Package 'gsdensity'

March 16, 2023

Type Package

Title Density-Based Gene Set Specificity Evaluation

Version 0.1.2

Description
Analysis frameworks for pathway heterogeneity and pathway activity evaluation in single-cell data, including scRNA-seq data and spatial genomics data.

License GPL-3

Encoding UTF-8

LazyData true

LazyDataCompression xz

RoxygenNote 7.2.1

Depends R (>= 4.0)

Imports
Seurat,CelliD,dnet,supraHex,Rgraphviz,infotheo,anticlust,multimode,philentropy,RANN,future.apply,MASS,reshape2,dp

NeedsCompilation no

Author Qingnan, Liang [aut, cre, cph],
Ken Chen [aut, cph]

Maintainer ``Qingnan, Liang" <qliang3@mdanderson.org>

Repository CRAN

Date/Publication 2023-03-16 11:10:02 UTC

R topics documented:

`ce` *cell embeddings for pbmc3k data*

Description

cell embeddings for pbmc3k data

Usage

```
ce
```

Format

A df

Source

created with pbmc3k data

Examples

```
data(ce)       #Lazy loading. Data becomes visible as soon as called
```

compute.cell.label *4.2. binarize the label propagation probability in the cell population; result in a binarized vector of cells with 'nagative' and 'positive' labels; 'positive' means that the cells are relevant to the gene set*

Description

4.2. binarize the label propagation probability in the cell population; result in a binarized vector of cells with 'nagative' and 'positive' labels; 'positive' means that the cells are relevant to the gene set

Usage

```
## S3 method for class 'cell.label'
compute(cell_vec)
```

Arguments

cell_vec	output of 'run.rwr'

Value

cell label of 'negative' or 'positive' for a given pathway

Examples

```
cells <- colnames(pbmc.mtx)
el <- gsdensity::compute.nn.edges(coembed = ce, nn.use = 300)
cv <- gsdensity::run.rwr(el = el,
                         gene_set = gene.set.list[["GOBP_B_CELL_ACTIVATION"]],
                         cells = cells)
cl <- compute.cell.label(cv)
```

compute.cell.label.df *similar to compute.cell.label; used when working with multiple gene sets*

Description

similar to compute.cell.label; used when working with multiple gene sets

Usage

```
## S3 method for class 'cell.label.df'
compute(cell_df)
```

Arguments

cell_df	output of 'run.rwr.list'

Value

cell labels of 'negative' or 'positive' for given pathways

Examples

```
cells <- colnames(pbmc.mtx)
el <- gsdensity::compute.nn.edges(coembed = ce, nn.use = 300)
cv <- gsdensity::run.rwr.list(el = el, gene_set = gene.set.list[1:30], cells = cells)
cl <- compute.cell.label.df(cv)
```

---

compute.db *this function is called by 'compute.kld' to aggregate the density contribution of each gene to each grid point, and then normalize the densities of grid points to 1.*

Description

this function is called by 'compute.kld' to aggregate the density contribution of each gene to each grid point, and then normalize the densities of grid points to 1.

Usage

```
## S3 method for class 'db'
compute(density.df)
```

Arguments

density.df	an intermediate object in 'compute.kld'

Value

distribution

compute.grid.coords *2. compute density of gene sets of interest 2.1 compute grid point coordinates*

Description

2. compute density of gene sets of interest 2.1 compute grid point coordinates

Usage

```
## S3 method for class 'grid.coords'
compute(coembed, genes.use, n.grids = 100)
```

Arguments

coembed	the result from compute.mca
genes.use	which genes to use; no default; can use genes based on the gene set selection or use rownames(object)
n.grids	number of grid points used for gene set density estimation; larger number is more accurate and slower; default is 100 (recommended to test 100 first)

Value

grid coordinates

Examples

```
compute.grid.coords(coembed = ce, genes.use = intersect(rownames(ce), rownames(pbmc.mtx)))
```

compute.jsd *5. compute the specificity of gene set when cell partition information is available; the information could be clustering, sample origins, or other conditions inspired by https://github.com/FloWuenne/scFunctions/blob/0d9ea609fa72210a151f7270e61bdee008e8fc88/*

Description

this function is called by compute.spec.single to calculate the similarity between two vectors

Usage

```
## S3 method for class 'jsd'
compute(x, y)
```

Arguments

x	x
y	y

Value

returns jsd_distance

`compute.kld` *2.2 compute KL-divergence (some are adapted from https://github.com/alexisvdb/singleCellHaystack/)*

Description

2.2 compute KL-divergence (some are adapted from https://github.com/alexisvdb/singleCellHaystack/)

Usage

```
## S3 method for class 'kld'
compute(
  coembed,
  genes.use,
  n.grids = 100,
  gene.set.list,
  gene.set.cutoff = 3,
  n.times = 100
)
```

Arguments

coembed	the result from compute.mca
genes.use	which genes to use; no default; can use genes based on the gene set selection or use rownames(object)
n.grids	number of grid points used for gene set density estimation; larger number is more accurate and slower; default is 100 (recommended to test 100 first) 'coembed', 'genes.use', 'n.grids' are passed to 'compute.grid.coords()'
gene.set.list	a list of gene sets; e.g., gene.set.list <- list(gene.set.a = c("A", "B", "C"), gene.set.b = c("a", "b", "c"))
gene.set.cutoff	gene sets with length less than this cutoff will not be used; the length is after the intersection of the gene set and genes.use
n.times	to evaluate how likely the gene set density is not caused by randomness, size-matched gene sets will be used to compute the background density distribution; This simulation will be done n.times; default is 100

Value

kl-divergence between given gene set and random gene sets

Examples

```
compute.kld(coembed = ce,
            genes.use = intersect(rownames(ce), rownames(pbmc.mtx)),
            gene.set.list = gene.set.list[1:10])
```

`compute.mca` *1. compute MCA embeddings*

Description

1. compute MCA embeddings

Usage

```
## S3 method for class 'mca'
compute(object, dims.use = 1:10, genes.use = rownames(object))
```

Arguments

`object`	a seurat object
`dims.use`	which mca dimensions to use; default is the first 10 dimensions
`genes.use`	which genes to use; default is all genes in the object

Value

returns a dataframe with cells as rows and mca coordinates as columns

Examples

```
pbmc <- Seurat::CreateSeuratObject(pbmc.mtx, meta.data = pbmc.meta)
ce <- compute.mca(object = pbmc)
```

`compute.nn.edges` *3. compute nearest neighbor graph for genes and cells This graph will be used for fetching the most relevant cells of a gene set*

Description

3. compute nearest neighbor graph for genes and cells This graph will be used for fetching the most relevant cells of a gene set

Usage

```
## S3 method for class 'nn.edges'
compute(coembed, nn.use = 300)
```

Arguments

`coembed`	the result from compute.mca
`nn.use`	the number of nearest neighbors for building the graph; default 300

Value

nearest neighbor graph (edges)

Examples

```
compute.nn.edges(coembed = ce)
```

`compute.spatial.kld` *6. find gene sets with spatial relevance*

Description

This function is to calculate how likely the cells relevant to a gene set is randomly distributed spatially

Usage

```
## S3 method for class 'spatial.kld'
compute(spatial.coords, weight_vec, n = 10, n.times = 20)
```

Arguments

spatial.coords	a data frame with each row as a cell and each column as a spatial coordinate (usually 2: x and y)
weight_vec	output of run.rwr
n	split the spatial map for local density estimation; n is the number of split for each dimension; for n = 10, the spatial map is split to n * n = 100 grids for the density estimation
n.times	the weight_vec is shuffled several times (n.times) to generate a background distribution (shuffled weights vs. equal weights) for statistical significance estimation (p.value); larger n.times will be more time-consuming and more accurate

Value

spatial kl-divergence

---

compute.spatial.kld.df

This function is to calculate how likely the cells relevant to multiple gene sets are randomly distributed spatially

---

Description

This function is to calculate how likely the cells relevant to multiple gene sets are randomly distributed spatially

Usage

```
## S3 method for class 'spatial.kld.df'
compute(spatial.coords, weight_df, n = 10, n.times = 20)
```

Arguments

spatial.coords	a data frame with each row as a cell and each column as a spatial coordinate (usually 2: x and y)
weight_df	output of run.rwr.list
n	split the spatial map for local density estimation; n is the number of split for each dimension; for n = 10, the spatial map is split to n * n = 100 grids for the density estimation
n.times	the same as n.times in function 'compute.spatial.kld'

Value

spatial kl-divergence for multiple gene sets

Examples

```
compute.spatial.kld.df(spatial.coords = coords.df, weight_df = weight_df)
```

`compute.spec` *This is to calculate the similarity between: 1. the label propagation probability of cells for gene sets and 2. the identify of cells in partitions*

Description

This is to calculate the similarity between: 1. the label propagation probability of cells for gene sets and 2. the identify of cells in partitions

Usage

```
## S3 method for class 'spec'
compute(cell_df, metadata, cell_group)
```

Arguments

`cell_df`	the output of run.rwr.list
`metadata`	a data frame with cell information (each row is a cell; usually object@meta.data)
`cell_group`	cell partition vector (usually a column name

Value

specificity of a pathway activity and other levels of cell annotations (e.g., cell type) in object@meta.data)

Examples

```
cells <- colnames(pbmc.mtx)
el <- gsdensity::compute.nn.edges(coembed = ce, nn.use = 300)
cv <- gsdensity::run.rwr.list(el = el, gene_set = gene.set.list[1:30], cells = cells)
jsd.df <- compute.spec(cell_df = cv,
                       metadata = pbmc.meta,
                       cell_group = "seurat_annotations"
                      )
```

`compute.spec.single` *This is to calculate the similarity between: 1. the label propagation probability of cells for gene sets and 2. the identify of cells in a certain partition This is called by 'compute.spec'; can also run by itself*

Description

This is to calculate the similarity between: 1. the label propagation probability of cells for gene sets and 2. the identify of cells in a certain partition This is called by 'compute.spec'; can also run by itself

Usage

```
## S3 method for class 'spec.single'
compute(vec, positive, cell_df)
```

Arguments

`vec`	cell partition vector (usually a column name in object@meta.data)
`positive`	the positive label, e.g. "disease" or "cluster_1"
`cell_df`	the output of run.rwr.list

Value

specificity of a pathway activity and other levels of cell annotations (e.g., cell type)

`coords.df` *mouse brain coords*

Description

mouse brain coords

Usage

```
coords.df
```

Format

A df

Source

created with brain data

Examples

```
data(coords.df)       #Lazy loading. Data becomes visible as soon as called
```

`el_nn_search` *this function is called by 'compute.nn.edges' to convert nearest neighbor identity matrix to edge list*

Description

this function is called by 'compute.nn.edges' to convert nearest neighbor identity matrix to edge list

Usage

```
el_nn_search(nn2_out)
```

Arguments

`nn2_out`	nn2_out

Value

returns edge list

`gene.set.list` *A gene set list containing multiple human GO gene sets*

Description

A gene set list containing multiple human GO gene sets

Usage

```
gene.set.list
```

Format

A list

Source

created based on msigdbr human C5 (biological process) gene sets

Examples

```
data(gene.set.list)       #Lazy loading. Data becomes visible as soon as called
```

`kde2d.weighted` *based on https://stat.ethz.ch/pipermail/r-help/2006-June/107405.html this is called by compute.spatial.kld to calculate the kernel density estimation in 2d space with each data point weighted.*

Description

based on https://stat.ethz.ch/pipermail/r-help/2006-June/107405.html this is called by compute.spatial.kld to calculate the kernel density estimation in 2d space with each data point weighted.

Usage

```
kde2d.weighted(x, y, w, h, n, lims = c(range(x), range(y)))
```

Arguments

x	x
y	y
w	w
h	h
n	n
lims	lims

Value

weighted kde2d estimation

`pbmc.meta` *pbmc3k meta*

Description

pbmc3k meta

Usage

```
pbmc.meta
```

Format

A df

Source

created with pbmc3k data

Examples

```
data(pbmc.meta)       #Lazy loading. Data becomes visible as soon as called
```

---

pbmc.mtx *pbmc3k matrix*

Description

pbmc3k matrix

Usage

```
pbmc.mtx
```

Format

A matrix

Source

created with pbmc3k data

Examples

```
data(pbmc.mtx)       #Lazy loading. Data becomes visible as soon as called
```

---

run.rwr *4.1 To calculate the label propagation probability for a gene set among cells; result in a vector (length = number of cells) reflecting the probability each cell is labeled during the propagation (relevance to the gene set)*

Description

4.1 To calculate the label propagation probability for a gene set among cells; result in a vector (length = number of cells) reflecting the probability each cell is labeled during the propagation (relevance to the gene set)

Usage

```
run.rwr(el, gene_set, cells, restart = 0.75)
```

Arguments

el	edge list; output of 'compute.nn.edges'
gene_set	a vector of genes of interest
cells	name of cells; usually the same as 'colnames(object)'
restart	the probability of the propagation to restart

Value

cell vector (representing gene set activity)

Examples

```
cells <- colnames(pbmc.mtx)
el <- gsdensity::compute.nn.edges(coembed = ce, nn.use = 300)
cv <- run.rwr(el = el, gene_set = gene.set.list[1], cells = cells)
```

---

run.rwr.list *result in a matrix (number of rows = number of cells; number of columns = number of gene sets) reflecting the probability each cell is labeled during the propagation (relevance to the gene set); same idea as run.rwr but with multiple gene sets*

Description

result in a matrix (number of rows = number of cells; number of columns = number of gene sets) reflecting the probability each cell is labeled during the propagation (relevance to the gene set); same idea as run.rwr but with multiple gene sets

Usage

```
run.rwr.list(el, gene_set_list, cells, restart = 0.75)
```

Arguments

el	edge list; output of 'compute.nn.edges'
gene_set_list	a list of gene sets
cells	name of cells; usually the same as 'colnames(object)'
restart	the probability of the propagation to restart

Value

activity of pathways in cells

Examples

```
cells <- colnames(pbmc.mtx)
el <- gsdensity::compute.nn.edges(coembed = ce, nn.use = 300)
cv <- run.rwr.list(el = el, gene_set = gene.set.list[1:3], cells = cells)
```

`sample.kld` *this function is called by 'compute.kld' to calculate the kl-divergence between sampled (background) gene set and the ref (all) gene set*

Description

this function is called by 'compute.kld' to calculate the kl-divergence between sampled (background) gene set and the ref (all) gene set

Usage

```
sample.kld(density.df, ref, len.gene.set)
```

Arguments

`density.df`	density.df
`ref`	ref
`len.gene.set`	len.gene.set

Value

returns random klds

`sample.spatial.kld` *this function is called by 'compute.spatial.kld' to calculate the kl-divergence between cell-weighted with shuffled weight vector and the ref (all cells, unweighted)*

Description

this function is called by 'compute.spatial.kld' to calculate the kl-divergence between cell-weighted with shuffled weight vector and the ref (all cells, unweighted)

Usage

```
sample.spatial.kld(weight_vec, spatial.coords, n, ref)
```

Arguments

weight_vec	weight_vec
spatial.coords	spatial.coords
n	n
ref	ref

Value

returns randomly sampled spatial klds for gene sets

---

seed.mat *4. compute label propagation from gene set to cells this function is to form a 'seed matrix' used by the dRWR function (dnet R package); the seed matrix is specifying which nodes are the sources for label propagation*

Description

4. compute label propagation from gene set to cells this function is to form a 'seed matrix' used by the dRWR function (dnet R package); the seed matrix is specifying which nodes are the sources for label propagation

Usage

```
seed.mat(gene_set, graph.use)
```

Arguments

gene_set	gene_set
graph.use	graph.use

Value

returns seed matrix

`seed.mat.list` *this function is used when more than one 'seed sets' will be used (when there are multiple gene sets of interest)*

Description

this function is used when more than one 'seed sets' will be used (when there are multiple gene sets of interest)

Usage

```
seed.mat.list(gene_set_list, graph.use)
```

Arguments

`gene_set_list`	gene_set_list
`graph.use`	graph.use

Value

returns seed matrix

`vectorized_pdist` *from an excellent post: https://www.r-bloggers.com/2013/05/pairwise-distances-in-r/ enhanced the speed this function is called by 'compute.kld' to quickly compute the distance between genes to grid points*

Description

from an excellent post: https://www.r-bloggers.com/2013/05/pairwise-distances-in-r/ enhanced the speed this function is called by 'compute.kld' to quickly compute the distance between genes to grid points

Usage

```
vectorized_pdist(A, B)
```

Arguments

`A`	matrix
`B`	matrix

Value

returns pairwise-distances

`weight_df` *mouse brain gene set activities*

Description

mouse brain gene set activities

Usage

```
weight_df
```

Format

A df

Source

created with brain data

Examples

```
data(weight_df)       #Lazy loading. Data becomes visible as soon as called
```

Index