Package 'nnSVG'

October 18, 2022

Version 1.0.4

Title Scalable identification of spatially variable genes in spatially-resolved transcriptomics data

Description Method for scalable identification of spatially variable genes (SVGs) in spatially-resolved transcriptomics data. The method is based on nearest-neighbor Gaussian processes and uses the BRISC algorithm for model fitting and parameter estimation. Allows identification and ranking of SVGs with flexible length scales across a tissue slide or within spatial domains defined by covariates. Scales linearly with the number of spatial locations and can be applied to datasets containing thousands or more spatial locations.

URL https://github.com/lmweber/nnSVG

BugReports https://github.com/lmweber/nnSVG/issues

License MIT + file LICENSE

Encoding UTF-8

biocViews Spatial, SingleCell, Transcriptomics, GeneExpression, Preprocessing

Depends R (>= 4.2)

Imports SpatialExperiment, SingleCellExperiment, SummarizedExperiment, BRISC, BiocParallel, Matrix, matrixStats, stats

VignetteBuilder knitr

Suggests BiocStyle, knitr, rmarkdown, STexampleData, scran, ggplot2, testthat

RoxygenNote 7.1.2

git_url https://git.bioconductor.org/packages/nnSVG

git_branch RELEASE_3_15

git_last_commit ef8699b

git_last_commit_date 2022-07-18

Date/Publication 2022-10-18

Author Lukas M. Weber [aut, cre] (<https://orcid.org/0000-0002-3282-1730>),
Stephanie C. Hicks [aut] (<https://orcid.org/0000-0002-7858-0231>)

Maintainer Lukas M. Weber <lukas.weber.edu@gmail.com>

R topics documented:

`filter_genes` *Preprocessing function to filter genes*

Description

Preprocessing function to filter low-expressed genes and/or mitochondrial genes for 'nnSVG'.

Usage

```
filter_genes(
  spe,
  filter_genes_ncounts = 3,
  filter_genes_pcspots = 0.5,
  filter_mito = TRUE
)
```

Arguments

`spe`	`SpatialExperiment`: Input data, assumed to be formatted as a `SpatialExperiment` object with an `assay` slot named `counts` containing raw expression counts.
`filter_genes_ncounts`	`numeric`: Filtering parameter for low-expressed genes. Filtering retains genes containing at least `filter_genes_ncounts` expression counts in at least `filter_genes_pcspots` percent of the total number of spatial locations (spots). Defaults: `filter_genes_ncounts` = 3, `filter_genes_pcspots` = 0.5, i.e. keep genes with at least 3 counts in at least 0.5 percent of spots. Set to NULL to disable.
`filter_genes_pcspots`	`numeric`: Second filtering parameter for low-expressed genes. Set to NULL to disable. See `filter_genes_ncounts` for details.
`filter_mito`	`logical`: Whether to filter out mitochondrial genes, identified by gene names starting with "MT" or "mt". This requires that the `rowData` slot of the input object contains a column named `gene_name`. Default = TRUE. Set to FALSE to disable.

Details

Preprocessing function to filter low-expressed genes and/or mitochondrial genes for 'nnSVG'.

This function can be used to filter out low-expressed genes and/or mitochondrial genes before additional preprocessing (calculating log-transformed normalized counts or deviance residuals) and running 'nnSVG'.

We use this function in the examples and vignettes in the 'nnSVG' package, and provide default filtering parameter values that are appropriate for 10x Genomics Visium data.

The use of this function is optional. Users can also perform filtering and preprocessing separately, and run nnSVG on a preprocessed `SpatialExperiment` object.

Value

Returns `SpatialExperiment` with filtered genes (rows) removed.

Examples

```
library(SpatialExperiment)
library(STexampleData)

# load example dataset from STexampleData package
spe <- Visium_humanDLPFC()

# preprocessing steps

# keep only spots over tissue
spe <- spe[, colData(spe)$in_tissue == 1]
dim(spe)

# filter low-expressed and mitochondrial genes
spe <- filter_genes(spe)
dim(spe)
```

---

nnSVG *nnSVG*

Description

Function to run 'nnSVG' method to identify spatially variable genes (SVGs) in spatially-resolved transcriptomics data.

Usage

```
nnSVG(
  spe,
  X = NULL,
  assay_name = "logcounts",
```

```
  n_neighbors = 10,
  order = "AMMD",
  n_threads = 1,
  BPPARAM = NULL,
  verbose = FALSE
)
```

Arguments

spe	SpatialExperiment: Input data, assumed to be formatted as a SpatialExperiment object with an assay slot containing either log-transformed normalized counts (e.g. from the scran package) or deviance residuals (e.g. from the scry package), and a spatialCoords slot containing spatial coordinates of the measurements.
X	numeric matrix: Optional design matrix containing columns of covariates per spatial location, e.g. known spatial domains. Number of rows must match the number of spatial locations. Default = NULL, which fits an intercept-only model.
assay_name	character: Name of the assay slot in the input object containing the preprocessed gene expression values. For example, logcounts for log-transformed normalized counts from the scran package, or binomial_deviance_residuals for deviance residuals from the scry package. Default = "logcounts".
n_neighbors	integer: Number of nearest neighbors for fitting the nearest-neighbor Gaussian process (NNGP) model with BRISC. The default value is 10, which we recommend for most datasets. Higher numbers (e.g. 15) may give slightly improved likelihood estimates in some datasets (at the expense of slower runtime), and smaller numbers (e.g. 5) will give faster runtime (at the expense of reduced performance). Default = 10.
order	character: Ordering scheme to use for ordering coordinates with BRISC. Default = "AMMD" for "approximate maximum minimum distance", which is recommended for datasets with at least 65 spots. For very small datasets (n <= 65), "Sum_coords" can be used instead. See BRISC documentation for details. Default = "AMMD".
n_threads	integer: Number of threads for parallelization. Default = 1. We recommend setting this equal to the number of cores available (if working on a laptop or desktop) or around 10 or more (if working on a compute cluster).
BPPARAM	BiocParallelParam: Optional additional argument for parallelization. This argument is provided for advanced users of BiocParallel for further flexibility for parallelization on some operating systems. If provided, this should be an instance of BiocParallelParam. For most users, the recommended option is to use the n_threads argument instead. Default = NULL, in which case n_threads will be used instead.
verbose	logical: Whether to display verbose output for model fitting and parameter estimation from BRISC. Default = FALSE.

Details

Function to run 'nnSVG' method to identify spatially variable genes (SVGs) in spatially-resolved transcriptomics data.

The 'nnSVG' method is based on nearest-neighbor Gaussian processes (Datta et al. 2016) and uses the BRISC algorithm (Saha and Datta 2018) for model fitting and parameter estimation. The method scales linearly with the number of spatial locations, and can be applied to datasets containing thousands or more spatial locations. For more details on the method, see our paper.

This function runs 'nnSVG' for a full dataset. The function fits a separate model for each gene, using parallelization with BiocParallel for faster runtime. The parameter estimates from BRISC (sigma.sq, tau.sq, phi) for each gene are stored in 'Theta' in the BRISC output.

'nnSVG' performs inference on the spatial variance parameter estimates (sigma.sq) using a likelihood ratio (LR) test against a simpler linear model without spatial terms (i.e. without tau.sq or phi). The estimated LR statistics can then be used to rank SVGs. P-values are calculated from the LR statistics using the asymptotic chi-squared distribution with 2 degrees of freedom, and multiple testing adjusted p-values are calculated using the Benjamini-Hochberg method. We also calculate an effect size, defined as the proportion of spatial variance, 'prop_sv = sigma.sq / (sigma.sq + tau.sq)'.

The function assumes the input is provided as a `SpatialExperiment` object containing an `assay` slot containing either log-transformed normalized counts (e.g. from the `scran` package) or deviance residuals (e.g. from the `scry` package), which have been preprocessed, quality controlled, and filtered to remove low-quality spatial locations.

Value

Returns output values as additional columns in the `rowData` slot of the input object, including spatial variance parameter estimates, likelihood ratio (LR) statistics, effect sizes (proportion of spatial variance), p-values, and multiple testing adjusted p-values.

Examples

```
library(SpatialExperiment)
library(STexampleData)
library(scran)

# load example dataset from STexampleData package
spe <- Visium_humanDLPFC()

# preprocessing steps

# keep only spots over tissue
spe <- spe[, colData(spe)$in_tissue == 1]

# skip spot-level quality control, since this has been performed previously
# on this dataset

# filter low-expressed and mitochondrial genes
spe <- filter_genes(spe)

# calculate log-transformed normalized counts using scran package
set.seed(123)
```

```
qclus <- quickCluster(spe)
spe <- computeSumFactors(spe, cluster = qclus)
spe <- logNormCounts(spe)

# select small number of genes for faster runtime in this example
set.seed(123)
ix <- sample(seq_len(nrow(spe)), 4)
spe <- spe[ix, ]

# run nnSVG
set.seed(123)
spe <- nnSVG(spe)

# show results
# for more details see extended example in vignette
rowData(spe)
```

Index