Package 'SPOTlight'

April 11, 2023

Version 1.2.0

Type Package

Title `SPOTlight`: Spatial Transcriptomics Deconvolution

Description `SPOTlight` provides a method to deconvolute spatial transcriptomics spots using a seeded NMF approach along with visualization tools to assess the results. Spatially resolved gene expression profiles are key to understand tissue organization and function. However, novel spatial transcriptomics (ST) profiling techniques lack single-cell resolution and require a combination with single-cell RNA sequencing (scRNA-seq) information to deconvolute the spatially indexed datasets. Leveraging the strengths of both data types, we developed SPOTlight, a computational tool that enables the integration of ST with scRNA-seq data to infer the location of cell types and states within a complex tissue. SPOTlight is centered around a seeded non-negative matrix factorization (NMF) regression, initialized using cell-type marker genes and non-negative least squares (NNLS) to subsequently deconvolute ST capture locations (spots).

Depends R (>= 4.1)

Imports ggplot2, NMF, Matrix, matrixStats, nnls, SingleCellExperiment, stats

Suggests BiocStyle, colorBlindness, ExperimentHub, DelayedArray, ggcorrplot, grid, igraph, jpeg, knitr, methods, png, rmarkdown, scater, scatterpie, scran, Seurat, SeuratObject, SpatialExperiment, SummarizedExperiment, S4Vectors, TabulaMurisSenisData, TENxVisiumData, testthat

biocViews SingleCell, Spatial, StatisticalMethod

License GPL-3

Encoding UTF-8

RoxygenNote 7.1.2

VignetteBuilder knitr

URL https://github.com/MarcElosua/SPOTlight

BugReports https://github.com/MarcElosua/SPOTlight/issues

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

git_branch RELEASE_3_16

git_last_commit e6d919d

git_last_commit_date 2022-11-01

Date/Publication 2023-04-10

Author Marc Elosua-Bayes [aut, cre],
Helena L. Crowell [aut]

Maintainer Marc Elosua-Bayes <elosua.marc@gmail.com>

R topics documented:

`data` *Synthetic single-cell, mixture and marker data*

Description

`mockSC/mockSP()` are designed to generate synthetic single-cell and spatial mixture data. These data are not meant to represent biologically meaningful use-cases, but are solely intended for use in examples, for unit-testing, and to demonstrate `SPOTlight`'s general functionality. Finally, `.get_mgs()` implements a statistically naive way to select markers from single-cell data; again, please don't use it in real life.

Usage

```
mockSC(ng = 200, nc = 50, nt = 3)

mockSP(x, ns = 100)

getMGS(x, n_top = 10)
```

Arguments

ng, nc, nt, ns	integer scalar specifying the number of genes, cells, types (groups) and spots to simulate.
x	Single cell experiment object ç
n_top	integer specifying the number of marker genes to extract for each cluster.

Value

- mockSC returns a SingleCellExperiment with rows = genes, columns = single cells, and cell metadata (colData) column type containing group identifiers.
- mockSP returns a SingleCellExperiment with rows = genes, columns = single cells, and cell metadata (colData) column type containing group identifiers.
- getMGS returns a data.frame with nt*n_top rows and 3 columns: gene and type (group) identifier, as well as the gene's weight = the proportion of counts accounted for by that type.

Examples

```
sce <- mockSC()
spe <- mockSP(sce)
mgs <- getMGS(sce)
```

---

plotCorrelationMatrix *Plot Correlation Matrix*

Description

This function takes in a matrix with the predicted proportions for each spot and returns a correlation matrix between cell types.

Usage

```
plotCorrelationMatrix(
  x,
  cor.method = c("pearson", "kendall", "spearman"),
  insig = c("blank", "pch"),
  colors = c("#6D9EC1", "white", "#E46726"),
  hc.order = TRUE,
  p.mat = TRUE,
  ...
)
```

Arguments

x	numeric matrix with rows = samples and columns = cell types Must have at least two rows and two columns.
cor.method	Method to use for correlation: c("pearson", "kendall", "spearman"). By default pearson.
insig	character, specialized insignificant correlation coefficients, "pch", "blank" (default). If "blank", wipe away the corresponding glyphs; if "pch", add characters (see pch for details) on corresponding glyphs.
colors	character vector with three colors indicating the lower, mid, and high color. By default c("#6D9EC1", "white", "#E46726").
hc.order	logical value. If TRUE, correlation matrix will be hc.ordered using hclust function.
p.mat	logical value. If TRUE (default), correlation significance will be used. If FALSE arguments sig.level, insig, pch, pch.col, pch.cex are invalid.
...	additional graphical parameters passed to ggcorrplot.

Value

ggplot object

Author(s)

Marc Elosua Bayes & Helena L Crowell

Examples

```
set.seed(321)
x <- replicate(m <- 25, runif(10, 0, 1))
rownames(x) <- paste0("spot", seq_len(nrow(x)))
colnames(x) <- paste0("type", seq_len(ncol(x)))

# The most basic example
plotCorrelationMatrix(x = x)

# Showing the non-significant correlatinos
plotCorrelationMatrix(x = x, insig = "pch")

# A more elaborated
plotCorrelationMatrix(
    x = x,
    hc.order = FALSE,
    type = "lower",
    outline.col = "lightgrey",
    method = "circle",
    colors = c("#64ccc9", "#b860bd", "#e3345d"))
```

plotImage *Plot JP(E)G/PNG/Raster/RGB images*

Description

This function takes in an image-related object - path to JP(E)G/PNG file, raster object, RGBarray. It returns a ggplot object with the selected image.

Arguments

x	A variety of objects can be passed: character string corresponding to an image file path, valid file types are JPG, JPEG and PNG. It can also take as input objects of class raster and RGB arrays. It can also take a SpatialExperiment or Seurat object from which the image will be extracted.
slice	Character string indicating which image slice to use when SpatialExperiment or Seurat objects are passed. By default uses the first slice available.

Value

ggplot object

Author(s)

Marc Elosua Bayes & Helena L Crowell

Examples

```
# Filename
path <- file.path(
  system.file(package = "SPOTlight"),
  "extdata/SPOTlight.png")
plotImage(x = path)
# array
png_img <- png::readPNG(path)
plotImage(png_img)
# Seurat Object
# library(SeuratData)
# so <- LoadData("stxBrain", type = "anterior1")
# plotImage(so)
# SpatialExperiment
```

`plotInteractions` *Plot group interactions*

Description

This function takes in a matrix with the predicted proportions for each spot and returns a heatmap `which = plotHeatmap` or a network graph `which = plotNetwork` to show which cells are interacting spatially.

Usage

```
plotInteractions(
  x,
  which = c("heatmap", "network"),
  metric = c("prop", "jaccard"),
  min_prop = 0,
  ...
)
```

Arguments

`x`	numeric matrix with rows = samples and columns = groups. Must have at least one row and column, and at least two columns.
`which`	character string specifying the type of visualization: one of "heatmap" or "network".
`metric`	character string specifying which metric to show: one of "prop" or "jaccard".
`min_prop`	scalar specifying the value above which a group is considered to be contributing to a given sample. An interaction between groups i and j is counted for sample s only when both x[s, i] and x[s, j] fall above `min_prop`.
`...`	additional graphical parameters passed to `plot.igraph` when `which = "network"` (see `?igraph.plotting`).

Value

base R plot

Author(s)

Marc Elosua Bayes & Helena L Crowell

Examples

```
library(ggplot2)
mat <- replicate(8, rnorm(100, runif(1, -1, 1)))
# Basic example
plotInteractions(mat)
```

```
### heatmap ###
# This returns a ggplot object that can be modified as such
plotInteractions(mat, which = "heatmap") +
    scale_fill_gradient(low = "#f2e552", high = "#850000") +
    labs(title = "Interaction heatmap", fill = "proportion")

### Network ###
# specify node names
nms <- letters[seq_len(ncol(mat))]
plotInteractions(mat, which = "network", vertex.label = nms)

# or set column names instead
colnames(mat) <- nms
plotInteractions(mat, which = "network")

# pass additional graphical parameters for aesthetics
plotInteractions(mat,
    which = "network",
    edge.color = "cyan",
    vertex.color = "pink",
    vertex.label.font = 2,
    vertex.label.color = "maroon")
```

`plotSpatialScatterpie` *Spatial scatterpie*

Description

This function takes in the coordinates of the spots and the proportions of the cell types within each spot. It returns a plot where each spot is a piechart showing proportions of the cell type composition.

Usage

```
plotSpatialScatterpie(
  x,
  y,
  cell_types = colnames(y),
  img = FALSE,
  slice = NULL,
  scatterpie_alpha = 1,
  pie_scale = 0.4,
  ...
)
```

Arguments

x	Object containig the spots coordinates, it can be an object of class SpatialExperiment, Seurat, dataframe or matrix. For the latter two rownames should have the spot barcodes to match x. If a matrix it has to of dimensions nrow(y) x 2 where the columns are the x and y coordinates in that order.

y	Matrix or dataframe containing the deconvoluted spots. rownames need to be the spot barcodes to match to x.
cell_types	Vector of cell type names to plot. By default uses the column names of y.
img	Logical TRUE or FALSE indicating whether to plot the image or not. Objects of classes accepted by plotImage can also be passed and that image will be used. By default FALSE.
slice	Character string indicating which slice to plot if img is TRUE. By default uses the first image.
scatterpie_alpha	Numeric scalar to set the alpha of the pie charts. By default 1.
pie_scale	Numeric scalar to set the size of the pie charts. By default 0.4.
...	additional parameters to geom_scatterpie

Value

ggplot object

Author(s)

Marc Elosua Bayes & Helena L Crowell

Examples

```
set.seed(321)

# Coordinates
x <- replicate(2, rnorm(100))
rownames(x) <- paste0("spot", seq_len(nrow(x)))
colnames(x) <- c("imagecol", "imagerow")

# Proportions
y <- replicate(m <- 5, runif(nrow(x), 0, 1))
y <- prop.table(y, 1)

rownames(y) <- paste0("spot", seq_len(nrow(y)))
colnames(y) <- paste0("type", seq_len(ncol(y)))

(plt <- plotSpatialScatterpie(x = x, y = y))
```

---

plotTopicProfiles *Plot NMF topic profiles*

Description

This function takes in the fitted NMF model and returns the topic profiles learned for each cell facet = FALSE or cell type facet = TRUE. Ideal training will return all the cell from the same cell type to share a unique topic profile.

Usage

```
plotTopicProfiles(x, y, facet = FALSE, min_prop = 0.01, ncol = NULL)
```

Arguments

x	NMFfit object
y	vector of group labels. Should be of length ncol(coef(x)).
facet	logical indicating whether to stratify by group. If FALSE (default), weights will be the median across cells for each group (point = topic weight for a given cell type). If TRUE, cell-specific weights will be shown (point = topic weight of a given cell).
min_prop	scalar in [0,1]. When facet = TRUE, only cells with a weight > min_prop will be included.
ncol	integer scalar specifying the number of facet columns.

Value

ggplot object

Author(s)

Marc Elosua Bayes & Helena L Crowell

Examples

```
library(ggplot2)
x <- mockSC()
y <- mockSP(x)
z <- getMGS(x)

res <- SPOTlight(x, y,
    groups = x$type,
    mgs = z,
    group_id = "type",
    verbose = FALSE)

plotTopicProfiles(res[[3]], x$type, facet = TRUE)
plotTopicProfiles(res[[3]], x$type, facet = FALSE)
```

---

runDeconvolution *Run Deconvolution using NNLS model*

Description

This function takes in the mixture data, the trained model & the topic profiles and returns the proportion of each cell type within each mixture

Usage

```
runDeconvolution(
  x,
  mod,
  ref,
  scale = TRUE,
  min_prop = 0.01,
  verbose = TRUE,
  assay = "RNA",
  slot = "counts"
)
```

Arguments

x	mixture dataset. Can be a numeric matrix, SingleCellExperiment, SpatialExperiment or SeuratObjecy.
mod	object of class NMFfit as obtained from trainNMF.
ref	bject of class matrix containing the topic profiles for each cell type as obtained from trainNMF.
scale	logical specifying whether to scale single-cell counts to unit variance. This gives the user the option to normalize the data beforehand as you see fit (CPM, FPKM, ...) when passing a matrix or specifying the slot from where to extract the count data.
min_prop	scalar in [0,1] setting the minimum contribution expected from a cell type in x to observations in y. By default 0.
verbose	logical. Should information on progress be reported?
assay	if the object is of Class Seurat, character string specifying the assay from which to extract the expression matrix. By default "RNA". Ignore for the rest of x input classes.
slot	if the object is of Class Seurat, character string specifying the slot from which to extract the expression matrix. If the object is of class SpatialExperiment indicates matrix to use. By default "counts".

Value

base a list where the first element is an NMFfit object and the second is a matrix contatining the topic profiles learnt.

Author(s)

Marc Elosua Bayes & Helena L Crowell

Examples

```
set.seed(321)
# mock up some single-cell, mixture & marker data
sce <- mockSC(ng = 200, nc = 10, nt = 3)
```

```
spe <- mockSP(sce)
mgs <- getMGS(sce)

res <- trainNMF(
    x = sce,
    y = spe,
    groups = sce$type,
    mgs = mgs,
    weight_id = "weight",
    group_id = "type",
    gene_id = "gene")
# Run deconvolution
decon <- runDeconvolution(
    x = spe,
    mod = res[["mod"]],
    ref = res[["topic"]])
```

SPOTlight *Deconvolution of mixture using single-cell data*

Description

This is the backbone function which takes in single cell expression data to deconvolute spatial transcriptomics spots.

Usage

```
SPOTlight(
  x,
  y,
  groups = NULL,
  mgs,
  n_top = NULL,
  gene_id = "gene",
  group_id = "cluster",
  weight_id = "weight",
  hvg = NULL,
  scale = TRUE,
  model = c("ns", "std"),
  min_prop = 0.01,
  verbose = TRUE,
  assay = "RNA",
  slot = "counts",
  ...
)
```

Arguments

x, y	single-cell and mixture dataset, respectively. Can be a numeric matrix, SingleCellExperiment or SeuratObjecy.
groups	vector of group labels for cells in x. When x is a SingleCellExperiment or SeuratObject, defaults to colLabels and Idents(x), respectively.
mgs	data.frame or DataFrame of marker genes. Must contain columns holding gene identifiers, group labels and the weight (e.g., logFC, -log(p-value) a feature has in a given group.
n_top	integer scalar specifying the number of markers to select per group. By default NULL uses all the marker genes to initialize the model.
gene_id, group_id, weight_id	character specifying the column in mgs containing gene identifiers, group labels and weights, respectively.
hvg	character vector containing hvg to include in the model. By default NULL.
scale	logical specifying whether to scale single-cell counts to unit variance. This gives the user the option to normalize the data beforehand as you see fit (CPM, FPKM, ...) when passing a matrix or specifying the slot from where to extract the count data.
model	character string indicating which model to use when running NMF. Either "ns" (default) or "std".
min_prop	scalar in [0,1] setting the minimum contribution expected from a cell type in x to observations in y. By default 0.
verbose	logical. Should information on progress be reported?
assay	if the object is of Class Seurat, character string specifying the assay from which to extract the expression matrix. By default "RNA".
slot	if the object is of Class Seurat, character string specifying the slot from which to extract the expression matrix. If the object is of class SingleCellExperiment indicates matrix to use. By default "counts".
...	additional parameters.

Details

SPOTlight uses a Non-Negative Matrix Factorization approach to learn which genes are important for each cell type. In order to drive the factorization and give more importance to cell type marker genes we previously compute them and use them to initialize the basis matrix. This initialized matrices will then be used to carry out the factorization with the single cell expression data. Once the model has learn the topic profiles for each cell type we use non-negative least squares (NNLS) to obtain the topic contributions to each spot. Lastly, NNLS is again used to obtain the proportion of each cell type for each spot by finding the fitting the single-cell topic profiles to the spots topic contributions.

Value

a numeric matrix with rows corresponding to samples and columns to groups

Author(s)

Marc Elosua-Bayes & Helena L. Crowell

Examples

```
library(scater)
library(scran)
library(TabulaMurisSenisData)
library(TENxVisiumData)

# Get Visium data from 'TENxVisiumData'
spe <- MouseKidneyCoronal()

# Use symbols instead of Ensembl IDs as feature names
rownames(spe) <- rowData(spe)$symbol

# Load SCE data
sce <- TabulaMurisSenisDroplet(tissues = "Kidney")$Kidney

# Keep cells from 18m mice with clear cell type annotations
sce <- subset(sce, , age == "18m")
sce <- subset(sce, , ! free_annotation %in% c("nan", "CD45"))

# Get the top 3000 highly variable genes
sce <- logNormCounts(sce)
dec <- modelGeneVar(sce)
# For the purpose of the example we will set hvg to NULL but using
# 2000-3000 HVG is recommended.
# hvg <- getTopHVGs(dec, n = 3000)
hvg <- NULL
colLabels(sce) <- colData(sce)$free_annotation

# Get vector indicating which genes
# are neither ribosomal or mitochondrial
genes <- !grepl("^Rp[l|s]|Mt", rownames(sce))

# Compute marker genes
mgs <- scoreMarkers(sce, subset.row = genes)
mgs_ls <- lapply(names(mgs), function(i){
  x <- mgs[[i]]
  # Filter and keep relevant marker genes, those with AUC > 0.8
  x <- x[x$mean.AUC > 0.8, ]
  # Sort the genes from highest to lowest weight
  x <- x[order(x$mean.AUC, decreasing = TRUE), ]
  # Add gene and cluster id to the dataframe
  x$gene <- rownames(x)
  x$cluster <- i
  data.frame(x)
})
mgs_df <- do.call(rbind, mgs_ls)

# split cell indices by identity
```

```
idx <- split(seq(ncol(sce)), sce$free_annotation)
# downsample to at most 5 cells per identity
# Note that 3 is for example purpose, in real life scenarios n_cells
# should be ~100
n_cells <- 3
cs_keep <- lapply(idx, function(i) {
  n <- length(i)
  if (n < n_cells)
    n_cells <- n
  sample(i, n_cells)
})
sce <- sce[, unlist(cs_keep)]

res <- SPOTlight(
    x = counts(sce),
    y = counts(spe),
    groups = sce$free_annotation,
    mgs = mgs_df,
    hvg = hvg,
    weight_id = "mean.AUC",
    group_id = "cluster",
    gene_id = "gene")
```

`trainNMF` *train NMF model*

Description

This is the training function used by SPOTLight. This function takes in single cell expression data, trains the model and learns topic profiles for each cell type

Usage

```
trainNMF(
  x,
  y,
  groups = NULL,
  mgs,
  n_top = NULL,
  gene_id = "gene",
  group_id = "cluster",
  weight_id = "weight",
  hvg = NULL,
  model = c("ns", "std"),
  scale = TRUE,
  verbose = TRUE,
  assay = "RNA",
  slot = "counts",
  ...
)
```

Arguments

x, y	single-cell and mixture dataset, respectively. Can be a numeric matrix, SingleCellExperiment or SeuratObjecy.
groups	character vector of group labels for cells in x. When x is a SingleCellExperiment or SeuratObject, defaults to colLabels(x) and Idents(x), respectively. Make sure groups is not a Factor.
mgs	data.frame or DataFrame of marker genes. Must contain columns holding gene identifiers, group labels and the weight (e.g., logFC, -log(p-value) a feature has in a given group.
n_top	integer scalar specifying the number of markers to select per group. By default NULL uses all the marker genes to initialize the model.
gene_id, group_id, weight_id	character specifying the column in mgs containing gene identifiers, group labels and weights, respectively.
hvg	character vector containing hvg to include in the model. By default NULL.
model	character string indicating which model to use when running NMF. Either "ns" (default) or "std".
scale	logical specifying whether to scale single-cell counts to unit variance. This gives the user the option to normalize the data beforehand as you see fit (CPM, FPKM, ...) when passing a matrix or specifying the slot from where to extract the count data.
verbose	logical. Should information on progress be reported?
assay	if the object is of Class Seurat, character string specifying the assay from which to extract the expression matrix. By default "RNA".
slot	if the object is of Class Seurat, character string specifying the slot from which to extract the expression matrix. If the object is of class SingleCellExperiment indicates matrix to use. By default "counts".
...	additional parameters.

Value

base a list where the first element is an NMFfit object and the second is a matrix contatining the topic profiles learnt.

Author(s)

Marc Elosua Bayes & Helena L Crowell

Examples

```
set.seed(321)
# mock up some single-cell, mixture & marker data
sce <- mockSC(ng = 200, nc = 10, nt = 3)
spe <- mockSP(sce)
mgs <- getMGS(sce)
```

```
res <- trainNMF(
    x = sce,
    y = spe,
    groups = sce$type,
    mgs = mgs,
    weight_id = "weight",
    group_id = "type",
    gene_id = "gene")
# Get NMF model
res[["mod"]]
# Get topic profiles
res[["topic"]]
```

Index