Package 'SpaNorm'

November 12, 2026

Title Spatially-aware normalisation for spatial transcriptomics data

Version 1.6.0

Description This package implements the spatially aware library size normalisation algorithm, SpaNorm. SpaNorm normalises out library size effects while retaining biology through the modelling of smooth functions for each effect. Normalisation is performed in a gene- and cell-/spot- specific manner, yielding library size adjusted data.

License GPL (>= 3)

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

biocViews Software, GeneExpression, Transcriptomics, Spatial, CellBiology

Depends R (>= 4.4)

Imports edgeR, ggplot2, Matrix, matrixStats, methods, rlang, scran, SeuratObject, SingleCellExperiment, SpatialExperiment, stats, SummarizedExperiment, S4Vectors, utils, BiocParallel, BiocSingular

Suggests testthat (>= 3.0.0), knitr, rmarkdown, prettydoc, pkgdown, covr, BiocStyle, scater, Seurat (>= 5.0.0), patchwork, ggforce, ggnewscale, tensorflow

URL https://bhuvad.github.io/SpaNorm

BugReports https://github.com/bhuvad/SpaNorm/issues

VignetteBuilder knitr

Config/testthat/edition 3

LazyData false

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

git_branch RELEASE_3_23

git_last_commit 05d4f4c

git_last_commit_date 2026-04-28

Repository Bioconductor 3.23

Date/Publication 2026-11-11

Author Dharmesh D. Bhuva [aut, cre] (ORCID:
<https://orcid.org/0000-0002-6398-9157>),
Agus Salim [aut] (ORCID: <https://orcid.org/0000-0003-3999-7701>),
Ahmed Mohamed [aut] (ORCID: <https://orcid.org/0000-0001-6507-5300>)

Maintainer Dharmesh D. Bhuva <dharmesh.bhuva@adelaide.edu.au>

Contents

fastSizeFactors . . . 2
filterGenes . . . 3
getSVGResults . . . 4
HumanDLPFC . . . 4
plotCovariate . . . 5
plotSpatial . . . 6
SpaNorm . . . 7
SpaNormFit . . . 10
SpaNormPCA . . . 11
SpaNormSVG . . . 13
topSVGs . . . 14

Index **15**

`fastSizeFactors` *Filter genes based on expression*

Description

This function computes the size factors using a fast but inaccurate approach. Size factors are computed using the direct estimate of library sizes (sum of all counts). Though fast, this approach does not cater for compositional biases in the data and therefore is less accurate than scran-based estimates.

Usage

```
fastSizeFactors(spe)

## S4 method for signature 'SpatialExperiment'
fastSizeFactors(spe)
```

Arguments

`spe`	a SpatialExperiment, Seurat, or SpatialFeatureExperiment object containing count data.

Value

a SpatialExperiment, Seurat, or SpatialFeatureExperiment, containing size factors in the 'sizeFactor' column of the column annotation.

Examples

```
data(HumanDLPFC)
HumanDLPFC <- fastSizeFactors(HumanDLPFC)
head(HumanDLPFC$sizeFactor)
```

`filterGenes` *Filter genes based on expression*

Description

This function removes genes that are very lowly expressed.

Usage

```
filterGenes(spe, prop = 0.1)

## S4 method for signature 'SpatialExperiment'
filterGenes(spe, prop = 0.1)

## S4 method for signature 'Seurat'
filterGenes(spe, prop = 0.1)
```

Arguments

`spe`	a SpatialExperiment, Seurat, or SpatialFeatureExperiment object containing count data.
`prop`	a numeric, indicating the proportion of loci/cells where the gene should be expressed (default is 0.1, i.e., genes should be expressed in at least 10% of the loci/cells).

Value

a logical vector encoding which genes should be kept for further analysis.

Examples

```
data(HumanDLPFC)
keep <- filterGenes(HumanDLPFC)
table(keep)
```

getSVGResults *Get SVG results from a SpatialExperiment object*

Description

Get SVG results from a SpatialExperiment object

Usage

```
getSVGResults(spe, stop_if_missing = TRUE)
```

Arguments

spe	a SpatialExperiment object
stop_if_missing	logical indicating whether to stop if SVG results are missing (default TRUE)

Value

A data frame containing SVG results if they exist

HumanDLPFC *Human dorsolateral prefrontal cortex (DLPFC) visium sample*

Description

Human DLPFC sample profiled using the 10x Visium platform. Lowly expressed genes were filtered out using the `filterGenes` function to retain genes expressed in at least 10% of samples. This version was obtained from the SpatialLIBD R/Bioconductor package.

Usage

```
data(HumanDLPFC)
```

Format

A SpatialExperiment containing region annotations in the `colData` column 'AnnotatedCluster'.

Source

SpatialLIBD R/Bioconductor package.

References

Maynard KR, Collado-Torres L, Weber LM, Uytingco C, Barry BK, Williams SR, Catallini JL, Tran MN, Besich Z, Tippani M, Chew J. Transcriptome-scale spatial gene expression in the human dorsolateral prefrontal cortex. Nature neuroscience. 2021 Mar;24(3):425-36.

plotCovariate *Diagnostic plot of predicted expression for a covariate*

Description

This function can be used to spatially visualise the library size, biology or batch specific effect modelled for each gene.

Usage

```
plotCovariate(spe, covariate = c("biology", "ls", "batch"), ...)
```

Arguments

spe	a SpatialExperiment object.
covariate	a character, specifying the type of covariate to be plot: "biology" (default), "ls" to plot the library size effect, and "batch" to plot the batch-specific effect.
...	additional parameters to be passed to the plotSpatial function.

Value

a ggplot2 object

Examples

```
library(SpatialExperiment)
library(ggplot2)

data(HumanDLPFC)

HumanDLPFC = SpaNorm(HumanDLPFC, sample.p = 0.05, df.tps = 2, tol = 1e-2)
# plot spatial region annotations
p1 <- plotCovariate(HumanDLPFC, covariate = "biology", colour = ENSG00000075624) +
  scale_colour_viridis_c(option = "F")
p1

p2 <- plotCovariate(HumanDLPFC, covariate = "ls", colour = ENSG00000075624) +
  scale_colour_viridis_c(option = "F")
p2
```

plotSpatial *Plot spatial transcriptomic annotations per spot*

Description

Plot spatial transcriptomic annotations per spot

Usage

```
plotSpatial(
  spe,
  what = c("annotation", "expression", "reduceddim"),
  ...,
  assay = SummarizedExperiment::assayNames(spe),
  dimred = SingleCellExperiment::reducedDimNames(spe),
  img = FALSE,
  crop = FALSE,
  imgAlpha = 1,
  rl = 1,
  circles = FALSE
)
```

Arguments

spe	a SpatialExperiment object.
what	a character, specifying what aspect should be plot, "annotation", "expression", or reduced dimension ("reduceddim").
...	additional aesthetic mappings or fixed parameters (e.g., shape = ".").
assay	a character or numeric, specifying the assay to plot (default is the first assay).
dimred	a character or numeric, specifying the reduced dimension to plot (default is the first reduced dimension).
img	a logical, indicating whether the tissue image (if present) should be plot (default = FALSE).
crop	a logical, indicating whether the image should be cropped to the spatial coordinates (default = FALSE).
imgAlpha	a numeric, specifying the alpha value for the image (default = 1).
rl	a numeric, specifying the relative size of the text (default = 1).
circles	a logical, indicating whether the spots should be plotted as circles (default = FALSE). This can be slower for large datasets.

Value

a ggplot2 object

Examples

```
library(SpatialExperiment)
library(ggplot2)

# load data
data(HumanDLPFC)

# plot spatial region annotations
p1 <- plotSpatial(HumanDLPFC, colour = AnnotatedCluster)
p1

# change colour scale
p1 + scale_colour_brewer(palette = "Paired")

# plot spatial expression
plotSpatial(HumanDLPFC, what = "expression", colour = ENSG00000075624) +
 scale_colour_viridis_c(option = "F")

# plot logcounts
logcounts(HumanDLPFC) <- log2(counts(HumanDLPFC) + 1)
plotSpatial(HumanDLPFC, what = "expression", colour = ENSG00000075624, assay = "logcounts") +
 scale_colour_viridis_c(option = "F")

# change point shape
plotSpatial(HumanDLPFC, what = "expression", colour = ENSG00000075624, assay = "logcounts", shape = 18) +
 scale_colour_viridis_c(option = "F")
```

SpaNorm *Spatially-dependent normalisation for spatial transcriptomics data*

Description

Performs normalisation of spatial transcriptomics data using spatially-dependent spot- and gene-specific size factors.

Usage

```
SpaNorm(
  spe,
  sample.p = 0.25,
  gene.model = c("nb"),
  adj.method = c("auto", "logpac", "pearson", "medbio", "meanbio"),
  scale.factor = 1,
  df.tps = 6,
  lambda.a = 1e-04,
  batch = NULL,
  tol = 1e-04,
  step.factor = 0.5,
```

```
  maxit.nb = 50,
  maxit.psi = 25,
  maxn.psi = 500,
  overwrite = FALSE,
  backend = c("auto", "cpu", "gpu"),
  verbose = TRUE,
  ...
)

## S4 method for signature 'SpatialExperiment'
SpaNorm(
  spe,
  sample.p = 0.25,
  gene.model = c("nb"),
  adj.method = c("auto", "logpac", "pearson", "medbio", "meanbio"),
  scale.factor = 1,
  df.tps = 6,
  lambda.a = 1e-04,
  batch = NULL,
  tol = 1e-04,
  step.factor = 0.5,
  maxit.nb = 50,
  maxit.psi = 25,
  maxn.psi = 500,
  overwrite = FALSE,
  backend = c("auto", "cpu", "gpu"),
  verbose = TRUE,
  ...
)

## S4 method for signature 'Seurat'
SpaNorm(
  spe,
  sample.p = 0.25,
  gene.model = c("nb"),
  adj.method = c("auto", "logpac", "pearson", "medbio", "meanbio"),
  scale.factor = 1,
  df.tps = 6,
  lambda.a = 1e-04,
  batch = NULL,
  tol = 1e-04,
  step.factor = 0.5,
  maxit.nb = 50,
  maxit.psi = 25,
  maxn.psi = 500,
  overwrite = FALSE,
  backend = c("auto", "cpu", "gpu"),
  verbose = TRUE,
```

```
  ...
)
```

Arguments

spe	a SpatialExperiment or Seurat object, with the count data stored in 'counts' or 'data' assays respectively.
sample.p	a numeric, specifying the (maximum) proportion of cells/spots to sample for model fitting (default is 0.25).
gene.model	a character, specifying the model to use for gene/protein abundances (default 'nb'). This should be 'nb' for count based datasets.
adj.method	a character, specifying the method to use to adjust the data (default 'auto', see details)
scale.factor	a numeric, specifying the sample-specific scaling factor to scale the adjusted count.
df.tps	a numeric, of length 1 or 2, specifying the maximum degrees of freedom for the thin-plate spline for the biology and library size effects, respectively (default is 6, see details).
lambda.a	a numeric, of length 1 or 2, specifying the smoothing parameter for regularizing regression coefficients (default is 0.0001, see details). Actual lambda.a used is lambda.a * ncol(spe).
batch	a vector or numeric matrix, specifying the batch design to regress out (default NULL, representing no batch effects). See details for more information on how to define this variable.
tol	a numeric, specifying the tolerance for convergence (default is 1e-4).
step.factor	a numeric, specifying the multiplicative factor to decrease IRLS step by when log-likelihood diverges (default is 0.5).
maxit.nb	a numeric, specifying the maximum number of IRLS iteration for estimating NB mean parameters for a given dispersion parameter (default is 50).
maxit.psi	a numeric, specifying the maximum number of IRLS iterations to estimate the dispersion parameter (default is 25).
maxn.psi	a numeric, specifying the maximum number of cells/spots to sample for dispersion estimation (default is 500).
overwrite	a logical, specifying whether to force recomputation and overwrite an existing fit (default FALSE). Note that if df.tps, batch, lambda.a, or gene.model are changed, the model is recomputed and overwritten.
backend	a character, specifying the backend to use for computations (default 'auto', see details). If 'gpu', GPU-based computations are used if available, otherwise CPU-based computations are used.
verbose	a logical, specifying whether to show update messages (default TRUE).
...	other parameters fitting parameters.

Details

SpaNorm works by first fitting a spatial regression model for library size to the data. Normalised data can then be computed using various adjustment approaches. When a negative binomial gene-model is used, the data can be adjusted using the following approaches: 'logpac', 'pearson', 'med-bio', and 'meanbio'.

The `df.tps` parameter specifies the degrees of freedom for the thin-plate spline. If only 1 value is provided, it specifies the degrees of freedom of the biology with the degrees of freedom of the library size being half of that. If 2 values are provided, the first value specifies the degrees of freedom of the biology and the second value specifies the degrees of freedom of the library size. For rectangular tissues, df.tps specifies the degrees of freedom along the length, with the degrees of freedom along the width calculated ceiling(width / length * df.tps).

Similarly, the `lambda.a` parameter specifies the smoothing parameter for regularizing regression coefficients. If only 1 value is provided, it specifies the lambda.a for both the biology and library size functions. If 2 values are provided, the first value specifies the lambda.a for the biology and the second value specifies the lambda.a for the library size. Batch effects are not regularised.

Batch effects can be specified using the `batch` parameter. If this parameter is a vector, a design matrix will be created within the function using `model.matrix`. If a custom design is provided in the form of a numeric matrix, this should ideally be created using `model.matrix`. The batch matrix should be created with an intercept term. The SpaNorm function will automatically detect the intercept term and remove the relevant column. Alternatively, users can subset the model matrix to remove this column manually. Please note that the model formula should include the intercept term and that the intercept column should be subset out after.

Value

a SpatialExperiment or Seurat object with the adjusted data stored in 'logcounts' or 'data', respectively.

Examples

```
data(HumanDLPFC)

SpaNorm(HumanDLPFC, sample.p = 0.05, df.tps = 2, tol = 1e-2)
```

`SpaNormFit` *An S4 class to store a SpaNorm model fit*

Description

An S4 class to store a SpaNorm model fit

Usage

```
## S4 method for signature 'SpaNormFit'
x$name
```

Arguments

x	an object of class SpaNormFit.
name	a character, specifying the name of the slot to retrieve.

Value

Return value varies depending on method.

Slots

ngenes a numeric, specifying the number of genes in the dataset.

ncells a numeric, specifying the number of cells/spots in the dataset.

gene.model a character, specifying the gene-specific model to used (see getGeneModels()).

df.tps an integer, specifying the degrees of freedom to used for the thin plate spline.

sample.p a numeric, specifying the proportion of samples used to approximated the model.

lambda.a a numeric, specifying the shinkage parameter used.

batch a vector or matrix, specifying the batch design used (if any).

W a matrix, specifying the covariate matrix of the linear model.

alpha a matrix, specifying the coefficients of the linear model.

gmean a numeric, specifying the mean estimate for each gene in the linear model.

psi a numeric, specifying the over-dispersion parameter for each gene if a negative binomial model was used (or a vector of NAs if another gene model is used).

wtype a factor, specifying the covariate types of columns in the covariate matrix, W. These could be "biology", "ls", or "batch".

loglik a numeric, specifying the log-likelihood of the model at each external iteration.

sampling a factor, specifying the cells/spots used for dispersion estimation ('dispersion'), GLM fitting ('glm' and 'dispersion'), all other cells/spots ('all').

Examples

```
example(SpaNorm)
```

SpaNormPCA *GLM-based (SpaNorm) PCA*

Description

GLM-based PCA using the SpaNorm model. The null model is considered to consist of the library size effects, batch effects, and the gene mean. GLM-PCA is approximated by regressing the null model from the data, and performing PCA on the residuals (Pearson or deviance).

Usage

```
SpaNormPCA(
  spe,
  nsvgs = 3000,
  ncomponents = 50,
  svg.fdr = 1,
  BSPARAM = bsparam(),
  BPPARAM = SerialParam(),
  residuals = c("deviance", "pearson"),
  name = "PCA"
)

## S4 method for signature 'SpatialExperiment'
SpaNormPCA(
  spe,
  nsvgs = 3000,
  ncomponents = 50,
  svg.fdr = 1,
  BSPARAM = bsparam(),
  BPPARAM = SerialParam(),
  residuals = c("deviance", "pearson"),
  name = "PCA"
)
```

Arguments

spe	a SpatialExperiment or Seurat object, with the count data stored in 'counts' or 'data' assays respectively, and a SpaNorm model fit.
nsvgs	the number of SVGs to use for PCA.
ncomponents	the number of components to compute.
svg.fdr	the FDR threshold for SVG calling.
BSPARAM	a BiocSingularParam object specifying which algorithm should be used to perform the PCA.
BPPARAM	a BiocParallelParam object specifying whether the PCA should be parallelized.
residuals	the type of residuals to use for PCA. Either "deviance" (default) or "pearson".
name	the name of the reducedDim to store the PCA results.

Details

SpaNorm PCA works by using the SpaNorm model fit for data normalisation to approximate a GLM-based PCA as described in Townes et al. (Genome Biology, 2019). The model used for normalisation represents the library size effects and the gene mean. Regressing these covariates, we remain with the deviance or Pearson residuals, upon which PCA can be performed to approximate the GLM-PCA.

Value

a SpatialExperiment or Seurat object with PCA results. For SpatialExperiment objects, these are stored in the reducedDims.

Examples

```
library(SpatialExperiment)
library(ggplot2)

data(HumanDLPFC)

HumanDLPFC = SpaNorm(HumanDLPFC, sample.p = 0.05, df.tps = 2, tol = 1e-2)
HumanDLPFC = SpaNormSVG(HumanDLPFC)
HumanDLPFC = SpaNormPCA(HumanDLPFC)
reducedDims(HumanDLPFC)
```

SpaNormSVG *Model-based spatially variable gene (SVG) calling*

Description

Spatially variable gene (SVG) calling using the SpaNorm model.

Usage

```
SpaNormSVG(spe, backend = c("auto", "cpu", "gpu"), verbose = TRUE)

## S4 method for signature 'SpatialExperiment'
SpaNormSVG(spe, backend = c("auto", "cpu", "gpu"), verbose = TRUE)
```

Arguments

spe	a SpatialExperiment or Seurat object, with the count data stored in 'counts' or 'data' assays respectively, and a SpaNorm model fit.
backend	a character, specifying the backend to use for computations. Options are "auto" (default), "cpu", or "gpu". If "auto", it will use GPU if available, otherwise CPU.
verbose	a logical, specifying whether to show update messages (default TRUE).

Details

SpaNorm SVG calling works by using the SpaNorm model fit for data normalisation to perform a likelihood ratio test (LRT). The model used for normalisation is considered to be the full model. A second nested model is fit without the splines representing biology. These nested models are then compared using a LRT to identify genes where the splines representing biology contain strong signal.

Value

a SpatialExperiment or Seurat object with F-statistics, false discovery rates (FDRs). For SpatialExperiment objects, these are stored in the rowData.

Examples

```
library(SpatialExperiment)
library(ggplot2)

data(HumanDLPFC)

HumanDLPFC = SpaNorm(HumanDLPFC, sample.p = 0.05, df.tps = 2, tol = 1e-2)
HumanDLPFC = SpaNormSVG(HumanDLPFC)
head(rowData(HumanDLPFC))
```

---

topSVGs *Export top SVG results to a data frame*

Description

Export top SVG results to a data frame

Usage

```
topSVGs(spe, n = 10, fdr = 1)
```

Arguments

spe	a SpatialExperiment object with SVG results from SpaNormSVG.
n	a numeric, specifying the number of top SVGs to call.
fdr	a numeric, specifying the false discovery rate (FDR) threshold for calling SVGs.

Value

A data frame containing the top SVGs from F-test results including F-statistics, p-values and FDR.

Examples

```
library(SpatialExperiment)
library(ggplot2)

data(HumanDLPFC)

HumanDLPFC = SpaNorm(HumanDLPFC, sample.p = 0.05, df.tps = 2, tol = 1e-2)
HumanDLPFC = SpaNormSVG(HumanDLPFC)
topSVGs = topSVGs(HumanDLPFC, n = 10)
```

Index

∗ **datasets**
HumanDLPFC, 4
∗ **internal**
getSVGResults, 4
$,SpaNormFit-method *(*SpaNormFit*)*, 10

fastSizeFactors, 2
fastSizeFactors,SpatialExperiment-method *(*fastSizeFactors*)*, 2
filterGenes, 3
filterGenes,Seurat-method *(*filterGenes*)*, 3
filterGenes,SpatialExperiment-method *(*filterGenes*)*, 3

getSVGResults, 4

HumanDLPFC, 4

plotCovariate, 5
plotSpatial, *5*, 6

SpaNorm, 7
SpaNorm,Seurat-method *(*SpaNorm*)*, 7
SpaNorm,SpatialExperiment-method *(*SpaNorm*)*, 7
SpaNormFit, 10
SpaNormFit-class *(*SpaNormFit*)*, 10
SpaNormPCA, 11
SpaNormPCA,SpatialExperiment-method *(*SpaNormPCA*)*, 11
SpaNormSVG, 13
SpaNormSVG,SpatialExperiment-method *(*SpaNormSVG*)*, 13

topSVGs, 14