, swap_rownames = "symbol", show_np = FALSE)

---

colFeatureData

**Get metadata of colData, rowData, and geometries**

**Description**
Results of spatial analyses on columns in colData, rowData, and geometries are stored in their metadata, which can be accessed by the metadata function. The colFeatureData function allows the users to more directly access these results.

**Usage**
colFeatureData(sfe)
rowFeatureData(sfe)
geometryFeatureData(sfe, type, MARGIN = 2L)
reducedDimFeatureData(sfe, dimred)

**Arguments**
sfe
  An SFE object.
type
  Which geometry, can be name (character) or index (integer)
MARGIN
dimred
  Name of a dimension reduction, can be seen in reducedDimNames.

**Value**
A DataFrame.

**See Also**
getParams
### Examples

```r
library(SpatialFeatureExperiment)
library(SingleCellExperiment)
library(SFEData)
sfe <- McKellarMuscleData("small")
colGraph(sfe, "visium") <- findVisiumGraph(sfe)
# Moran's I for colData
sfe <- colDataMoransI(sfe, "nCounts")
colFeatureData(sfe)
```

---

### ditto_colors

**Colorblind friendly palette from dittoSeq**

#### Description

Just to get the palette without having to install all those dependencies of dittoSeq.

#### Usage

```r
ditto_colors
```

#### Format

A character vector of hex colors of the palette. There are 40 colors.

#### Source

The dittoSeq package.

---

### ElbowPlot

**Plot the elbow plot or scree plot for PCA**

#### Description

Apparently, there is no apparent way to plot the PC elbow plot other than extracting the variance explained attribute of the dimred slot, because even the OSCA book makes the elbow plot this way, which I find kind of cumbersome compared to Seurat. So I’m writing this function to make the elbow plot with SCE less cumbersome.
Usage

ElbowPlot(
  sce,
  ndims = 20,
  nfnega = 0,
  reduction = "PCA",
  sample_id = "all",
  facet = FALSE,
  ncol = NULL
)

Arguments

sce
  A SingleCellExperiment object, or anything that inherits from SingleCellExperiment.

ndims
  Number of components with positive eigenvalues, such as PCs in non-spatial PCA.

nfnega
  Number of nega eigenvalues and their eigenvectors to compute. These indicate negative spatial autocorrelation.

reduction
  Name of the dimension reduction to use. It must have an attribute called either "percentVar" or "eig" for eigenvalues. Defaults to "PCA".

tsample_id
  Sample(s) in the SFE object whose cells/spots to use. Can be "all" to compute metric for all samples; the metric is computed separately for each sample.

facet
  Logical, whether to facet by samples when multiple samples are present. This is relevant when spatial PCA is run separately for each sample, which gives different results from running jointly for all samples.

ncol
  Number of columns of facets if facetting.

Value

A ggplot object. The y axis is eigenvalues or percentage of variance explained if relevant.

Examples

library(SFEData)
library(scater)
sfe <- McKellarMuscleData("small")
sfe <- runPCA(sfe, ncomponents = 10, exprs_values = "counts")
ElbowPlot(sfe, ndims = 10)

getDivergeRange

Get beginning and end of palette to center a divergent palette

Description

This function is no longer used internally as it’s unnecessary for scico divergent palettes. But it can be useful when using divergent palettes outside scico where one must specify beginning and end but not midpoint, to override the default palette.
**getParams**

Usage

getDivergeRange(values, diverge_center = 0)

Arguments

values Numeric vector to be colored.
diverge_center Value to center on, defaults to 0.

Value

A numeric vector of length 2, the first element is for beginning, and the second for end. The values are between 0 and 1.

Examples

v <- rnorm(10)
getDivergeRange(v, diverge_center = 0)

---

**getParams**

Get parameters used in spatial methods

Description

The getParams function allows users to access the parameters used to compute the results that may be stored in colFeatureData.

Usage

getParams(
  sfe,
  name,
  local = FALSE,
  colData = FALSE,
  colGeometryName = NULL,
  annotGeometryName = NULL,
  reducedDimName = NULL
)

Arguments

sfe A SpatialFeatureExperiment object.
name Name used to store the results.
local Logical, whether the results of interest come from a local spatial method.
colData Logical, whether the results were computed for a column of colData(sfe).
colGeometryName To get results for a colGeometry.
listSFEMethods

annotGeometryName
To get results for an annotGeometry; colGeometry has precedence so this argument is ignored if colGeometryName is specified.

reducedDimName
Name of a dimension reduction, can be seen in reducedDimNames. colGeometryName and annotGeometryName have precedence over reducedDimName.

Value
A named list showing the parameters

Examples
library(SFEData)
library(scater)
sfe <- McKellarMuscleData("small")
colGraph(sfe, "visium") <- findVisiumGraph(sfe)
sfe <- colDataMoransI(sfe, "nCounts")
getParams(sfe, "moran", colData = TRUE)

listSFEMethods

List all spatial methods in Voyager package

Description
This package ships with many spatial statistics methods as SFEMethod objects. The user can adapt the uniform user interface of this package to other spatial methods by creating new SFEMethod objects. This function lists the names of all methods within Voyager, to use for the type argument in calculateUnivariate, calculateBivariate, and calculateMultivariate.

Usage
listSFEMethods(variate = c("uni", "bi", "multi"), scope = c("global", "local"))

Arguments
variate
Uni-, bi-, or multi-variate.
scope
whether it’s local or global.

Value
A data frame with a column for the name and another for a brief description.

Examples
listSFEMethods("uni", "local")
listw2sparse

Convert listw into sparse adjacency matrix

Description

Edge weights are used in the adjacency matrix. Because most elements of the matrix are 0, using sparse matrix greatly reduces memory use.

Usage

listw2sparse(listw)

Arguments

listw A listw object for spatial neighborhood graph.

Value

A sparse dgCMatrix, whose row represents each cell or spot and whose columns represent the neighbors. The matrix does not have to be symmetric. If region.id is present in the listw object, then it will be the row and column names of the output matrix.

Examples

library(SFEData)
sfe <- McKellarMuscleData("small")
g <- findVisiumGraph(sfe)
mat <- listw2sparse(g)

moranBounds

Compute the bounds of Moran’s I given spatial neighborhood graph

Description

Values Moran’s I can take depends on the spatial neighborhood graph. The bounds of Moran’s I given the graph, C, are given by the minimum and maximum eigenvalues of the double centered – i.e. subtracting column means and row means – adjacency matrix \((I - \varphi \varphi^T / n)C(I - \varphi \varphi^T / n)\), where \(\varphi\) is a vector of all 1’s. This implementation follows the implementation in ade spatial and uses the RSpectra package to more quickly find only the minimum and maximum eigenvalues without performing unnecessary work to find the full spectrum as done in base R’s eigen.

Usage

moranBounds(listw)
Arguments

listw A listw object for spatial neighborhood graph.

Value

A numeric vector of minimum and maximum Moran’s I given the spatial neighborhood graph.

Note

After double centering, the adjacency matrix is no longer sparse, so this function can take up a lot of memory for larger datasets.

References


Examples

# example code
library(SFEData)
sfe <- McKellarMuscleData("small")
g <- findVisiumGraph(sfe)
moranBounds(g)

moranPlot Use ggplot to plot the moran.plot results

Description

This function uses ggplot2 to plot the Moran plot. The plot would be more aesthetically pleasing than the base R version implemented in spdep. In addition, contours are plotted to show point density on the plot, and the points can be colored by a variable, such as clusters. The contours may also be filled and only influential points plotted. When filled, the viridis E option is used.

Usage

moranPlot(
  sfe,
  feature,
  graphName = 1L,
  sample_id = "all",
  contour_color = "cyan",
  color_by = NULL,
  colGeometryName = NULL,
  annotGeometryName = NULL,
  plot_singletons = TRUE,
  binned = FALSE,
Arguments

sfe  
A SpatialFeatureExperiment object.

feature  
Name of one variable to show on the plot. It will be converted to sentence case on the x axis and lower case in the y axis appended after "Spatially lagged". One feature at a time since the colors in color_by may be specific to this feature (e.g. from clusterMoranPlot).

graphName  
Name of the colGraph or annotGraph, the spatial neighborhood graph used to compute the Moran plot. This is to determine which points are singletons to plot differently on this plot.

sample_id  
One sample_id for the sample whose graph to plot.

colGeometryName  
Name of a colGeometry sf data frame whose numeric columns of interest are to be used to compute the metric. Use colGeometryNames to look up names of the sf data frames associated with cells/spots.

annotGeometryName  
Name of a annotGeometry of the SFE object, to annotate the gene expression plot.

plot_singletons  
Logical, whether to plot items that don’t have spatial neighbors.

binned  
Logical, whether to plot 2D histograms. This argument has precedence to filled.

filled  
Logical, whether to plot filled contours for the non-influential points and only plot influential points as points.

divergent  
Logical, whether a divergent palette should be used.

diverge_center  
If divergent = TRUE, the center from which the palette should diverge. If NULL, then not centering.
swap_rownames

Column name of rowData(object) to be used to identify features instead of rownames(object) when labeling plot elements. If not found in rowData, then rownames of the gene count matrix will be used.

bins

If binning the colGeometry in space due to large number of cells or spots, the number of bins, passed to geom_bin2d or geom_hex. If NULL (default), then the colGeometry is plotted without binning. If binning, a point geometry is recommended. If the geometry is not point, then the centroids will be used.

binwidth

Width of bins, passed to geom_bin2d or geom_hex.

hex

Logical, whether to use geom_hex. Note that geom_hex is broken in ggplot2 version 3.4.0. Please update ggplot2 if you are getting horizontal stripes when hex = TRUE.

plot_influential

Logical, whether to plot influential points with different palette if binned = TRUE.

bins_contour

Number of bins in the point density contour. Use a smaller number to make sparser contours.

name

Name under which the Moran plot results are stored. By default "moran.plot".

...

Other arguments to pass to geom_density2d.

Value

A ggplot object.

Examples

library(SpatialFeatureExperiment)
library(SingleCellExperiment)
library(SFEData)
library(bluster)
library(scater)
sfe <- McKellarMuscleData("full")
sfe <- sfe[, colData(sfe)$in_tissue]
sfe <- logNormCounts(sfe)
colGraph(sfe, "visium") <- findVisiumGraph(sfe)
sfe <- runUnivariate(sfe, type = "moran.plot", features = "Myh1", swap_rownames = "symbol")
clust <- clusterMoranPlot(sfe, "Myh1", BLUSPARAM = KmeansParam(2), swap_rownames = "symbol")
moranPlot(sfe, "Myh1", graphName = "visium", color_by = clust[, 1], swap_rownames = "symbol")

multispati_rsp

A faster implementation of MULTISPATI PCA
multispati_rsp

Description
This implementation uses the RSpectra package to efficiently compute a small subset of eigenvalues and eigenvectors, as a small subset is typically used. Hence it’s much faster and memory efficient than the original implementation in adespatial. However, this implementation here does not support row and column weighting other than the standard ones for PCA, so the adespatial implementation is more general.

Usage
multispati_rsp(x, listw, nfposi = 30L, nfnega = 30L, scale = TRUE)

Arguments
x A matrix whose columns are features and rows are cells.
listw A listw object, a spatial neighborhood graph for the cells in x. The length must be equal to the number of row of x.

nfposi Number of positive eigenvalues and their eigenvectors to compute.
nfnega Number of nega eigenvalues and their eigenvectors to compute. These indicate negative spatial autocorrelation.
scale Logical, whether to scale the data.

Value
A matrix for the cell embeddings in each spatial PC, with attribute loading for the eigenvectors or gene loadings, and attribute eig for the eigenvalues.

Note
Eigen decomposition will fail if any feature has variance zero leading to NaN in the scaled matrix.

References

Examples
library(SFEData)
library(scater)
sfe <- McKellarMuscleData("small")
sfe <- sfe[,sfe$in_tissue]
sfe <- logNormCounts(sfe)
inds <- order(rowSums(logcounts(sfe)), decreasing = TRUE)[1:50]
mat <- logcounts(sfe)[inds,]
g <- findVisiumGraph(sfe)
out <- multispati_rsp(t(mat), listw = g, nfposi = 10, nfnega = 10)
multi_listw2sparse  

Convert multiple listw graphs into a single sparse adjacency matrix

Description

Each sample in the SFE object has a separate spatial neighborhood graph. Spatial analyses performed jointly on multiple samples require a combined spatial neighborhood graph from the different samples, where the different samples would be disconnected components of the graph. This combined adjacency matrix can be used in MULTISPATI PCA.

Usage

multi_listw2sparse(listws)

Arguments

listws  
A list of listw objects.

Value

A sparse dgCMatrix of the combined spatial neighborhood graph, with the original spatial neighborhood graphs of the samples on the diagonal. When the input is an SFE object, the rows and columns will match the column names of the SFE object.

Examples

# example code

plotCellBin2D  

Plot cell density as 2D histogram

Description

This function plots cell density in histological space as 2D histograms, especially helpful for larger smFISH-based datasets.

Usage

plotCellBin2D(
sfe,
sample_id = "all",
bins = 200,
binwidth = NULL,
hex = FALSE,
ncol = NULL,
bbox = NULL)
)
**plotColDataFreqpoly**

**Arguments**

- `sfe` A SpatialFeatureExperiment object.
- `sample_id` Sample(s) in the SFE object whose cells/spots to use. Can be "all" to compute metric for all samples; the metric is computed separately for each sample.
- `bins` Number of bins. Can be a vector of length 2 to specify for x and y axes separately.
- `binwidth` Width of bins, passed to `geom_bin2d` or `geom_hex`.
- `hex` Logical, whether to use hexagonal bins.
- `ncol` Number of columns if plotting multiple features. Defaults to NULL, which means using the same logic as `facet_wrap`, which is used by patchwork's `wrap_plots` by default.
- `bbox` A bounding box to specify a smaller region to plot, useful when the dataset is large. Can be a named numeric vector with names "xmin", "xmax", "ymin", and "ymax", in any order. If plotting multiple samples, it should be a matrix with sample IDs as column names and "xmin", "ymin", "xmax", and "ymax" as row names. If multiple samples are plotted but bbox is a vector rather than a matrix, then the same bounding box will be used for all samples. You may see points at the edge of the geometries if the intersection between the bounding box and a geometry happens to be a point there. If NULL, then the entire tissue is plotted.

**Value**

A ggplot object.

**Examples**

```r
library(SFEData)
sfe <- HeNSCLCData()
plotCellBin2D(sfe)
```

---

**plotColDataFreqpoly**  *Plot frequency polygons for colData and rowData columns*

**Description**

This function is recommended instead of `plotColDataHistogram` when coloring by multiple categories and log transforming the y axis, which causes problems in stacked histograms.

**Usage**

```r
plotColDataFreqpoly(
  sce,
  feature,
  color_by = NULL,
  subset = NULL,
)```
```r
plotColDataFreqpoly(
  bins = 100,
  binwidth = NULL,
  linewidth = 1.2,
  scales = "free",
  ncol = 1,
  position = "identity"
)

plotRowDataFreqpoly(
  sce,
  feature,
  color_by = NULL,
  subset = NULL,
  bins = 100,
  binwidth = NULL,
  linewidth = 1.2,
  scales = "free",
  ncol = 1,
  position = "identity"
)

Arguments

sce             A SingleCellExperiment object.
feature      Names of columns in colData or rowData to plot. When multiple features are specified, they will be plotted in separate facets.
color_by      Name of a categorical column in colData or rowData to color the polygons.
subset       Name of a logical column to only plot a subset of the data.
bins          Number of bins. Overridden by binwidth. Defaults to 30.
binwidth      The width of the bins. Can be specified as a numeric value or as a function that calculates width from unscaled x. Here, "unscaled x" refers to the original x values in the data, before application of any scale transformation. When specifying a function along with a grouping structure, the function will be called once per group. The default is to use the number of bins in bins, covering the range of the data. You should always override this value, exploring multiple widths to find the best to illustrate the stories in your data.
               The bin width of a date variable is the number of days in each time; the bin width of a time variable is the number of seconds.
linewidth      Line width of the polygons, defaults to a thicker 1.2.
scales         Should scales be fixed ("fixed", the default), free ("free"), or free in one dimension ("free_x", "free_y")?
ncol            Number of columns in the facetting.
position      Position adjustment, either as a string naming the adjustment (e.g. "jitter" to use position_jitter), or the result of a call to a position adjustment function. Use the latter if you need to change the settings of the adjustment.
```
See Also

plotColDataHistogram

Examples

library(SBEData)
sfe <- McKellarMuscleData()
plotColDataFreqpoly(sfe, c("nCounts", "nGenes"), color_by = "in_tissue",
                   bins = 50)
plotColDataFreqpoly(sfe, "nCounts", subset = "in_tissue")
sfe2 <- sfe[, sfe$in_tissue]
plotColDataFreqpoly(sfe2, c("nCounts", "nGenes"), bins = 50)

plotColDataHistogram  Plot histograms for colData and rowData columns

Description

Plot histograms for colData and rowData columns

Usage

plotColDataHistogram(
  sce,
  feature,
  fill_by = NULL,
  facet_by = NULL,
  subset = NULL,
  bins = 100,
  binwidth = NULL,
  scales = "free",
  ncol = 1,
  position = "stack",
  ...
)

plotRowDataHistogram(
  sce,
  feature,
  fill_by = NULL,
  facet_by = NULL,
  subset = NULL,
  bins = 100,
  binwidth = NULL,
  scales = "free",
  ncol = 1,
  position = "stack",
)
Arguments

sce
A SingleCellExperiment object.

feature
Names of columns in colData or rowData to plot. When multiple features are specified, they will be plotted in separate facets.

fill_by
Name of a categorical column in colData or rowData to fill the histogram.

facet_by
Column in colData or rowData to facet with. When multiple features are plotted, the features will be in different facets. In this case, setting facet_by will call facet_grid so the features are in rows and categories in facet_by will be in columns.

subset
Name of a logical column to only plot a subset of the data.

bins
Numeric vector giving number of bins in both vertical and horizontal directions. Set to 100 by default.

binwidth
The width of the bins. Can be specified as a numeric value or as a function that calculates width from unscaled x. Here, "unscaled x" refers to the original x values in the data, before application of any scale transformation. When specifying a function along with a grouping structure, the function will be called once per group. The default is to use the number of bins in bins, covering the range of the data. You should always override this value, exploring multiple widths to find the best to illustrate the stories in your data.

The bin width of a date variable is the number of days in each time; the bin width of a time variable is the number of seconds.

scales
Should scales be fixed ("fixed", the default), free ("free"), or free in one dimension ("free_x", "free_y")?

ncol
Number of columns in the facetting.

position
Position adjustment, either as a string naming the adjustment (e.g. "jitter" to use position_jitter), or the result of a call to a position adjustment function. Use the latter if you need to change the settings of the adjustment.

... Other arguments passed on to layer(). These are often aesthetics, used to set an aesthetic to a fixed value, like colour = "red" or size = 3. They may also be parameters to the paired geom/stat.

Value

A ggplot object

See Also

plotColDataFreqpoly
Examples

library(SFEData)
sfe <- McKellarMuscleData()
plotColDataHistogram(sfe, c("nCounts", "nGenes"), fill_by = "in_tissue",
                     bins = 50, position = "stack")
plotColDataHistogram(sfe, "nCounts", subset = "in_tissue")
sfe2 <- sfe[, sfe$in_tissue]
plotColDataHistogram(sfe2, c("nCounts", "nGenes"), bins = 50)

plotColGraph
Plot spatial graphs

Description
A ggplot version of spdep::plot.nb, reducing boilerplate for SFE objects.

Usage
plotColGraph(
  sfe,
  colGraphName = 1L,
  colGeometryName = 1L,
  sample_id = "all",
  weights = FALSE,
  segment_size = 0.5,
  geometry_size = 0.5,
  ncol = NULL,
  bbox = NULL
)

plotAnnotGraph(
  sfe,
  annotGraphName = 1L,
  annotGeometryName = 1L,
  sample_id = "all",
  weights = FALSE,
  segment_size = 0.5,
  geometry_size = 0.5,
  ncol = NULL,
  bbox = NULL
)

Arguments
sfe A SpatialFeatureExperiment object.
colGraphName Name of graph associated with columns of the gene count matrix to be plotted.
plotColGraph

colGeometryName
Name of a colGeometry sf data frame whose numeric columns of interest are to be used to compute the metric. Use colGeometryNames to look up names of the sf data frames associated with cells/spots.

sample_id
Sample(s) in the SFE object whose cells/spots to use. Can be "all" to compute metric for all samples; the metric is computed separately for each sample.

weights
Whether to plot weights. If TRUE, then transparency (alpha) of the segments will represent edge weights.

segment_size
Thickness of the segments that represent graph edges.

geometry_size
Point size (for POINT geometries) or line thickness (for LINESTRING and POLYGON) to plot the geometry in the background.

ncol
Number of columns if plotting multiple features. Defaults to NULL, which means using the same logic as facet_wrap, which is used by patchwork’s wrap_plots by default.

bbox
A bounding box to specify a smaller region to plot, useful when the dataset is large. Can be a named numeric vector with names "xmin", "xmax", "ymin", and "ymax", in any order. If plotting multiple samples, it should be a matrix with sample IDs as column names and "xmin", "ymin", "xmax", and "ymax" as row names. If multiple samples are plotted but bbox is a vector rather than a matrix, then the same bounding box will be used for all samples. You may see points at the edge of the geometries if the intersection between the bounding box and a geometry happens to be a point there. If NULL, then the entire tissue is plotted.

annotGraphName
Name of the annotation graph to plot.

annotGeometryName
Name of the annotGeometry, which is associated with the graph specified with annotGraphName, for spatial coordinates of the graph nodes and for context.

Value
A ggplot2 object.

Examples
library(SpatialFeatureExperiment)
library(SFEData)
library(sf)
sfe <- McKellarMuscleData("small")
colGraph(sfe, "visium") <- findVisiumGraph(sfe)
plotColGraph(sfe, colGraphName = "visium", colGeometryName = "spotPoly")
# Make the myofiber segmentations a valid POLYGON geometry
tag <- annotGeometry(sfe, "myofiber_simplified")
tag <- st_buffer(tag, 0)
tag <- tag[!st_is_empty(tag), ]
annotGeometry(sfe, "myofiber_simplified") <- tag
annotGraph(sfe, "myofibers") <-
findSpatialNeighbors(sfe,
  type = "myofiber_simplified", MARGIN = 3,
  method = "tri2nb", dist_type = "idw"
plotCorrelogram

Description

Use ggplot2 to plot correlograms computed by `runUnivariate`, pulling results from `rowData`. Correlograms of multiple genes with error bars can be plotted, and they can be colored by any numeric or categorical column in `rowData` or a vector with the same length as `nrow` of the SFE object. The coloring is useful when the correlograms are clustered to show types of length scales or patterns of decay of spatial autocorrelation. For `method = "I"`, the error bars are twice the standard deviation of the estimated Moran’s I value.

Usage

```r
plotCorrelogram(
  sfe,
  features,
  sample_id = "all",
  method = "I",
  color_by = NULL,
  facet_by = c("sample_id", "features"),
  ncol = NULL,
  colGeometryName = NULL,
  annotGeometryName = NULL,
  reducedDimName = NULL,
  plot_signif = TRUE,
  p_adj_method = "BH",
  divergent = FALSE,
  diverge_center = NULL,
  swap_rownames = NULL,
  name = "sp.correlogram"
)
```

Arguments

- `sfe` A `SpatialFeatureExperiment` object.
- `features` Features to plot, must be in rownames of the gene count matrix, colnames of `colData` or a `colGeometry`, colnames of cell embeddings in `reducedDim`, or numeric indices of dimension reduction components.
plotCorrelogram

**sample_id**
Sample(s) in the SFE object whose cells/spots to use. Can be "all" to compute metric for all samples; the metric is computed separately for each sample.

**method**
"corr" for correlation, "I" for Moran's I, "C" for Geary's C

**color_by**
Name of a column in rowData(sfe) or in the featureData of colData (see colFeatureData), colGeometry, or annotGeometry by which to color the correlogram of each feature. Alternatively, a vector of the same length as features, or a data frame from clusterCorrelograms.

**facet_by**
Whether to facet by sample_id (default) or features. If facetting by sample_id, then different features will be plotted in the same facet for comparison. If facetting by features, then different samples will be compared for each feature. Ignored if only one sample is specified.

**ncol**
Number of columns if facetting.

**colGeometryName**
Name of a colGeometry sf data frame whose numeric columns of interest are to be used to compute the metric. Use colGeometryNames to look up names of the sf data frames associated with cells/spots.

**annotGeometryName**
Name of an annotGeometry of the SFE object, to annotate the gene expression plot.

**reducedDimName**
Name of a dimension reduction, can be seen in reducedDimNames. colGeometryName and annotGeometryName have precedence over reducedDimName.

**plot_signif**
Logical, whether to plot significance symbols: p < 0.001: ***, p < 0.01: **, p < 0.05 * , p < 0.1: , otherwise no symbol. The p-values are two sided, based on the assumption that the estimated Moran's I is normally distributed with mean from a randomized version of the data. The mean and variance come from moran.test for Moran's I and geary.test for Geary's C. Take the results with a grain of salt if the data is not normally distributed.

**p_adj_method**
Multiple testing correction method as in p.adjust, to correct for multiple testing (number of lags times number of features) in the Moran’s I estimates if plot_signif = TRUE.

**divergent**
Logical, whether a divergent palette should be used.

**diverge_center**
If divergent = TRUE, the center from which the palette should diverge. If NULL, then not centering.

**swap_rownames**
Column name of rowData(object) to be used to identify features instead of rownames(object) when labeling plot elements. If not found in rowData, then rownames of the gene count matrix will be used.

**name**
Name under which the correlogram results are stored, which is by default "sp.correlogram".

**Value**
A ggplot object.
Examples

library(SpatialFeatureExperiment)
library(SFEData)
library(bluster)
library(scater)
sfe <- McKellarMuscleData("small")
sfe <- logNormCounts(sfe)
colGraph(sfe, "visium") <- findVisiumGraph(sfe)
inds <- c(1, 3, 4, 5)
features <- rownames(sfe)[inds]
sfe <- runUnivariate(sfe,
    type = "sp.correlogram", features = features,
    exprs_values = "counts", order = 5
)
clust <- clusterCorrelograms(sfe,
    features = features,
    BLUSPARAM = KmeansParam(2)
)
# Color by features
plotCorrelogram(sfe, features)
# Color by something else
plotCorrelogram(sfe, features, color_by = clust$cluster)
# Facet by features
plotCorrelogram(sfe, features, facet_by = "features")

plotCrossVariogram  
Plot cross variogram

Description

Equivalent to gstat::plot.gstatVariogram, but using ggplot2 to be more customizable.

Usage

plotCrossVariogram(res, show_np = TRUE)

Arguments

res  
Cross variogram results for one sample, from calculateBivariate. Global bivariate results are not stored in the SFE object.

show_np  
Logical, whether to show number of pairs of cells at each distance bin.

Value

A ggplot object. Unfortunately I haven’t figured out a way to collect all the facet labels to the top of the entire plot.
See Also

plotCrossVariogramMap

Examples

```r
library(SFEData)
library(scater)
sfe <- McKellarMuscleData()
sfe <- sfe[, sfe$in_tissue]
sfe <- logNormCounts(sfe)

res <- calculateBivariate(sfe, type = "cross_variogram",
feature1 = c("Myh1", "Myh2", "Csrp3"), swap_rownames = "symbol")
plotCrossVariogram(res)
```

---

`plotCrossVariogramMap`  
*Plot cross variogram map*

### Description

Equivalent to `gstat::plot.gstatVariogram`, but using `ggplot2` to be more customizable.

### Usage

```r
plotCrossVariogramMap(res, plot_np = FALSE)
```

### Arguments

- **res**  
  Cross variogram results for one sample, from `calculateBivariate`. Global bivariate results are not stored in the SFE object.

- **plot_np**  
  Logical, whether to plot the number of pairs in each distance bin instead of the variance.

### Value

A `ggplot` object.

### See Also

plotCrossVariogram
Examples

```r
library(SFEData)
library(scater)
sfe <- McKellarMuscleData()
sfe <- sfe[,sfe$in_tissue]
sfe <- logNormCounts(sfe)

res <- calculateBivariate(sfe, type = "cross_variogram_map",
 feature1 = c("Myh1", "Myh2", "Csrp3"), swap_rownames = "symbol",
 width = 500, cutoff = 2000)
plotCrossVariogramMap(res)
```

Description

Just like Seurat’s VizDimLoadings function. I haven’t found an equivalent for SCE but find it useful. But I’m not trying to reproduce that Seurat function exactly. For instance, I don’t like it when Seurat imposes a ggplot theme, and I don’t like the cowplot theme. Maybe I should rewrite it in base R but for now I’m using Tidyverse.

Usage

```r
plotDimLoadings(
  sce,
  dims = 1:4,
  nfeatures = 10,
  swap_rownames = NULL,
  reduction = "PCA",
  balanced = TRUE,
  ncol = 2,
  sample_id = "all"
)
```

Arguments

- **sce**: A `SingleCellExperiment` object, or anything that inherits from `SingleCellExperiment`.
- **dims**: Numeric vector specifying which PCs to plot. For MULTISPATI, PCs with negative eigenvalues are in the right most columns of the embedding and loading matrices. See the `ElbowPlot`
- **nfeatures**: Number of genes to plot.
- **swap_rownames**: Column name of `rowData(object)` to be used to identify features instead of `rownames(object)` when labeling plot elements. If not found in `rowData`, then `rownames of the gene count matrix` will be used.
Name of the dimension reduction to use. It must have an attribute called either "percentVar" or "eig" for eigenvalues. Defaults to "PCA".

Return an equal number of genes with + and - scores. If FALSE, returns the top genes ranked by the scores absolute values.

Number of columns in the facetted plot.

Sample(s) in the SFE object whose cells/spots to use. Can be "all" to compute metric for all samples; the metric is computed separately for each sample.

A ggplot object. Loadings for different PCs are plotted in different facets so one ggplot object is returned.

Different samples are plotted in separate facets.

Name of the geometry associated with the MARGIN of interest for which to compute the graph.

A SpatialFeatureExperiment object.

Name of the geometry associated with the MARGIN of interest for which to compute the graph.
### plotLocalResult

Plot local results

**Description**

Plot results of local spatial analyses in space, such as local Getis-Ord Gi* values.

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>MARGIN</strong></td>
<td>Just like in <code>apply</code>, where 1 stands for row, 2 stands for column. Here, in addition, 3 stands for annotation, to query the <code>annotGeometries</code>, such as nuclei segmentation in a Visium data.</td>
</tr>
<tr>
<td><strong>sample_id</strong></td>
<td>Sample(s) in the SFE object whose cells/spots to use. Can be &quot;all&quot; to compute metric for all samples; the metric is computed separately for each sample.</td>
</tr>
<tr>
<td><strong>ncol</strong></td>
<td>Number of columns if plotting multiple features. Defaults to NULL, which means using the same logic as <code>facet_wrap</code>, which is used by patchwork’s <code>wrap_plots</code> by default.</td>
</tr>
<tr>
<td><strong>bbox</strong></td>
<td>A bounding box to specify a smaller region to plot, useful when the dataset is large. Can be a named numeric vector with names &quot;xmin&quot;, &quot;xmax&quot;, &quot;ymin&quot;, and &quot;ymax&quot;, in any order. If plotting multiple samples, it should be a matrix with sample IDs as column names and &quot;xmin&quot;, &quot;ymin&quot;, &quot;xmax&quot;, and &quot;ymax&quot; as row names. If multiple samples are plotted but <code>bbox</code> is a vector rather than a matrix, then the same bounding box will be used for all samples. You may see points at the edge of the geometries if the intersection between the bounding box and a geometry happens to be a point there. If NULL, then the entire tissue is plotted.</td>
</tr>
<tr>
<td><strong>image_id</strong></td>
<td>ID of the image to plot behind the geometries. If NULL, then not plotting images. Use <code>imgData</code> to see image IDs present.</td>
</tr>
<tr>
<td><strong>maxcell</strong></td>
<td>Maximum number of pixels to plot in the image. If the image is larger, it will be resampled so it have less than this number of pixels to save memory and for faster plotting. We recommend reducing this number when plotting multiple facets.</td>
</tr>
</tbody>
</table>

**Value**

A ggplot object.

**Examples**

```r
library(SFEData)
sfe1 <- McKellarMuscleData("small")
sfe2 <- McKellarMuscleData("small2")
sfe <- cbind(sfe1, sfe2)
sfe <- removeEmptySpace(sfe)
plotGeometry(sfe, "spotPoly")
plotGeometry(sfe, "myofiber_simplified", MARGIN = 3)
```
plotLocalResult

Usage

plotLocalResult(
  sfe,
  name,
  features,
  attribute = NULL,
  sample_id = "all",
  colGeometryName = NULL,
  annotGeometryName = NULL,
  ncol = NULL,
  ncol_sample = NULL,
  annot_aes = list(),
  annot_fixed = list(),
  bbox = NULL,
  image_id = NULL,
  maxcell = 5e+05,
  aes_use = c("fill", "color", "shape", "linetype"),
  divergent = FALSE,
  diverge_center = NULL,
  annot_divergent = FALSE,
  annot_diverge_center = NULL,
  size = 0.5,
  shape = 16,
  linewidth = 0,
  linetype = 1,
  alpha = 1,
  color = "black",
  fill = "gray80",
  swap_rownames = NULL,
  scattermore = FALSE,
  pointsize = 0,
  bins = NULL,
  summary_fun = sum,
  hex = FALSE,
  dark = FALSE,
  type = name,
  ...
)

Arguments

sfe A SpatialFeatureExperiment object.
name Which local spatial results. Use localResultNames to see which types of results have already been calculated.
features Character vector of vectors. To see which features have the results of a given type, see localResultFeatures.
attribute Which field in the local results of the type and features. If the result of each feature is a vector, the this argument is ignored. But if the result is a data frame
or a matrix, then this is the column name of the result, such as "Ii" for local Moran's I. For each local spatial analysis method, there's a default attribute. See Details. Use `localResultAttrs`.

**sample_id**
Sample(s) in the SFE object whose cells/spots to use. Can be "all" to compute metric for all samples; the metric is computed separately for each sample.

**colGeometryName**
Name of a `colGeometry` sf data frame whose numeric columns of interest are to be used to compute the metric. Use `colGeometryNames` to look up names of the sf data frames associated with cells/spots.

**annotGeometryName**
Name of a `annotGeometry` of the SFE object, to annotate the gene expression plot.

**ncol**
Number of columns if plotting multiple features. Defaults to NULL, which means using the same logic as `facet_wrap`, which is used by `patchwork`'s `wrap_plots` by default.

**ncol_sample**
If plotting multiple samples as facets, how many columns of such facets. This is distinct from `ncols`, which is for multiple features. When plotting multiple features for multiple samples, then the result is a multi-panel plot each panel of which is a plot for each feature faceted by samples.

**annot_aes**
A named list of plotting parameters for the annotation sf data frame. The names are which geom (as in ggplot2, such as color and fill), and the values are column names in the annotation sf data frame. Tidyeval is NOT supported.

**annot_fixed**
Similar to `annot_aes` but for fixed aesthetic settings, such as color = "gray". The defaults are the same as the relevant defaults for this function.

**bbox**
A bounding box to specify a smaller region to plot, useful when the dataset is large. Can be a named numeric vector with names "xmin", "xmax", "ymin", and "ymax", in any order. If plotting multiple samples, it should be a matrix with sample IDs as column names and "xmin", "ymin", "xmax", and "ymax" as row names. If multiple samples are plotted but bbox is a vector rather than a matrix, then the same bounding box will be used for all samples. You may see points at the edge of the geometries if the intersection between the bounding box and a geometry happens to be a point there. If NULL, then the entire tissue is plotted.

**image_id**
ID of the image to plot behind the geometries. If NULL, then not plotting images. Use `imgData` to see image IDs present.

**maxcell**
Maximum number of pixels to plot in the image. If the image is larger, it will be resampled so it have less than this number of pixels to save memory and for faster plotting. We recommend reducing this number when plotting multiple facets.

**aes_use**
Aesthetic to use for discrete variables. For continuous variables, it's always "fill" for polygons and point shapes 21-25. For discrete variables, it can be fill, color, shape, or linetype, whenever applicable. The specified value will be changed to the applicable equivalent. For example, if the geometry is point but "linetype" is specified, then "shaped" will be used instead.

**divergent**
Logical, whether a divergent palette should be used.
If `divergent = TRUE`, the center from which the palette should diverge. If `NULL`, then not centering.

Just as `divergent`, but for the `annotGeometry` in case it’s different.

Just as `diverge_center`, but for the `annotGeometry` in case it’s different.

Fixed size of points. For points defaults to 0.5. Ignored if `size_by` is specified.

Fixed shape of points, ignored if `shape_by` is specified and applicable.

Width of lines, including outlines of polygons. For polygons, this defaults to 0, meaning no outlines.

Fixed line type, ignored if `linetype_by` is specified and applicable.

Transparency.

Fixed color for `colGeometry` if `color_by` is not specified or not applicable, or for `annotGeometry` if `annot_color_by` is not specified or not applicable.

Similar to `color`, but for fill.

Column name of `rowData(object)` to be used to identify features instead of `rownames(object)` when labeling plot elements. If not found in `rowData`, then `rownames` of the gene count matrix will be used.

Logical, whether to use the `scattermore` package to greatly speed up plotting numerous points. Only used for POINT `colGeometries`. If the geometry is not POINT, then the centroids are used. Recommended for plotting hundreds of thousands or more cells where the cell polygons can’t be seen when plotted due to the large number of cells and small plot size such as when plotting multiple panels for multiple features.

Radius of rasterized point in `scattermore`. Default to 0 for single pixels (fastest).

If binning the `colGeometry` in space due to large number of cells or spots, the number of bins, passed to `geom_bin2d` or `geom_hex`. If `NULL` (default), then the `colGeometry` is plotted without binning. If binning, a point geometry is recommended. If the geometry is not point, then the centroids will be used.

Function to summarize the feature value when the `colGeometry` is binned.

Logical, whether to use `geom_hex`. Note that `geom_hex` is broken in `ggplot2` version 3.4.0. Please update `ggplot2` if you are getting horizontal stripes when `hex = TRUE`.

Logical, whether to use dark theme. When using dark theme, the palette will have lighter color represent higher values as if glowing in the dark. This is intended for plotting gene expression on top of fluorescent images.

An `SFEMethod` object or a string corresponding to the name of one of such objects in the environment. If the `localResult` of interest was manually added outside `runUnivariate` and `runBivariate`, so the method is not recorded, then the `type` argument can be used to specify the method to properly get the title and labels. By default, this argument is set to be the same as argument name. If the method parameters are recorded, then the `type` argument is ignored.

Other arguments passed to `wrap_plots`. 
plotLocalResult

Details

Many local spatial analyses return a data frame or matrix as the results, whose columns can be the statistic of interest at each location, its variance, expected value from permutation, p-value, and etc. The attribute argument specifies which column to use when there are multiple columns. Below are the defaults for each local method supported by this package what they mean:

- **localmoran and localmoran_perm Ii**, local Moran’s I statistic at each location.
- **localC_perm localC**, the local Geary C statistic at each location.
- **localG and localG_perm localG**, the local Getis-Ord Gi or Gi* statistic. If include_self = TRUE when calculateUnivariate or runUnivariate was called, then it would be Gi*. Otherwise it’s Gi.
- **LOSH and LOSH.mc Hi**, local spatial heteroscedasticity
- **moran.plot wx**, the average of the value of each neighbor of each location. Moran plot is best plotted as a scatter plot of wx vs x. See moranPlot.

Other local methods not listed above return vectors as results. For instance, localC returns a vector by default, which is the local Geary’s C statistic.

Value

A ggplot2 object if plotting one feature. A patchwork object if plotting multiple features.

Note

While this function shares internals with plotSpatialFeature, there are some important differences. In plotSpatialFeature, the annotGeometry is indeed only used for annotation and the protagonist is the colGeometry, since it’s easy to directly use ggplot2 to plot the data in annotGeometry sf data frames while overlaying annotGeometry and colGeometry involves more complicated code. In contrast, in this function, local results for annotGeometry can be plotted separately without anything related to colGeometry. Note that when annotGeometry local results are plotted without colGeometry, the annot_* arguments are ignored. Use the other arguments for aesthetics as if it’s for colGeometry.

Examples

```r
library(SpatialFeatureExperiment)
library(SFEData)
library(scater)
sfe <- McKellarMuscleData("small")
sfe <- sfe[,sfe$in_tissue]
colGraph(sfe, "visium") <- findVisiumGraph(sfe)
feature_use <- rownames(sfe)[1]
sfe <- logNormCounts(sfe)
sfe <- runUnivariate(sfe, "localmoran", feature_use)
# Which types of results are available?
localResultNames(sfe)
# Which features for localmoran?
localResultFeatures(sfe, "localmoran")
# Which columns does the localmoran results have?
```
localResultAttrs(sfe, "localmoran", feature_use)
plotLocalResult(sfe, "localmoran", feature_use, "Ii",
    colGeometryName = "spotPoly"
)

# For annotGeometry
# Make sure it's type POLYGON
annotGeometry(sfe, "myofiber_simplified") <-
    sf::st_buffer(annotGeometry(sfe, "myofiber_simplified"), 0)
annotGraph(sfe, "poly2nb_myo") <-
    findSpatialNeighbors(sfe,
        type = "myofiber_simplified", MARGIN = 3,
        method = "poly2nb", zero.policy = TRUE
    )
sfe <- annotGeometryUnivariate(sfe, "localmoran",
    features = "area",
    annotGraphName = "poly2nb_myo",
    annotGeometryName = "myofiber_simplified",
    zero.policy = TRUE
)
plotLocalResult(sfe, "localmoran", "area", "Ii",
    annotGeometryName = "myofiber_simplified",
    size = 0.3, color = "cyan"
)
plotLocalResult(sfe, "localmoran", "area", "Z.Ii",
    annotGeometryName = "myofiber_simplified"
)
# don’t use annot_* arguments when annotGeometry is plotted without colGeometry

---

plotMoranMC  

*Plot Moran/Geary Monte Carlo results*

**Description**

Plot the simulations as a density plot or histogram compared to the observed Moran’s I or Geary’s C, with ggplot2 so it looks nicer. Unlike the plotting function in spdep, this function can also plot the same feature in different samples as facets or plot different features or samples together for comparison.

**Usage**

```r
plotMoranMC(
    sfe,
    features,
    sample_id = "all",
    facet_by = c("sample_id", "features"),
    ncol = NULL,
    colGeometryName = NULL,
    annotGeometryName = NULL,
)```

```r
# Example usage...
plotMoranMC(sfe, features, sample_id = "myofiber_simplified")
```

---
Arguments

`sfe` A SpatialFeatureExperiment object.

`features` Features to plot, must be in rownames of the gene count matrix, colnames of `colData` or a `colGeometry`, colnames of cell embeddings in `reducedDim`, or numeric indices of dimension reduction components.

`sample_id` Sample(s) in the SFE object whose cells/spots to use. Can be "all" to compute metric for all samples; the metric is computed separately for each sample.

`facet_by` Whether to facet by sample_id (default) or features. If facetting by sample_id, then different features will be plotted in the same facet for comparison. If facetting by features, then different samples will be compared for each feature. Ignored if only one sample is specified.

`ncol` Number of columns if facetting.

`colGeometryName` Name of a `colGeometry` `sf` data frame whose numeric columns of interest are to be used to compute the metric. Use `colGeometryNames` to look up names of the `sf` data frames associated with cells/spots.

`annotGeometryName` Name of a `annotGeometry` of the SFE object, to annotate the gene expression plot.

`reducedDimName` Name of a dimension reduction, can be seen in `reducedDimNames`. `colGeometryName` and `annotGeometryName` have precedence over `reducedDimName`.

`ptype` Plot type, one of "density", "histogram", or "freqpoly".

`swap_rownames` Column name of `rowData(object)` to be used to identify features instead of `rownames(object)` when labeling plot elements. If not found in `rowData`, then `rownames of the gene count matrix will be used.

`name` Name under which the Monte Carlo results are stored, which defaults to "moran.mc". For Geary's C Monte Carlo, the default is "geary.mc".

... Other arguments passed to `geom_density`, `geom_histogram`, or `geom_freqpoly`, depending on `ptype`.

Value

A ggplot2 object.

Examples

```r
library(SpatialFeatureExperiment)
library(SFEData)
```
sfe <- McKellarMuscleData("small")
colGraph(sfe, "visium") <- findVisiumGraph(sfe)
sfe <- colDataUnivariate(sfe, type = "moran.mc", "nCounts", nsim = 100)
plotM MoranMC(sfe, "nCounts")

---

**plotSpatialFeature**  
*Plot gene expression in space*

**Description**

Unlike Seurat and ggspavis, plotting functions in this package uses `geom_sf` whenever applicable.

**Usage**

```r
plotSpatialFeature(
  sfe, 
  features, 
  colGeometryName = 1L, 
  sample_id = "all", 
  ncol = NULL, 
  ncol_sample = NULL, 
  annotGeometryName = NULL, 
  annot_aes = list(), 
  annot_fixed = list(), 
  exprs_values = "logcounts", 
  bbox = NULL, 
  image_id = NULL, 
  maxcell = 5e+05, 
  aes_use = c("fill", "color", "shape", "linetype"), 
  divergent = FALSE, 
  diverge_center = NA, 
  annot_divergent = FALSE, 
  annot_diverge_center = NA, 
  size = 0.5, 
  shape = 16, 
  linewidth = 0, 
  linetype = 1, 
  alpha = 1, 
  color = "black", 
  fill = "gray80", 
  swap_rownames = NULL, 
  scattermore = FALSE, 
  pointsize = 0, 
  bins = NULL, 
  summary_fun = sum, 
  hex = FALSE, 
  dark = FALSE,
```

Arguments

sfe
A SpatialFeatureExperiment object.

features
Features to plot, must be in rownames of the gene count matrix, colnames of
colData or a colGeometry.

colGeometryName
Name of a colGeometry sf data frame whose numeric columns of interest are
to be used to compute the metric. Use colGeometryNames to look up names of
the sf data frames associated with cells/spots.

sample_id
Sample(s) in the SFE object whose cells/spots to use. Can be "all" to compute
metric for all samples; the metric is computed separately for each sample.

ncol
Number of columns if plotting multiple features. Defaults to NULL, which means
using the same logic as facet_wrap, which is used by patchwork’s wrap_plots
by default.

ncol_sample
If plotting multiple samples as facets, how many columns of such facets. This
is distinct from ncol, which is for multiple features. When plotting multiple
features for multiple samples, then the result is a multi-panel plot each panel of
which is a plot for each feature faceted by samples.

annotGeometryName
Name of a annotGeometry of the SFE object, to annotate the gene expression
plot.

annot_aes
A named list of plotting parameters for the annotation sf data frame. The names
are which geom (as in ggplot2, such as color and fill), and the values are column
names in the annotation sf data frame. Tidyeval is NOT supported.

annot_fixed
Similar to annot_aes, but for fixed aesthetic settings, such as color = "gray".
The defaults are the same as the relevant defaults for this function.

exprs_values
Integer scalar or string indicating which assay of x contains the expression val-
ues.

bbox
A bounding box to specify a smaller region to plot, useful when the dataset is
large. Can be a named numeric vector with names "xmin", "xmax", "ymin", and
"ymax", in any order. If plotting multiple samples, it should be a matrix with
sample IDs as column names and "xmin", "ymin", "xmax", and "ymax" as row
names. If multiple samples are plotted but bbox is a vector rather than a matrix,
then the same bounding box will be used for all samples. You may see points at
the edge of the geometries if the intersection between the bounding box and a
geometry happens to be a point there. If NULL, then the entire tissue is plotted.

image_id
ID of the image to plot behind the geometries. If NULL, then not plotting images.
Use imgData to see image IDs present.

maxcell
Maximum number of pixels to plot in the image. If the image is larger, it will
be resampled so it have less than this number of pixels to save memory and for
faster plotting. We recommend reducing this number when plotting multiple
facets.
plotSpatialFeature

aes_use Aesthetic to use for discrete variables. For continuous variables, it’s always "fill" for polygons and point shapes 21-25. For discrete variables, it can be fill, color, shape, or linetype, whenever applicable. The specified value will be changed to the applicable equivalent. For example, if the geometry is point but "linetype" is specified, then "shaped" will be used instead.

divergent Logical, whether a divergent palette should be used.
diverge_center If divergent = TRUE, the center from which the palette should diverge. If NULL, then not centering.
annot_divergent Just as divergent, but for the annotGeometry in case it’s different.
annot_diverge_center Just as diverge_center, but for the annotGeometry in case it’s different.
size Fixed size of points. For points defaults to 0.5. Ignored if size_by is specified.
shape Fixed shape of points, ignored if shape_by is specified and applicable.
linewidth Width of lines, including outlines of polygons. For polygons, this defaults to 0, meaning no outlines.
linetype Fixed line type, ignored if linetype_by is specified and applicable.
alpha Transparency.
color Fixed color for colGeometry if color_by is not specified or not applicable, or for annotGeometry if annot_color_by is not specified or not applicable.
fill Similar to color, but for fill.
swap_rownames Column name of rowData(object) to be used to identify features instead of rownames(object) when labeling plot elements. If not found in rowData, then rownames of the gene count matrix will be used.
scattermore Logical, whether to use the scattermore package to greatly speed up plotting numerous points. Only used for POINT colGeometries. If the geometry is not POINT, then the centroids are used. Recommended for plotting hundreds of thousands or more cells where the cell polygons can’t be seen when plotted due to the large number of cells and small plot size such as when plotting multiple panels for multiple features.
pointsize Radius of rasterized point in scattermore. Default to 0 for single pixels (fastest).
bins If binning the colGeometry in space due to large number of cells or spots, the number of bins, passed to geom_bin2d or geom_hex. If NULL (default), then the colGeometry is plotted without binning. If binning, a point geometry is recommended. If the geometry is not point, then the centroids will be used.
summary_fun Function to summarize the feature value when the colGeometry is binned.
hex Logical, whether to use geom_hex. Note that geom_hex is broken in ggplot2 version 3.4.0. Please update ggplot2 if you are getting horizontal stripes when hex = TRUE.
dark Logical, whether to use dark theme. When using dark theme, the palette will have lighter color represent higher values as if glowing in the dark. This is intended for plotting gene expression on top of fluorescent images.

... Other arguments passed to wrap_plots.
Details

In the documentation of this function, a "feature" can be a gene (or whatever entity that corresponds to rows of the gene count matrix), a column in colData, or a column in the colGeometry sf data frame specified in the colGeometryName argument.

In the light theme, for continuous variables, the Blues palette from colorbrewer is used if divergent = FALSE, and the roma palette from the scico package if divergent = TRUE. In the dark theme, the nuuk palette from scico is used if divergent = FALSE, and the berlin palette from scico is used if divergent = TRUE. For discrete variables, the dittoSeq palette is used.

For annotation, the YlOrRd colorbrewer palette is used for continuous variables in the light theme. In the dark theme, the acton palette from scico is used when divergent = FALSE and the vanimo palette from scico is used when divergent = FALSE. The other end of the dittoSeq palette is used for discrete variables.

Each individual palette should be colorblind friendly, but when plotting continuous variables coloring a colGeometry and a annotGeometry simultaneously, the combination of the two palettes is not guaranteed to be colorblind friendly.

In addition, when plotting an image behind the geometries, the colors of the image may distort color perception of the values of the geometries.

theme_void is used for all spatial plots in this package, because the units in the spatial coordinates are often arbitrary. This can be overridden to show the axes by using a different theme as normally done in ggplot2.

Value

A ggplot2 object if plotting one feature. A patchwork object if plotting multiple features.

Examples

library(SFEData)
library(sf)
sfe <- McKellarMuscleData("small")
# features can be genes or colData or colGeometry columns
plotSpatialFeature(sfe, c("nCounts", rownames(sfe)[1]),
  exprs_values = "counts",
  colGeometryName = "spotPoly",
  annotGeometryName = "tissueBoundary"
)
# Change fixed aesthetics
plotSpatialFeature(sfe, "nCounts",
  colGeometryName = "spotPoly",
  annotGeometryName = "tissueBoundary",
  annot_fixed = list(color = "blue", size = 0.3, fill = NA),
  alpha = 0.7)
# Make the myofiber segmentations a valid POLYGON geometry
ag <- annotGeometry(sfe, "myofiber_simplified")
ag <- st_buffer(ag, 0)
ag <- ag[!st_is_empty(ag), ]
annotGeometry(sfe, "myofiber_simplified") <- ag
# Also plot an annotGeometry variable
plotVariogram(sfe, "nCounts",
        colGeometryName = "spotPoly",
        annotGeometryName = "myofiber_simplified",
        annot_aes = list(fill = "area")
    )

# Use a bounding box to zoom in
bbox <- c(xmin = 5500, ymin = 13500, xmax = 6000, ymax = 14000)
plotSpatialFeature(sfe, "nCounts", colGeometryName = "spotPoly",
        annotGeometry = "myofiber_simplified",
        bbox = bbox, annot_fixed = list(linewidth = 0.3))

---

**Description**

This function plots the variogram of a feature and its fitted variogram models, showing the nugget, range, and sill of the model. Unlike the plotting functions in package `automap` that uses `lattice`, this function uses `ggplot2` to make prettier and more customizable plots.

**Usage**

```r
plotVariogram(
    sfe,
    features,
    sample_id = "all",
    color_by = NULL,
    group = c("none", "sample_id", "features", "angles"),
    use_lty = TRUE,
    show_np = TRUE,
    ncol = NULL,
    colGeometryName = NULL,
    annotGeometryName = NULL,
    reducedDimName = NULL,
    divergent = FALSE,
    diverge_center = NULL,
    swap_rownames = NULL,
    name = "variogram"
)
```

**Arguments**

- **sfe** A `SpatialFeatureExperiment` object.
- **features** Features to plot, must be in rownames of the gene count matrix, colnames of `colData` or a `colGeometry`, colnames of cell embeddings in `reducedDim`, or numeric indices of dimension reduction components.
- **sample_id** Sample(s) in the SFE object whose cells/spots to use. Can be "all" to compute metric for all samples; the metric is computed separately for each sample.
plotVariogram

**color_by**
Name of a column in rowData(sfe) or in the featureData of colData (see colFeatureData), colGeometry, or annotGeometry by which to color the correlogram of each feature. Alternatively, a vector of the same length as features, or a data frame from clusterCorrelograms.

**group**
Which of samples, features, and angles to show in the same facet for comparison when there are multiple. Default to "none", meaning each facet will contain one variogram. When grouping multiple variograms in the same facet, the text with model, nugget, sill, and range of the variograms will not be shown.

**use_lty**
Logical, whether to use linetype or point shape to distinguish between the different features or samples in the same facet. If FALSE, then the different features or samples are not distinguished and the patterns are shown only.

**show_np**
Logical, whether to show number of pairs of cells at each distance bin.

**ncol**
Number of columns if facetting.

**colGeometryName**
Name of a colGeometry sf data frame whose numeric columns of interest are to be used to compute the metric. Use colGeometryNames to look up names of the sf data frames associated with cells/spots.

**annotGeometryName**
Name of a annotGeometry of the SFE object, to annotate the gene expression plot.

**reducedDimName**
Name of a dimension reduction, can be seen in reducedDimNames. colGeometryName and annotGeometryName have precedence over reducedDimName.

**divergent**
Logical, whether a divergent palette should be used.

**diverge_center**
If divergent = TRUE, the center from which the palette should diverge. If NULL, then not centering.

**swap_rownames**
Column name of rowData(object) to be used to identify features instead of rownames(object) when labeling plot elements. If not found in rowData, then rownames of the gene count matrix will be used.

**name**
Name under which the correlogram results are stored, which is by default "sp.correlogram".

**Value**
A ggplot object. The empirical variogram at each distance bin is plotted as points, and the fitted variogram model is plotted as a line for each feature. The number next to each point is the number of pairs of cells in that distance bin.

**See Also**
plotVariogramMap

**Examples**

```r
library(SFEData)
sfe <- McKellarMuscleData()
sfe <- colDataUnivariate(sfe, "variogram", features = "nCounts", model = "Sph")
plotVariogram(sfe, "nCounts")
```
# Anisotropy, will get a message
sfe <- colDataUnivariate(sfe, "variogram", features = "nCounts",
model = "Sph", alpha = c(30, 90, 150), name = "variogram_anis")
# Facet by angles by default
plotVariogram(sfe, "nCounts", name = "variogram_anis")
# Plot angles with different colors
plotVariogram(sfe, "nCounts", group = "angles", name = "variogram_anis")

plotVariogramMap

Plot variogram maps

Description
Plot variogram maps that show the variogram in all directions in a grid of distances in x and y coordinates.

Usage
plotVariogramMap(
  sfe,
  features,
  sample_id = "all",
  plot_np = FALSE,
  ncol = NULL,
  colGeometryName = NULL,
  annotGeometryName = NULL,
  reducedDimName = NULL,
  swap_rownames = NULL,
  name = "variogram_map"
)

Arguments
sfe A SpatialFeatureExperiment object.
features Features to plot, must be in rownames of the gene count matrix, colnames of
  colData or a colGeometry, colnames of cell embeddings in reducedDim, or numeric
  indices of dimension reduction components.
sample_id Sample(s) in the SFE object whose cells/spots to use. Can be "all" to compute
  metric for all samples; the metric is computed separately for each sample.
plot_np Logical, whether to plot the number of pairs in each distance bin instead of the
  variance.
ncol Number of columns if facetting.
colGeometryName Name of a colGeometry sf data frame whose numeric columns of interest are
  to be used to compute the metric. Use colGeometryNames to look up names of
  the sf data frames associated with cells/spots.
### SFEMethod

This S4 class is used to wrap spatial analysis methods, taking inspiration from the caret and tidymodels packages.

#### Description

The SFEMethod class is used to wrap spatial analysis methods, taking inspiration from the caret and tidymodels packages.

#### Usage

```r
SFEMethod(
  name,
  fun,
  reorganize_fun,
  package,
  variate = c("uni", "bi", "multi"),
  scope = c("global", "local"),
  title = NULL,
  default_attr = NA,
  args_not_check = NA,
  joint = FALSE,
)
```

#### Value

A ggplot object.

#### See Also

- `plotVariogram`

#### Examples

```r
library(SFEData)
sfe <- McKellarMuscleData()
sfe <- colDataUnivariate(sfe, "variogram_map", features = "nCounts",
width = 500, cutoff = 5000)
plotVariogramMap(sfe, "nCounts")
```
use_graph = TRUE,
use_matrix = FALSE,
dest = c("reducedDim", "colData")
)

## S4 method for signature 'SFEMethod'
info(x, type)

## S4 method for signature 'SFEMethod'
is_local(x)

## S4 method for signature 'SFEMethod'
fun(x)

## S4 method for signature 'SFEMethod'
reorganize_fun(x)

## S4 method for signature 'SFEMethod'
args_not_check(x)

## S4 method for signature 'SFEMethod'
is_joint(x)

## S4 method for signature 'SFEMethod'
use_graph(x)

## S4 method for signature 'SFEMethod'
use_matrix(x)

### Arguments

- **name**: Name of the method, used by user-facing functions to specify the method to use, such as "moran" for Moran's I.
- **fun**: Function to run the method. See Details.
- **reorganize_fun**: Function to reorganize results to add to the SFE object. See Details.
- **package**: Name of the package whose implementation of the method is used here, used to check if the package is installed.
- **variate**: How many variables this method works with, must be one of "uni" for univariate, "bi" for bivariate, or "multi" for multivariate.
- **scope**: Either "global", returning one result for the entire dataset, or "local", returning one result for each spatial location. For multivariate methods, this is irrelevant.
- **title**: Descriptive title to show when plotting the results.
- **default_attr**: For local methods that return multiple fields, such as local Moran values and their p-values, the default field to use when plotting.
- **args_not_check**: A character vector indicating which argument are not to be checked when comparing parameters in with those of a previous run.
**SFEMethod**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>joint</td>
<td>Logical, whether it makes sense to run this method to multiple samples jointly. If TRUE, then <code>fun</code> must be able to handle an adjacency matrix for the <code>listw</code> argument because there's no straightforward way to concatenate <code>listw</code> objects from multiple samples.</td>
</tr>
<tr>
<td>use_graph</td>
<td>Logical, to indicate whether the method uses a spatial neighborhood graph because unifying the user facing functions have an argument asking for the graph as most though not all methods require the graph.</td>
</tr>
<tr>
<td>use_matrix</td>
<td>Logical, whether the function in slot <code>fun</code> takes a matrix as input. The argument is only used for bivariate methods.</td>
</tr>
<tr>
<td>dest</td>
<td>Whether the results are more appropriate for <code>reducedDim</code> or <code>colData</code>. Only used for multivariate methods. This overrides the &quot;local&quot; field in <code>info</code>.</td>
</tr>
<tr>
<td>x</td>
<td>A SFEMethod object</td>
</tr>
<tr>
<td>type</td>
<td>One of the names of the <code>info</code> slot, see slot documentation.</td>
</tr>
</tbody>
</table>

**Details**

The `fun` slot should be specified as such:

For all methods, there must be arguments `x` for a vector, `listw` for a `listw` object specifying the spatial neighborhood graph, `zero.policy` specifying what to do with cells without neighbors (default `NULL`, use global option value; if `TRUE` assign zero to the lagged value of zones without neighbours, if `FALSE` assign NA), and optionally other method specific arguments and ... to pass to the underlying imported function. If the original function implementing the method in the package has different argument names or orders, write a thin wrapper to rearrange and/or rename the arguments.

For univariate methods that use a spatial neighborhood graph, the first two arguments must be `x` and `listw`. For univariate methods that don’t use a spatial neighborhood graph, such as the variogram, the first two arguments must be `x` for a numeric vector and `coords_df` for a `sf` data frame with cell locations and optionally other regressors. The `formula` argument is optional and can have defaults specifying regressors to use.

For bivariate methods, the first three arguments must be `x`, `y`, and `listw`.

For multivariate methods, the argument `x` is mandatory, for the matrix input. These arguments must be present but can be optional by having defaults: `listw` and `ncomponents` to set the number of dimensions in the output.

The `reorganize_fun` slot should be specified as such:

Univariate methods are meant to be run separately for each gene, so the input to `reorganize_fun` in the argument `out` should be a list of outputs; each element of the list corresponds to the output of a gene.

For univariate global methods, different fields of the result should be columns of a data frame with one row so results for multiple features will be a data frame. The arguments should be `out`, and `name` to rename the primary field if a more informative name is needed, and ... for other arguments specific to methods. The output of `reorganize_fun` should be a `DataFrame` whose rows correspond to the genes and columns correspond to fields in the output.

For univariate local methods, the arguments should be `out`, `nb` for a neighborhood list used for multiple testing correction, and `p.adjust.method` for a method to correct for multiple testing as in `p.adjust`, and .... The output of `reorganize_fun` should be a list of reorganized output. Each
element of the list corresponds to a gene, and the reorganized content of the element can be a vector, matrix, or data frame, but they must all have the same dimensions for all genes. Each element of the vector, or each row of the matrix or data frame corresponds to a cell.

For multivariate methods whose results go into `reducedDim`, `reorganize_fun` should have one argument out for the raw output. The output of `reorganize_fun` should be the cell embedding matrix ready to be added to `reducedDim`. Other relevant information such as gene loadings and eigenvalues should be added to the attributes of the cell embedding matrix.

For multivariate methods whose results can go into `colData`, the arguments should be `out`, `nb`, and `p.adjust.method`. Unlike the univariate local counterpart, `out` takes the raw output instead of a list of outputs. The output of `reorganize_fun` is a vector or a data frame ready to be added to `colData`.

**Value**

The constructor returns an `SFEMethod` object. The getters return the content of the corresponding slots.

**Slots**

- `info` A named character vector specifying information about the method.
- `fun` The function implementing the method. See Details.
- `reorganize_fun` Function to convert output from `fun` into a format to store in the SFE object. See Details.
- `misc` Miscellaneous information on how the method interacts with the rest of the package. This should be a named list.

**Examples**

```r
moran <- SFEMethod(
  name = "moran", title = "Moran's I", package = "spdep", variate = "uni",
  scope = "global",
  fun = function(x, listw, zero.policy = NULL)
    spdep::moran(x, listw, n = length(listw$neighbours), S0 = spdep::Szero(listw),
                 zero.policy = zero.policy),
  reorganize_fun = Voyager:::.moran2df
)
```

**spatialReducedDim**

*Plot dimension reduction components in space*

**Description**

Such as plotting the value of projection of gene expression of each cell to a principal component in space. At present, this function does not work for the 3D array of geographically weighted PCA (GWPCA), but a future version will deal with GWPCA results.
spatialReducedDim

Usage

spatialReducedDim(
  sfe,
  dimred,
  ncomponents = NULL,
  components = ncomponents,
  colGeometryName = 1L,
  sample_id = "all",
  ncol = NULL,
  ncol_sample = NULL,
  annotGeometryName = NULL,
  annot_aes = list(),
  annot_fixed = list(),
  exprs_values = "logcounts",
  bbox = NULL,
  image_id = NULL,
  maxcell = 5e+05,
  aes_use = c("fill", "color", "shape", "linetype"),
  divergent = FALSE,
  diverge_center = NULL,
  annot_divergent = FALSE,
  annot_diverge_center = NULL,
  size = 0,
  shape = 16,
  linewidth = 0,
  linetype = 1,
  alpha = 1,
  color = NA,
  fill = "gray80",
  scattermore = FALSE,
  pointsize = 0,
  bins = NULL,
  summary_fun = sum,
  hex = FALSE,
  dark = FALSE,
  ...
)

Arguments

sfe          A SpatialFeatureExperiment object.
dimred       A string or integer scalar indicating the reduced dimension result in reducedDims(sfe) to plot.
ncomponents  A numeric scalar indicating the number of dimensions to plot, starting from the first dimension. Alternatively, a numeric vector specifying the dimensions to be plotted.
components   A numeric scalar or vector specifying which dimensions to be plotted. Use this instead of ncomponents when plotting only one dimension.
colGeometryName
Name of a colGeometry sf data frame whose numeric columns of interest are to be used to compute the metric. Use colGeometryNames to look up names of the sf data frames associated with cells/spots.

sample_id
Sample(s) in the SFE object whose cells/spots to use. Can be "all" to compute metric for all samples; the metric is computed separately for each sample.

ncol
Number of columns if plotting multiple features. Defaults to NULL, which means using the same logic as facet_wrap, which is used by patchwork’s wrap_plots by default.

ncol_sample
If plotting multiple samples as facets, how many columns of such facets. This is distinct from ncolS, which is for multiple features. When plotting multiple features for multiple samples, then the result is a multi-panel plot each panel of which is a plot for each feature facetted by samples.

annotGeometryName
Name of a annotGeometry of the SFE object, to annotate the gene expression plot.

annot_aes
A named list of plotting parameters for the annotation sf data frame. The names are which geom (as in ggplot2, such as color and fill), and the values are column names in the annotation sf data frame. Tidyeval is NOT supported.

annot_fixed
Similar to annot_aes, but for fixed aesthetic settings, such as color = "gray". The defaults are the same as the relevant defaults for this function.

exprs_values
Integer scalar or string indicating which assay of x contains the expression values.

bbox
A bounding box to specify a smaller region to plot, useful when the dataset is large. Can be a named numeric vector with names "xmin", "xmax", "ymin", and "ymax", in any order. If plotting multiple samples, it should be a matrix with sample IDs as column names and "xmin", "ymin", "xmax", and "ymax" as row names. If multiple samples are plotted but bbox is a vector rather than a matrix, then the same bounding box will be used for all samples. You may see points at the edge of the geometries if the intersection between the bounding box and a geometry happens to be a point there. If NULL, then the entire tissue is plotted.

image_id
ID of the image to plot behind the geometries. If NULL, then not plotting images. Use imgData to see image IDs present.

maxcell
Maximum number of pixels to plot in the image. If the image is larger, it will be resampled so it have less than this number of pixels to save memory and for faster plotting. We recommend reducing this number when plotting multiple facets.

aes_use
Aesthetic to use for discrete variables. For continuous variables, it’s always "fill" for polygons and point shapes 21-25. For discrete variables, it can be fill, color, shape, or linetype, whenever applicable. The specified value will be changed to the applicable equivalent. For example, if the geometry is point but "linetype" is specified, then "shaped" will be used instead.

divergent
Logical, whether a divergent palette should be used.

diverge_center
If divergent = TRUE, the center from which the palette should diverge. If NULL, then not centering.
annot_divergent
Just as divergent, but for the annotGeometry in case it's different.

annot_diverge_center
Just as diverge_center, but for the annotGeometry in case it's different.

size
Fixed size of points. For points defaults to 0.5. Ignored if size_by is specified.

shape
Fixed shape of points, ignored if shape_by is specified and applicable.

linewidth
Width of lines, including outlines of polygons. For polygons, this defaults to 0, meaning no outlines.

linetype
Fixed line type, ignored if linetype_by is specified and applicable.

alpha
Transparency.

color
Fixed color for colGeometry if color_by is not specified or not applicable, or for annotGeometry if annot_color_by is not specified or not applicable.

fill
Similar to color, but for fill.

scattermore
Logical, whether to use the scattermore package to greatly speed up plotting numerous points. Only used for POINT colGeometries. If the geometry is not POINT, then the centroids are used. Recommended for plotting hundreds of thousands or more cells where the cell polygons can’t be seen when plotted due to the large number of cells and small plot size such as when plotting multiple panels for multiple features.

pointsze
Radius of rasterized point in scattermore. Default to 0 for single pixels (fastest).

bins
If binning the colGeometry in space due to large number of cells or spots, the number of bins, passed to geom_bin2d or geom_hex. If NULL (default), then the colGeometry is plotted without binning. If binning, a point geometry is recommended. If the geometry is not point, then the centroids will be used.

summary_fun
Function to summarize the feature value when the colGeometry is binned.

hex
Logical, whether to use geom_hex. Note that geom_hex is broken in ggplot2 version 3.4.0. Please update ggplot2 if you are getting horizontal stripes when hex = TRUE.

dark
Logical, whether to use dark theme. When using dark theme, the palette will have lighter color represent higher values as if glowing in the dark. This is intended for plotting gene expression on top of fluorescent images.

...
Other arguments passed to wrap_plots.

Value
Same as in plotSpatialFeature. A ggplot2 object if plotting one component. A patchwork object if plotting multiple components.

See Also
scater::plotReducedDim
Examples

```r
library(SFEData)
library(scater)
sfe <- McKellar MuscleData("small")
sfe <- logNormCounts(sfe)
sfe <- runPCA(sfe, ncomponents = 2)
spatialReducedDim(sfe, "PCA", ncomponents = 2, "spotPoly",
      annotGeometryName = "tissueBoundary",
      divergent = TRUE, diverge_center = 0
)
# Basically PC1 separates spots not on tissue from those on tissue.
```

Variogram-internal  Compute variograms

Description

Wrapper of automap::autofitVariogram to facilitate computing variograms for multiple genes in SFE objects as an EDA tool. These functions are written to conform to a uniform format for univariate methods to be called internally. These functions are not exported, but the documentation is written to show users the extra arguments to use when calling calculateUnivariate or runUnivariate.

Usage

```r
.variogram(x, coords_df, formula = x ~ 1, scale = TRUE, ...)
.variogram_bv(x, y, coords_df, scale = TRUE, map = FALSE, ...)
.cross_variogram(x, y, coords_df, scale = TRUE, ...)
.cross_variogram_map(x, y, coords_df, width, cutoff, scale = TRUE, ...)
.variogram_map(x, coords_df, formula = x ~ 1, width, cutoff, scale = TRUE, ...)
```

Arguments

- **x**: A numeric vector whose variogram is computed.
- **coords_df**: A sf data frame with the geometry and regressors for variogram modeling.
- **formula**: A formula defining the response vector and (possible) regressors, in case of absence of regressors, use `x ~ 1`.
- **scale**: Logical, whether to scale `x`. Defaults to `TRUE` so the variogram is easier to interpret and is more comparable between features with different magnitudes when the length scale of spatial autocorrelation is of interest.
... Other arguments passed to automap::autofitVariogram such as model and variogram such as alpha for anisotropy. Note that gstat does not fit anisotropic models and you will get a warning if you specify alpha. Nevertheless, plotting the empirical anisotropic variograms and comparing them to the variogram fitted to the entire dataset can be a useful EDA tool.

y For bivariate, another numeric vector whose variogram is computed.

map logical; if TRUE, and cutoff and width are given, a variogram map is returned. This requires package sp. Alternatively, a map can be passed, of class SpatialDataFrameGrid (see sp docs)

width the width of subsequent distance intervals into which data point pairs are grouped for semivariance estimates

cutoff spatial separation distance up to which point pairs are included in semivariance estimates; as a default, the length of the diagonal of the box spanning the data is divided by three.

Value

An autofitVariogram object.
Index

* Downstream analyses of univariate spatial results
  clusterCorrelograms, 18
  clusterMoranPlot, 19
  clusterVariograms, 20

* Extensibility
  SFEMethod, 59

* Internal spatial statistics functions
  multispati_rsp, 30
  variogram-internal, 66

* Non-spatial plotting
  ElbowPlot, 23
  plotColDataFreqpoly, 33
  plotColDataHistogram, 35
  plotDimLoadings, 43

* Plot spatial analysis results
  moranPlot, 28
  plotCorrelogram, 39
  plotCrossVariogram, 41
  plotCrossVariogramMap, 42
  plotLocalResult, 45
  plotMoranMC, 50
  plotVariogram, 56
  plotVariogramMap, 58

* Spatial analysis results
  colFeatureData, 22
  getParams, 25

* Spatial plotting
  plotCellBin2D, 32
  plotColGraph, 37
  plotGeometry, 44
  plotSpatialFeature, 52
  spatialReducedDim, 62

* Spatial statistics
  calculateBivariate, 3
  calculateMultivariate, 6
  calculateUnivariate, 9
  listSFEMethods, 26
  moranBounds, 27

* datasets
  ditto_colors, 23
  .cross_variogram(variogram-internal), 66
  .cross_variogram_map(variogram-internal), 66
  .variogram, 5, 15
  .variogram(variogram-internal), 66
  .variogram_bv(variogram-internal), 66
  .variogram_map(variogram-internal), 66
  annotGeometries, 45
  annotGeometry, 15
  annotGeometryMoransI(calculateUnivariate), 9
  annotGeometryNames, 15
  annotGeometryUnivariate(calculateUnivariate), 9
  annotGraphNames, 15
  apply, 45
  args_not_check(SFEMethod), 59
  args_not_check(SFEMethod-method)
  (SFEMethod), 59

BiocParallelParam, 5, 8, 14
BlusterParam, 18, 20, 21

calculateBivariate, 3, 26, 41, 42
calculateBivariate, ANY-method
  (calculateBivariate), 3
calculateBivariate, SpatialFeatureExperiment-method
  (calculateBivariate), 3
calculateMoransI(calculateUnivariate), 9
calculateMoransI,ANY-method
  (calculateUnivariate), 9
calculateMoransI, SpatialFeatureExperiment-method
  (calculateUnivariate), 9
calculateMultivariate, 6, 26
INDEX

calculateMultivariate, ANY, character-method (calculateMultivariate), 6
calculateMultivariate, ANY, SFEMethod-method (calculateMultivariate), 6
calculateMultivariate, SpatialFeatureExperiment, ANY-method (calculateMultivariate), 6
calculateUnivariate, 9, 19, 26, 49
calculateUnivariate, ANY, character-method (calculateUnivariate), 9
calculateUnivariate, ANY, SFEMethod-method (calculateUnivariate), 9
calculateUnivariate, SpatialFeatureExperiment, ANY-method (calculateUnivariate), 9
colDataMoransI (calculateUnivariate), 9
colDataUnivariate (calculateUnivariate), 9
colGeometryMoransI (calculateUnivariate), 9
colGeometryNames, 5, 15, 29, 38, 40, 47, 51, 53, 57, 58, 64
colGeometryUnivariate (calculateUnivariate), 9
colGraphNames, 5, 8, 15
ditto_colors, 23
eigen, 27
ElbowPlot, 23, 43
facet_grid, 36
fun (SFEMethod), 59
fun, SFEMethod-method (SFEMethod), 59
geary, 16
geary.mc, 16
geary.test, 16, 40
geom_bin2d, 30, 33, 48, 54, 65
gem_density, 51
gem_density2d, 30
gem_freepoly, 51
gem_hex, 30, 33, 48, 54, 65
gem_histogram, 51
geometryFeatureData, 16
geometryFeatureData (colFeatureData), 22
getDivergeRange, 24
getParams, 25
globalG.test, 16
imageR, 47, 53, 64
info (SFEMethod), 59
info, SFEMethod-method (SFEMethod), 59
is_joint (SFEMethod), 59
is_joint, SFEMethod-method (SFEMethod), 59
is_local (SFEMethod), 59
layer(), 36
lee, 3
lee.mc, 3
lee.test, 3
listSFEMethods, 4, 7, 14, 26
listw2sparse, 27
localC, 8, 16
localC_perm, 16
localG, 16
localG_perm, 16
localmoran, 16
localmoran_bv, 3
localmoran_perm, 16
localResult, 16
localResultAttrs, 47
localResultFeatures, 46
localResultNames, 46
localResults, 6, 16
LOSH, 16
LOSH.cs, 16
LOSH.mc, 16
metadata, 22
moran, 16
moran.mc, 16
moran.plot, 16
moran.test, 16, 40
moranBounds, 27
moranPlot, 28, 49
multi_listw2sparse, 32
multispati_rsp, 8, 30
p.adjust, 40, 61
p.adjust.methods, 5, 8, 14
p.adjustSP, 5, 8, 14, 16
plotAnnotGraph (plotColGraph), 37
plotCellBin2D, 32
plotColDataFreqpoly, 33
plotColDataHistogram, 33, 35
plotColGraph, 37
plotCorrelogram, 39
plotCrossVariogram, 41
plotCrossVariogramMap, 42
plotDimLoadings, 43
plotGeometry, 44
plotLocalResult, 45
plotMoranMC, 50
plotRowDataFreqpoly
  (plotColDataFreqpoly), 33
plotRowDataHistogram
  (plotColDataHistogram), 35
plotSpatialFeature, 49, 52, 65
plotVariogram, 56
plotVariogramMap, 58
reducedDim, 8
reducedDimFeatureData (colFeatureData), 22
reducedDimMoransI
  (calculateUnivariate), 9
reducedDimNames, 16, 18, 21, 22, 26, 40, 51, 57, 59
reducedDimUnivariate
  (calculateUnivariate), 9
reorganize_fun (SFEMethod), 59
reorganize_fun, SFEMethod-method (SFEMethod), 59
rowFeatureData (colFeatureData), 22
runBivariate, 48
runBivariate (calculateBivariate), 3
runMoransI (calculateUnivariate), 9
runMultivariate
  (calculateMultivariate), 6
runUnivariate, 39, 48, 49
runUnivariate (calculateUnivariate), 9
SFEMethod, 4, 7, 14, 26, 48, 59
SFEMethod-class (SFEMethod), 59
sp.correlogram, 16
spatialReducedDim, 62
use_graph (SFEMethod), 59
use_graph, SFEMethod-method (SFEMethod), 59
variogram, 67
variogram-internal, 66
wrap_plots, 33, 38, 45, 47, 48, 53, 54, 64, 65