Package ‘CellMixS’

March 11, 2024

Type Package

Title Evaluate Cellspecific Mixing

Version 1.18.0

Description CellMixS provides metrics and functions to evaluate batch effects, data integration and batch effect correction in single cell transcriptome data with single cell resolution. Results can be visualized and summarised on different levels, e.g. on cell, celltype or dataset level.

License GPL (>=2)

Imports BiocNeighbors, ggplot2, scater, viridis, cowplot, SummarizedExperiment, SingleCellExperiment, tidyr, magrittr, dplyr, gggridges, stats, purrr, methods, BiocParallel, BiocGenerics

Depends kSamples, R (>= 4.0)

biocViews SingleCell, Transcriptomics, GeneExpression, BatchEffect

BugReports https://github.com/almutlue/CellMixS/issues

URL https://github.com/almutlue/CellMixS

Encoding UTF-8

RoxygenNote 7.1.1

VignetteBuilder knitr

Suggests BiocStyle, knitr, rmarkdown, testthat, limma, Rtsne

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

git_branch RELEASE_3_18

git_last_commit f781a7c

git_last_commit_date 2023-10-24

Repository Bioconductor 3.18

Date/Publication 2024-03-11

Author Almut Lütge [aut, cre]

Maintainer Almut Lütge <almut.luetge@uzh.ch>
Description

**CellMixS** provides metrics and functions to evaluate batch effects, data integration and batch effect correction in single cell transcriptome data with single cell resolution. Results can be visualized and summarised on different levels, e.g. on cell, celltype or dataset level.

Details

In particular, **CellMixS** includes two main metrics: Cellspecific mixing scores to determine the probability of random mixing in each cell’s neighbourhood. It can be assesed via the `cms` function. Local Density Factor Differences to evaluate the effect of data integration methods on batch internal structures. It can be assesed via the `ldfDiff` function.

Author(s)

Almut Lütge <almut.luetge@uzh.ch>
Mark D Robinson <mark.robinson@imls.uzh.ch>
Description

Function to calculate a cellspecific mixing score (cms) of groups/batches.

Usage

.cmsCell(
  cell,
  group,
  knn,
  k_min = NA,
  batch_min = NULL,
  cell_min = 4,
  unbalanced = FALSE,
  sce
)

Arguments

cell Character. Name of the cell to calculate cms for. Needs to be one of rownames(knn).
group Character. Name of group/batch variable. Needs to be one of names(knn).
knn List with three elements. First "index" with indices of knn cells. Second "distance" with distances to knn cells. Third a slot named by group variable with group level of knn cells.
k_min Numeric. Minimum number of knn to include. Default is NA (see Details).
batch_min Numeric. Minimum number of cells per batch to include in to the AD test. If set neighbours will be included until batch_min cells from each batch are present.
cell_min Numeric. Minimum number of cells from each group to be included into the AD test. Should be > 4 to make 'ad.test' working.
unbalanced Boolean. If True neighbourhoods with only one batch present will be set to NA. This way they are not included into any summaries or smoothening.
sce A SingleCellExperiment object with the combined data.

Details

The cms function tests the hypothesis, that group-specific distance distributions of knn cells have the same underlying unspecified distribution. It performs Anderson-Darling tests as implemented in the kSamples package. In default the function uses all distances and group label defined in knn. If k_min is specified, the first local minimum of the overall distance distribution with at least k_min cells is used. This can be used to adapt to the local structure of the dataset e.g. prevent cells from a distinct different cluster to be included.
**.defineSubspace**

**Value**

A p.value as resulting from the ad.test.

**See Also**

ad.test, cms, .smoothCms

Other helper functions: .defineSubspace(), .filterKnn(), .filterLocMin(), .ldfKnn(), .smoothCms()

---

**Description**

Helper function for ldfSce and cms to define or recalculate the subspace for analysis.

**Usage**

```r
.defineSubspace(sce, assay_name, dim_red, n_dim)
```

**Arguments**

- **sce**: A SingleCellExperiment object with the data to define the subspace.
- **assay_name**: Character. Name of the assay to use for PCA. Only relevant if no existing 'dim_red' is provided. Must be one of `names(assays(sce))`.
- **dim_red**: Character. Name of embeddings to use as subspace.
- **n_dim**: Numeric. Number of subspace elements to include to define subspace.

**Details**

Function to determine the subspace for ldfDiff and cms. Checks whether the defined 'dim_red' is present. Only if no subspace is defined or present it will perform a PCA using runPCA. To calculate PCA counts defined in 'assay_name' are used.

**Value**

A matrix of cell embeddings with reduced dimensions as columns.

**See Also**

ldfSce, cms.

Other helper functions: .cmsCell(), .filterKnn(), .filterLocMin(), .ldfKnn(), .smoothCms()
.filterKnn

Description

.FilterKnn

Usage

.FilterKnn(knn_cell, batch_min, group, sce)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>knn_cell</td>
<td>Data frame with one column &quot;distance&quot; and one column named by the group variable. Rows correspond to the knn cells and do not need rownames.</td>
</tr>
<tr>
<td>batch_min</td>
<td>Numeric. Minimum number of cells per batch to include.</td>
</tr>
<tr>
<td>group</td>
<td>Character. Name of group/batch variable. Needs to be one of names(knn).</td>
</tr>
<tr>
<td>sce</td>
<td>A SingleCellExperiment object with the combined data.</td>
</tr>
</tbody>
</table>

Value

data.frame with two columns (index, distance) for filtered knn cells.

See Also

.cmsCell

Other helper functions: .cmsCell(), .defineSubspace(), .filterLocMin(), .ldfKnn(), .smoothCms()

---

.filterLocMin

Description

Function to filter knn by overall distance density distribution.

Usage

.filterLocMin(knn_cell, k_min)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>knn_cell</td>
<td>Data frame with one column &quot;distance&quot; and one column named by the group variable. Rows correspond to the knn cells and do not need rownames.</td>
</tr>
<tr>
<td>k_min</td>
<td>Numeric. Minimum number of Knn to include.</td>
</tr>
</tbody>
</table>
Details

Internal function to filter cells used for cms testing to come from a continuous overall density distribution function (similar to cluster definitions). ‘filterLocMin’ is only applied, if k-min is specified as parameter in `.cmsCell` or `cms`.

Value

data.frame with two columns (index, distance) for filtered knn cells.

See Also

`.cmsCell`

Other helper functions: `.cmsCell()`, `.defineSubspace()`, `.filterKnn()`, `.ldfKnn()`, `.smoothCms()`

Description

Calculates the Local Density Factor as implemented in the `DDoutlier` package with a predefined knn neighbourhood.

Usage

`.ldfKnn(dataset, knn_object, k = k, h = 1, c = 1)`

Arguments

- `dataset`: Matrix with cell embeddings with cells as rows and reduced dimensions as columns. Subspace to determine LDF in.
- `knn_object`: List with k-nearest neighbours (knn) as provided by `get.knn` from the FNN package. First element named "indices" contains indices of knn in dataset. Second element named "distance" contains distances of knn in dataset. Third element named "cell_name" contains rownames of knn in dataset.
- `k`: Numeric. Number of knn used. Should correspond to `knn_object`.
- `h`: Numeric. Bandwidth for kernel functions. The greater the bandwidth, the smoother kernels and lesser weight are put on outliers. Default is 1.
- `c`: Scaling constant for comparison of LDE to neighboring observations. Default is 1.

Details

LDF function modified from the `DDoutlier` package. Calculates a Local Density Estimate (LDE) and Local Density Factor (LDF) with a gaussian kernel. Modified to use a predefined knn neighbourhood. For `ldfSce` this is essential to determine LDF after data integration on the same set of cells.
Description

Performs weighted smoothening of cms scores

Usage

.smoothCms(knn, cms_raw, cell_names, k_min, k)

Arguments

knn List with three elements. First "index" with indices of knn cells. Second "distance" with distances to knn cells. Third a slot named by group variable with group level of knn cells.
cms_raw Matrix with raw cms scores for all cells specified in cell_names and knn. Column names need to be "cms".
cell_names Character vector with cell names corresponding to the rownames of the list elements in knn and rownames(cms_raw).
k_min Numeric. Minimum number of knn to include. Default is NA (see Details).
k Numeric. Number of k-nearest neighbours (knn) to use.

Details

Internal function to smooth cms scores. In a complete random setting cms scores are uniform distributed. To reduce the resulting random variance and enable visualization of local pattern cms scores can be smoothened assuming that within one region mixing is uniform. Generates smoothened cms scores using weighted means of cms scores within the k-nearest neighbourhood. Reciprocal distances are used as weights.

Value

matrix with two columns ("cms_smooth", "cms").
See Also

.cmsCell, cms

Other helper functions: .cmsCell(), .defineSubspace(), .filterKnn(), .filterLocMin(), .ldfKnn()

Description

Calculates cell-specific mixing scores based on euclidean distances within a subspace of integrated data.

Usage

cms(
  sce, k, group,
  dim_red = "PCA",
  assay_name = "logcounts",
  res_name = NULL,
  k_min = NA,
  smooth = TRUE,
  n_dim = 20,
  cell_min = 10,
  batch_min = NULL,
  unbalanced = FALSE,
  BPPARAM = SerialParam()
)

Arguments

sce A SingleCellExperiment object with the combined data.
k Numeric. Number of k-nearest neighbours (knn) to use.
group Character. Name of group/batch variable. Needs to be one of names(colData(sce))
dim_red Character. Name of embeddings to use as subspace for distance distributions. Default is "PCA".
assay_name Character. Name of the assay to use for PCA. Only relevant if no existing 'dim_red' is provided. Must be one of names(assays(sce)). Default is "logcounts".
res_name Character. Appendix of the result score’s name (e.g. method used to combine batches).
k_min Numeric. Minimum number of knn to include. Default is NA (see Details).
smooth Logical. Indicating if cms results should be smoothened within each neighbourhood using the weighted mean.
n_dim Numeric. Number of dimensions to include to define the subspace.
cell_min Numeric. Minimum number of cells from each group to be included into the AD test.
batch_min Numeric. Minimum number of cells per batch to include in to the AD test. If set neighbours will be included until batch_min cells from each batch are present.
unbalanced Boolean. If True neighbourhoods with only one batch present will be set to NA. This way they are not included into any summaries or smoothening.
BPPARAM A BiocParallelParam object specifying whether cms scores shall be calculated in parallel.

Details

The cms function tests the hypothesis, that group-specific distance distributions of knn cells have the same underlying unspecified distribution. It performs Anderson-Darling tests as implemented in the kSamples package. In default the function uses all distances and group label defined in knn. Alternative a density based neighbourhood can be defined by specifying k_min. In this case the first local minimum of the overall distance distribution with at least k_min cells is used. This can be used to adapt to the local structure of the dataset e.g. prevent cells from a different cluster to be included. Third the neighbourhood can be defined by batch occurrences. batch_min specifies the minimal number of cells from each batch that should be included to define the neighbourhood. If 'dim_red' is not defined or default cms will calculate a PCA using runPCA. Results will be appended to colData(sce). Names can be specified using res_name. If multiple cores are available cms scores can be calculated in parallel (does not work on Windows). Parallelization can be specified using BPPARAM.

Value

A SingleCellExperiment with cms (and cms_smooth) within colData.

References


See Also

.cmsCell, .smoothCms.

Examples

library(SingleCellExperiment)
sim_list <- readRDS(system.file("extdata/sim50.rds", package = "CellMixS"))
sce <- sim_list[[1]][[1:50]]
sce_cms <- cms(sce, k = 20, group = "batch", n_dim = 2)
**entropy**

**Description**

entropy

**Usage**

```r
entropy(
  sce,
  group,
  k,
  dim_red = "PCA",
  assay_name = "logcounts",
  n_dim = 10,
  res_name = NULL
)
```

**Arguments**

- **sce** SingleCellExperiment object, with the integrated data.
- **group** Character. Name of group/batch variable. Needs to be one of names(colData(sce)).
- **k** Numeric. Number of k-nearest neighbours (knn) to use.
- **dim_red** Character. Name of embeddings to use as subspace for distance distributions. Default is "PCA".
- **assay_name** Character. Name of the assay to use for PCA. Only relevant if no existing 'dim_red' is provided. Must be one of names(assays(sce)). Default is "logcounts".
- **n_dim** Numeric. Number of dimensions to include to define the subspace.
- **res_name** Character. Appendix of the result score’s name (e.g. method used to combine batches).

**Details**

The entropy function calculates the Shannon entropy of the group variable within each cell’s k-nearest neighbourhood. For balanced batches a Shannon entropy close to 1 indicates high randomness and mixing. For unbalanced batches entropy should be interpreted with caution, but could work as a relative measure in a comparative setting.

**Value**

A SingleCellExperiment with the entropy score within colData.
Examples

```r
library(SingleCellExperiment)
sim_list <- readRDS(system.file("extdata/sim50.rds", package = "CellMixS"))
sce <- sim_list[[1]][, c(1:15, 400:420, 16:30)]
sce <- entropy(sce, "batch", k = 20)
```

Description

Function to evaluate sc data integration providing a framework for different metrics. Metrics to evaluate mixing and preservance of the local/individual structure are provided.

Usage

```r
evalIntegration(
  metrics, 
  sce, 
  group, 
  dim_red = "PCA", 
  assay_name = "logcounts", 
  n_dim = 10, 
  res_name = NULL, 
  k = NULL, 
  k_min = NA, 
  smooth = TRUE, 
  cell_min = 10, 
  batch_min = NULL, 
  unbalanced = FALSE, 
  weight = TRUE, 
  k_pos = 5, 
  sce_pre_list = NULL, 
  dim_combined = dim_red, 
  assay_pre = "logcounts", 
  n_combined = 10, 
  BPPARAM = SerialParam()
)
```

Arguments

- `metrics` Character vector. Name of the metrics to apply. Must be one to all of 'cms', 'IdfDiff', 'isi', 'mixingMetric', 'localStructure', 'entropy'.
- `sce` SingleCellExperiment object, with the integrated data.
- `group` Character. Name of group/batch variable. Needs to be one of names(colData(sce)).
### evalIntegration

`evalIntegration` is a wrapper function for different metrics to understand results of integrated single cell data sets. In general there are metrics evaluating the *mixing* of datasets, that is, metrics that show whether there still is a bias for different datasets after integration. Furthermore there are metrics to evaluate how well the dataset internal structure has been retained, that is, metrics that show whether there has been (potentially biological) signal removed or noise added by integration.

- **dim_red**: Character. Name of embedding to use as subspace for distance distributions. Default is "PCA".
- **assay_name**: Character. Name of the assay to use for PCA. Only relevant if no existing 'dim_red' is provided. Must be one of names(assays(sce)). Default is "logcounts".
- **n_dim**: Numeric. Number of dimensions to include to define the subspace.
- **res_name**: Character vector. Appendix of the result score's name (e.g. method used to combine batches). Needs to have the same length as metrics or NULL.
- **k**: Numeric. Number of k-nearest neighbours (knn) to use.
- **k_min**: Numeric. Minimum number of knn to include (see cms). Relevant for metrics: 'cms'.
- **smooth**: Logical. Indicating if cms results should be smoothened within each neighbourhood using the weighted mean. Relevant for metric: 'cms'.
- **cell_min**: Numeric. Minimum number of cells from each group to be included into the AD test. Should be > 4. Relevant for metric: 'cms'.
- **batch_min**: Numeric. Minimum number of cells per batch to include in to the AD test. If set, neighbours will be included until batch_min cells from each batch are present. Relevant for metrics: 'cms'.
- **unbalanced**: Boolean. If TRUE, neighbourhoods with only one batch present will be set to NA. This way they are not included into any summaries or smoothening. Relevant for metric: 'cms'.
- **weight**: Boolean. If TRUE, batch probabilities to calculate the isi score are weighted by the mean distance of their cells towards the cell of interest. Relevant for metrics: 'isi'.
- **k_pos**: Numeric. Position of cell to be used as reference within mixing metric. See MixingMetric for details. Relevant for metric: 'mixingMetric'
- **sce_pre_list**: A list of SingleCellExperiment objects with single datasets before integration. Names should correspond to levels in colData(sce_combined)[,group]. Relevant for metric: 'ldfDiff'
- **dim_combined**: Character. Name of embeddings to use as subspace to calculate LDF after integration. Default is dim_red. Relevant for metric 'ldfDiff'.
- **assay_pre**: Character. Name of the assay to use for PCA. Only relevant if no existing 'dim_red' is provided. Must be one of names(assays(sce_pre)). Default is "logcounts". Relevant for metric 'ldfDiff'.
- **n_combined**: Number of PCs to use in original space. See LocalStruct for details. Relevant for metric 'localStructure'.
- **BPPARAM**: A BiocParallelParam object specifying whether cms scores shall be calculated in parallel. Relevant for metric: 'cms'.

### Details

EvalIntegration is a wrapper function for different metrics to understand results of integrated single cell data sets. In general there are metrics evaluating the *mixing* of datasets, that is, metrics that show whether there still is a bias for different datasets after integration. Furthermore there are metrics to evaluate how well the dataset internal structure has been retained, that is, metrics that show whether there has been (potentially biological) signal removed or noise added by integration.
**Value**

A SingleCellExperiment with the chosen metric’s score within colData.

**Metrics**

Here we provide the following metrics:

- **cms** Cellspecific Mixing Score. Metric that tests the hypothesis that group-specific distance distributions of knn cells have the same underlying unspecified distribution. The score can be interpreted as the data’s probability within an equally mixed neighbourhood according to the batch variable (see cms).

- **isi** Inverse Simpson Index. Metric that uses the Inverse Simpson’s Index to calculate the diversification within a specified neighbourhood. The Simpson index describes the probability that two entities are taken at random from the dataset and its inverse represent the effective number of batches in a neighbourhood. The inverse Simpson index has been proposed as a diversity score for batch mixing in single cell RNAseq by Korunsky et al. They provide a distance-based neighbourhood weightening in their Lisi package.

- **mixingMetric** Mixing Metric. Metric using the median position of the kth cell from each batch within its knn as a score. The lower the better mixed is the neighbourhood. We implemented an equivalent version to the one in the Seurat package (See MixingMetric and mixMetric.)

- **entropy** Shannon entropy. Metric calculating the Shannon entropy of the batch/group variable within each cell’s k-nearest neighbours. For balanced batches the entropy is closer to 1 the higher the variables randomness. For unbalanced batches entropy should only be used as a relative metric in a comparative setting (See entropy.)

- **ldfDiff** Local density factor differences. Metric that determines cell-specific changes in the Local Density Factor before and after data integration. A metric/difference close to 0 indicates no distortion of the previous structure (see ldfDiff).

- **localStructure** Local structure. Metric that compares the intersection of knn from the same batch before and after integration returning the average between all groups. The higher the more neighbours were reproduced after integration. Here we implemented an equivalent version to the one in the Seurat package (See LocalStruct and locStructure).

**References**


**Examples**

```r
library(SingleCellExperiment)
sim_list <- readRDS(system.file("extdata/sim50.rds", package = "CellMixS"))
sce <- sim_list[[1]][c(1:15, 300:320, 16:30)]
sce_batch1 <- sce[,colData(sce)$batch == "1"]
sce_batch2 <- sce[,colData(sce)$batch == "2"]
pre <- list("1" = sce_batch1, "2" = sce_batch2)
```
sce <- evalIntegration(metrics = c("cms", "mixingMetric", "isi", "entropy"), sce, "batch", k = 20)
sce <- evalIntegration("ldfDiff", sce, "batch", k = 20, sce_pre_list = pre)

isi

Description

isi

Usage

isi(
  sce,
  group,
  k,
  dim_red = "PCA",
  assay_name = "logcounts",
  n_dim = 10,
  weight = TRUE,
  res_name = NULL
)

Arguments

sce SingleCellExperiment object, with the integrated data.

group Character. Name of group/batch variable. Needs to be one of names(colData(sce)).

k Numeric. Number of k-nearest neighbours (knn) to use.

dim_red Character. Name of embeddings to use as subspace for distance distributions. Default is "PCA".

assay_name Character. Name of the assay to use for PCA. Only relevant if no existing 'dim_red' is provided.

n_dim Numeric. Number of dimensions to include to define the subspace.

weight Boolean. If TRUE, batch probabilities to calculate the isi score are weighted by the mean distance of their cells towards the cell of interest. Relevant for metrics: "isi".

res_name Character. Appendix of the result score's name (e.g. method used to combine batches).
Details

The isi function calculates the inverse Simpson index of the group variable within each cell's k-nearest neighbourhood. The Simpson index describes the probability that two entities are taken at random from the dataset and its inverse represent the effective number of batches in a neighbourhood. The inverse Simpson index has been proposed as a diversity score for batch mixing in single cell RNAseq by Korunsky et al. They provide a distance-based neighbourhood weightening in their Lisi package. Here, we provide a simplified way of weightening probabilities, if the weight argument is enabled.

Value

A SingleCellExperiment with the entropy score within colData.

References


Examples

library(SingleCellExperiment)
sim_list <- readRDS(system.file("extdata/sim50.rds", package = "CellMixS"))
sce <- sim_list[[1]][, c(1:15, 400:420, 16:30)]

sce <- isi(sce, "batch", k = 20)

ldfDiff

Description

 Determines cell-specific changes in the Local Density Factor before and after data integration.

Usage

ldfDiff(
    sce_pre_list,
    sce_combined,
    group,
    k = 75,
    dim_red = "PCA",
    dim_combined = dim_red,
    assay_pre = "logcounts",
    assay_combined = "logcounts",
    n_dim = 20,
    res_name = NULL
)
ldfDiff

Arguments

sce_pre_list  A list of SingleCellExperiment objects with single datasets before integration. Names should correspond to levels in colData(sce_combined)$group
sce_combined A SingleCellExperiment object with the combined data.
group Character. Name of group/batch variable that separates elements of sce_pre_list. Needs to be one of names(colData(sce_combined)).
k Numeric. Number of k-nearest neighbours (knn) to use.
dim_red Character. Name of embeddings to use as subspace to calculate LDF before integration. Default is "PCA".
dim_combined Character. Name of embeddings to use as subspace to calculate LDF after integration. Default is dim_red.
assay_pre Character. Name of the assay to use for PCA. Only relevant if no existing 'dim_red' is provided. Must be one of names(assays(sce_pre)). Default is "logcounts".
assay_combined Character. Name of the assay to use for PCA. Only relevant if no existing 'dim_red' is provided. Must be one of names(assays(sce_combined)). Default is "logcounts".
n_dim Numeric. Number of PCs to include to define subspaces.
res_name Character. Appendix of the result score's name (e.g. method used to combine batches). Used to specify result name for more than one run on the same input.

Details

The ldfDiff function calculates differences in LDF for each element in sce_pre_list and their corresponding cells in sce_combined using ldfSce. If 'dim_red' is not defined a PCA will be calculated using runPCA. In this case 'assay_pre' need to refer to the data slot that shall define the subspace. Similar refer 'dim-combined' and 'assay_combined' to the integrated subspace or to the resp. "corrected" count data slot. 'k' can be used to define the level of local structure that is tested. The smaller 'k' the more focus is on detailed structures, while a large k will tests overall changes.

Value

A SingleCellExperiment object.

References


See Also

ldfSce, ldfKnn.

Other ldf functions: ldfSce()
Examples

library(SingleCellExperiment)
sim_list <- readRDS(system.file("extdata/sim50.rds", package = "CellMixS"))
sce <- sim_list["batch20"][, c(1:50, 300:350)]
sce_batch1 <- sce[, colData(sce)$batch == "1"]
sce_batch2 <- sce[, colData(sce)$batch == "2"]
sce_pre_list <- list("1" = sce_batch1, "2" = sce_batch2)

sce_ldf <- ldfDiff(sce_pre_list, sce, k = 10, group = "batch",
                   dim_combined = "MNN", n_dim = 2)

Description

Determines cell-specific changes in the Local Density Factor before and after data integration for
one specific group.

Usage

ldfSce(
  sce_name,
  sce_pre_list,
  sce_combined,
  group,
  k = 75,
  dim_red = "PCA",
  dim_combined = dim_red,
  assay_pre = "logcounts",
  assay_combined = "logcounts",
  n_dim = 20
)

Arguments

  sce_name Character. Name of the element in sce_pre_list to calculate LDF differences in.
  sce_pre_list A list of SingleCellExperiment objects with single datasets before integration. Names need to correspond to levels in colData(sce_combined)$group and sce_name!!
  sce_combined A SingleCellExperiment object with combined data.
  group Character. Name of group/batch variable that separates elements of sce_pre_list. Needs to be one of names(colData(sce_combined)).
  k Numeric. Number of k-nearest neighbours (knn) to use.
The `ldfSce` function calculates differences in LDF for one specified element in `sce_pre_list` and their corresponding cells in `sce_combined`. If `dim_red` is not defined a PCA will be calculated using `runPCA`. In this case `assay_pre` need to refer to the data slot that shall define the subspace. Similar refer `dim_combined` and `assay_combined` to the integrated subspace or to the resp. "corrected" count data slot. `k` can be used to define the level of local structure that is tested. The smaller `k` the more focus is on detailed structures, while a large `k` will tests overall changes. K-nearest neighbours (knn) are determined in the subspaces before integration defined by `dim_red`. The same set of knn are used to determine LDF before and after integration.

Value

A data.frame with difference in LDF as column named "diff_ldf".

References


See Also

`ldfDiff`, `ldfKnn`.

Other ldf functions: `ldfDiff()`
locStructure

Description

locStructure

Usage

locStructure(
  sce,
  group,
  dim_combined,
  k = 100,
  dim_red = "PCA",
  assay_name = "logcounts",
  n_dim = 10,
  n_combined = 10,
  res_name = NULL
)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sce</td>
<td>SingleCellExperiment object, with the integrated data.</td>
</tr>
<tr>
<td>group</td>
<td>Character. Name of group/batch variable. Needs to be one of names(colData(sce)).</td>
</tr>
<tr>
<td>dim_combined</td>
<td>Character. Name of the reduced dimensional representation of the integrated data. Needs to be one of reducedDimNames(sce)).</td>
</tr>
<tr>
<td>k</td>
<td>Numeric. Number of k-nearest neighbours (knn) to use.</td>
</tr>
<tr>
<td>dim_red</td>
<td>Character. Name of embeddings to calculate neighbourhoods before integration. Default is &quot;PCA&quot;.</td>
</tr>
<tr>
<td>assay_name</td>
<td>Character. Name of the assay to use for PCA of the original (not integrated) data. Should not refer to &quot;corrected&quot; counts.</td>
</tr>
<tr>
<td>n_dim</td>
<td>Numeric. Number of dimensions to include for the original data.</td>
</tr>
<tr>
<td>n_combined</td>
<td>Numeric. Number of dimensions to include for the integrated data.</td>
</tr>
<tr>
<td>res_name</td>
<td>Character. Appendix of the result score’s name (e.g. method used to combine batches).</td>
</tr>
</tbody>
</table>

Details

The locStructure function implements the localStructure function from Seurat (See LocalStruct. For each group it calculates the k nearest neighbour within PCA space before integration and compares it to the knn within the reduced dimensional representation after integration. The score represents the proportion of overlapping neighbours. The LocalStruct function is based on the RunPCA function, while here runPCA is used. This can cause small deviance from the LocalStruct function, but overall these functions are equivalent.
mixMetric

Value

A SingleCellExperiment with the mixing metric within colData.

References


Examples

library(SingleCellExperiment)
sim_list <- readRDS(system.file("extdata/sim50.rds", package = "CellMixS"))
sce <- sim_list["batch20"][, c(1:50, 300:350)]
sce <- locStructure(sce, "batch", "MNN", k = 20, assay_name = "counts")

mixMetric

Description

mixMetric

Usage

mixMetric(
sce,
group,
k = 300,
dim_red = "PCA",
assay_name = "logcounts",
n_dim = 10,
k_pos = 5,
res_name = NULL
)

Arguments

sce SingleCellExperiment object, with the integrated data.
group Character. Name of group/batch variable. Needs to be one of names(colData(sce)).
k Numeric. Number of k-nearest neighbours (knn) to use.
dim_red Character. Name of embeddings to use as subspace for distance distributions. Default is "PCA".
assay_name Character. Name of the assay to use for PCA. Only relevant if no existing 'dim_red' is provided.
n_dim Numeric. Number of dimensions to include to define the subspace.
visCluster

\[ \begin{align*}
  \text{k_pos} & \quad \text{Position of the cell, which rank to use for scoring, defaults to 5.} \\
  \text{res_name} & \quad \text{Character. Appendix of the result score's name (e.g. method used to combine batches).}
\end{align*} \]

Details

The mixMetric function implements the mixingMetric function from Seurat (See MixingMetric. It takes the median rank of the '__k_pos__ neighbour from each batch as estimation for the data's entropy according to the batch variable. The same result can be assessed using the MixingMetric function and a seurat object from the __Seurat__ package.

Value

A SingleCellExperiment with the mixing metric within colData.

References


Examples

```
library(SingleCellExperiment)
sim_list <- readRDS(system.file("extdata/sim50.rds", package = "CellMixS"))
sce <- sim_list[[1]][, c(1:15, 400:420, 16:30)]
sce <- mixMetric(sce, "batch", k = 20)
```

visCluster

\[ \begin{align*}
  \text{visCluster} & \quad \text{visCluster}
\end{align*} \]

Description

Creates summary plots of metric scores for different groups/cluster.

Usage

```
visCluster(sce_cms, cluster_var, metric_var = "cms", violin = FALSE)
```

Arguments

\[ \begin{align*}
  \text{sce_cms} & \quad \text{A SingleCellExperiment object with the result scores (e.g. cms) to plot within colData(res_object).} \\
  \text{cluster_var} & \quad \text{Character. Name of the factor level variable to summarize metric scores on.} \\
  \text{metric_var} & \quad \text{Character Name of the metric scores to use. Default is "cms".} \\
  \text{violin} & \quad \text{A logical. If true violin plots are plotted, while the default (FALSE) will plot ridge plots.}
\end{align*} \]
Details

Plots summarized metric scores. This function is intended to visualize and compare metric scores among clusters or other dataset variables specified in 'cluster_var'.

Value

A ggplot object.

See Also

visIntegration

Other visualize functions: visGroup()

Examples

library(SingleCellExperiment)

sim_list <- readRDS(system.file("extdata/sim50.rds", package = "CellMixS"))
sce <- sim_list[[1]][, c(1:30, 300:320)]
sce_cms <- cms(sce, "batch", k = 20, n_dim = 2)
visCluster(sce_cms, "batch")
**Value**

A `ggplot` object.

**See Also**

`visOverview`, `visMetric`

Other visualize functions: `visCluster()`

**Examples**

```r
library(SingleCellExperiment)
sim_list <- readRDS(system.file("extdata/sim50.rds", package = "CellMixS"))
sce <- sim_list[[1]][[1]][, c(1:50, 300:350)]
visGroup(sce, "batch")
```

**Description**

Plot p-value histograms of metric score distributions

**Usage**

```r
visHist(
  res_object,
  metric = "cms",
  prefix = TRUE,
  n_col = 1,
  metric_prefix = NULL
)
```

**Arguments**

- `res_object` SingleCellExperiment object, matrix or data.frame. The SingleCellExperiment object should contain the result scores (e.g. cms) to plot in `colData(res_object)`. Matrix or data frame should have result scores in columns and cells in rows.
- `metric` Character vector. Specify names of `colData(sce)` to be plotted. Applies only if `res_object` is a SingleCellExperiment object. Default is 'cms'. If prefix is TRUE all columns starting with 'metric' will be plotted.
- `prefix` Boolean. Is 'metric' used to specify column’s prefix(true) or complete column names (False).
- `n_col` Numeric. Number of columns of the pval histogram.
- `metric_prefix` Former parameter to define prefix of the metric to be plotted. Will stop and ask for the new syntax.
Details

Plots metric score distribution similar to a pvalue histogram distribution. Without dataset-specific bias, cms scores should be approx. flat distributed. If `res_object` is a matrix or data.frame, it will create a histogram for each column. If `res_object` is a SingleCellExperiment object, it will create a histogram of all `colData(res_object)` that start with or are specified in `metric`.

Value

a ggplot object.

See Also

Other visualize metric functions: `visMetric()`, `visOverview()`

Examples

```r
library(SingleCellExperiment)
sim_list <- readRDS(system.file("extdata/sim50.rds", package = "CellMixS"))
sce <- sim_list[[1]][, c(1:50)]
sce_cms <- cms(sce, "batch", k = 20, n_dim = 2)
visHist(sce_cms)
```
**visMetric**

**Description**

Plot metric scores in a reduced dimensional plot.

**Usage**

```r
visMetric(sce_cms, metric_var = "cms", dim_red = "TSNE", log10_val = FALSE)
```
Arguments

sce_cms A SingleCellExperiment object with the result scores (e.g. cms) to plot within colData(res_object).

metric_var Character Name of the metric scores to use. Default is "cms".

dim_red Character. Name of embeddings to use as subspace for plotting. Default is "TSNE".

log10_val Logical. Indicating if -log10(metric) should be plotted.

Details

Plots a reduced dimension plot colored by metric scores. The dimension reduction embedding can be specified, but only tsne embeddings will automatically be computed using runTSNE. Embeddings from data integration methods (e.g. mnn.correct) can be used as long as they are present in reducedDimNames(sce).

Value

a ggplot object.

See Also

visOverview, visGroup

Other visualize metric functions: visHist(), visOverview()

Examples

library(SingleCellExperiment)
sim_list <- readRDS(system.file("extdata/sim50.rds", package = "CellMixS"))
sce <- sim_list[[1]][, c(1:30, 300:320)]
sce_cms <- cms(sce, "batch", k = 20, n_dim = 2)

visMetric(sce_cms)
Usage

```r
visOverview(
  sce_cms,
  group,
  metric = "cms",
  prefix = TRUE,
  dim_red = "TSNE",
  log10_val = FALSE,
  other_var = NULL,
  metric_prefix = NULL
)
```

Arguments

- `sce_cms`: A `SingleCellExperiment` object with the result scores (e.g. `cms`) to plot in `colData(sce_cms)`.
- `group`: Character. Name of group/batch variable. Needs to be one of `names(colData(sce))`.
- `metric`: Character vector. Specify names of `colData(sce)` to be plotted. Applies only if `res_object` is a `SingleCellExperiment` object. Default is 'cms'. If prefix is TRUE all columns starting with 'metric' will be plotted.
- `prefix`: Boolean. Is 'metric' used to specify column's prefix(true) or complete column names (False).
- `dim_red`: Character. Name of embeddings to use as subspace for plotting. Default is "TSNE".
- `log10_val`: Logical. Indicating if -log10(metric) should be plotted.
- `other_var`: Character string. Name(s) of other variables to be plotted asided. Need correspond to one of `colData(sce)`.
- `metric_prefix`: Former parameter to define prefix of the metric to be plotted. Will stop and ask for the new syntax.

Details

Plots reduced dimensions of cells colored by group variable and metric score. If 'red_dim' is not defined in `reducedDimNames(sce)` a tsne is calculated using `runTSNE`. Other color label as celltype label or smoothened scores can be plotted aside. Embeddings from data integration methods (e.g. `mnn.correct`) can be used if they are specified in `reducedDimNames(sce)`.

Value

A `ggplot` object.

See Also

- `visMetric`, `visGroup`

Other visualize metric functions: `visHist()`, `visMetric()`
Examples

library(SingleCellExperiment)
sim_list <- readRDS(system.file("extdata/sim50.rds", package = "CellMixS"))
sce <- sim_list[[1]][, c(1:30, 300:330)]
sce_cms <- cms(sce, "batch", k = 20, n_dim = 2)

visOverview(sce_cms, "batch", other_var = "batch")
Index

* cms functions
cms, 8

* helper functions
  .cmsCell, 3
  .defineSubspace, 4
  .filterKnn, 5
  .filterLocMin, 5
  .ldfKnn, 6
  .smoothCms, 7

* ldf functions
  ldfDiff, 15
  ldfSce, 17

* visualize functions
  visCluster, 21
  visGroup, 22
  visIntegration, 24

* visualize metric functions
  visHist, 23
  visMetric, 25
  visOverview, 26
  .cmsCell, 3, 4–9
  .defineSubspace, 4, 4, 5–8
  .filterKnn, 4, 5, 6–8
  .filterLocMin, 4, 5, 5, 7, 8
  .ldfKnn, 4–6, 6, 8, 16, 18
  .smoothCms, 4–7, 7, 9

ad.test, 4

BiocParallelParam, 9, 12

CellMixS-package, 2
cms, 2, 4, 6, 8, 8, 12, 13

entropy, 10, 13
evalIntegration, 11

isi, 14

ldfDiff, 2, 13, 15, 18
ldfSce, 4, 6, 7, 16, 17

LocalStruct, 12, 13, 19
locStructure, 13, 19

MixingMetric, 12, 13, 21
mixMetric, 13, 20

RunPCA, 19
runPCA, 19

visCluster, 21, 23, 25
visGroup, 22, 22, 26, 27
visHist, 23, 26, 27
visIntegration, 22, 24
visMetric, 23, 24, 25, 27
visOverview, 23, 24, 26, 26