

# Package ‘CellMixS’

March 17, 2025

**Type** Package

**Title** Evaluate Cellspecific Mixing

**Version** 1.22.0

**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, scatter, viridis, cowplot, SummarizedExperiment, SingleCellExperiment, tidy, magrittr, dplyr, ggridges, 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\_20

**git\_last\_commit** b431578

**git\_last\_commit\_date** 2024-10-29

**Repository** Bioconductor 3.20

**Date/Publication** 2025-03-17

**Author** Almut Lütge [aut, cre]

**Maintainer** Almut Lütge <almut.lue@gmail.com>

## Contents

CellMixS-package . . . . .	2
.cmsCell . . . . .	3
.defineSubspace . . . . .	4
.filterKnn . . . . .	5
.filterLocMin . . . . .	5
.ldfKnn . . . . .	6
.smoothCms . . . . .	7
cms . . . . .	8
entropy . . . . .	9
evalIntegration . . . . .	10
isi . . . . .	13
ldfDiff . . . . .	14
ldfSce . . . . .	16
locStructure . . . . .	18
mixMetric . . . . .	19
visCluster . . . . .	20
visGroup . . . . .	21
visHist . . . . .	22
visIntegration . . . . .	23
visMetric . . . . .	24
visOverview . . . . .	25
<b>Index</b>	<b>27</b>

---

CellMixS-package

*Toolbox to explore batch effects and data integration in scRNA data.*

---

### 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 assessed via the `cms` function. Local Density Factor Differences to evaluate the effect of data integration methods on batch internal structures. It can be assessed via the `ldfDiff` function.

### Author(s)

Almut Lütge <almut.luetge@uzh.ch>

Mark D Robinson <mark.robinson@imls.uzh.ch>

*.cmsCell**.cmsCell***Description**

Function to calculate a cell-specific mixing score (cms) of groups/batches.

**Usage**

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

**Arguments**

<code>cell</code>	Character. Name of the cell to calculate cms for. Needs to be one of rownames(knn).
<code>group</code>	Character. Name of group/batch variable. Needs to be one of names(knn).
<code>knn</code>	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.
<code>k_min</code>	Numeric. Minimum number of knn to include. Default is NA (see Details).
<code>batch_min</code>	Numeric. Minimum number of cells per batch to include in to the AD test. If set neighbours will be included until <code>batch_min</code> cells from each batch are present.
<code>cell_min</code>	Numeric. Minimum number of cells from each group to be included into the AD test. Should be > 4 to make 'ad.test' working.
<code>unbalanced</code>	Boolean. If True neighbourhoods with only one batch present will be set to NA. This way they are not included into any summaries or smoothing.
<code>sce</code>	A <code>SingleCellExperiment</code> 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 `kmin` 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.

**Value**

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

**See Also**

[ad.test](#), [cms](#), [.smoothCms](#)

Other helper functions: [.defineSubspace\(\)](#), [.filterKnn\(\)](#), [.filterLocMin\(\)](#), [.ldfKnn\(\)](#), [.smoothCms\(\)](#)

---

<code>.defineSubspace</code>	<i>.defineSubspace</i>
------------------------------	------------------------

---

**Description**

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

**Usage**

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

**Arguments**

<code>sce</code>	A <code>SingleCellExperiment</code> object with the data to define the subspace.
<code>assay_name</code>	Character. Name of the assay to use for PCA. Only relevant if no existing 'dim_red' is provided. Must be one of <code>names(assays(sce))</code> .
<code>dim_red</code>	Character. Name of embeddings to use as subspace.
<code>n_dim</code>	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                      *.filterKnn*

---

**Description**

.filterKnn

**Usage**

.filterKnn(knn\_cell, batch\_min, group, sce)

**Arguments**

knn_cell	Data frame with one column "distance" and one column named by the group variable. Rows correspond to the knn cells and do not need rownames.
batch_min	Numeric. Minimum number of cells per batch to include.
group	Character. Name of group/batch variable. Needs to be one of names(knn).
sce	A SingleCellExperiment object with the combined data.

**Value**

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

**See Also**

[.cmsCell](#)

Other helper functions: [.cmsCell\(\)](#), [.defineSubspace\(\)](#), [.filterLocMin\(\)](#), [.ldfKnn\(\)](#), [.smoothCms\(\)](#)

---

.filterLocMin                      *.filterLocMin*

---

**Description**

Function to filter knn by overall distance density distribution.

**Usage**

.filterLocMin(knn\_cell, k\_min)

**Arguments**

knn_cell	Data frame with one column "distance" and one column named by the group variable. Rows correspond to the knn cells and do not need rownames.
k_min	Numeric. Minimum number of Knn to include.

**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\(\)](#)

---

[.ldfKnn](#)

*.ldfKnn*

---

**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 <code>get.knn</code> from the <code>FNN</code> 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 <code>knn_object</code> .
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.

**Value**

List with two elements "LDE" and "LDF".

**See Also**

[ldfSce](#)

Other helper functions: [.cmsCell\(\)](#), [.defineSubspace\(\)](#), [.filterKnn\(\)](#), [.filterLocMin\(\)](#), [.smoothCms\(\)](#)

---

.smoothCms                      *.smoothCms*

---

### 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. Col-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 weighed 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\(\)](#)

cms

cms

## 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 smoothed within each neighbourhood using the weighed 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 smoothing.
BPPARAM	A <a href="#">BiocParallelParam</a> 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**

Scholz, F. W. and Stephens, M. A. (1987). K-Sample Anderson-Darling Tests. *J. Am. Stat. Assoc.*

**See Also**

[.cmsCell](#), [.smoothCms](#).

**Examples**

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

sce_cms <- cms(sce, k = 20, group = "batch", n_dim = 2)
```

---

entropy

*entropy*

---

**Description**

entropy

**Usage**

```
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**

```
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)
```

---

evalIntegration

*evalIntegration*


---

**Description**

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

**Usage**

```
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', 'ldfDiff', '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)).
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 <a href="#">cms</a> ). Relevant for metrics: 'cms'.
smooth	Logical. Indicating if cms results should be smoothed 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 smoothing. Relevant for metrics: '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 <a href="#">MixingMetric</a> for details. Relevant for metric: 'mixingMetric'

sce_pre_list	A list of <code>SingleCellExperiment</code> objects with single datasets before integration. Names should correspond to levels in <code>colData(sce_combined)[,group]</code> . Relevant for metric: 'ldfDiff'
dim_combined	Character. Name of embeddings to use as subspace to calculate LDF after integration. Default is <code>dim_red</code> . 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 <code>names(assays(sce_pre))</code> . Default is "logcounts". Relevant for metric 'ldfDiff'.
n_combined	Number of PCs to use in original space. See <a href="#">LocalStruct</a> for details. Relevant for metric 'localStructure'.
BPPARAM	A <a href="#">BiocParallelParam</a> 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 weighting 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

Korsunsky I Fan J Slowikowski K Zhang F Wei K et. al. (2018). Fast, sensitive, and accurate integration of single cell data with Harmony. bioRxiv (preprint).

Stuart T Butler A Hoffman P Hafemeister C Papalexi E et. al. (2019) Comprehensive Integration of Single-Cell Data. Cell.

## Examples

```
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

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 Korsunsky et al. They provide a distance-based neighbourhood weighting in their Lisi package. Here, we provide a simplified way of weighting probabilities, if the weight argument is enabled.

### Value

A SingleCellExperiment with the entropy score within colData.

### References

Korsunsky I Fan J Slowikowski K Zhang F Wei K et. al. (2018). Fast, sensitive, and accurate integration of single cell data with Harmony. bioRxiv (preprint)

### 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

*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
  )

```

### Arguments

sce_pre_list	A list of <code>SingleCellExperiment</code> objects with single datasets before integration. Names should correspond to levels in <code>colData(sce_combined)\$group</code>
sce_combined	A <code>SingleCellExperiment</code> object with the combined data.
group	Character. Name of group/batch variable that separates elements of <code>sce_pre_list</code> . Needs to be one of <code>names(colData(sce_combined))</code> .
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 <code>dim_red</code> .
assay_pre	Character. Name of the assay to use for PCA. Only relevant if no existing 'dim_red' is provided. Must be one of <code>names(assays(sce_pre))</code> . 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 <code>names(assays(sce_combined))</code> . 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

Latecki, Longin Jan and Lazarevic, Aleksandar and Pokrajac, Dragoljub (2007). Outlier Detection with Kernel Density Functions. *Mach. Learn. Data Min. Pattern Recognit.*. Springer Berlin Heidelberg.

### 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)

```

---

*ldfSce**ldfSce*

---

**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**

<code>sce_name</code>	Character. Name of the element in <code>sce_pre_list</code> to calculate LDF differences in.
<code>sce_pre_list</code>	A list of <code>SingleCellExperiment</code> objects with single datasets before integration. Names need to correspond to levels in <code>colData(sce_combined)\$group</code> and <code>sce_name!!</code>
<code>sce_combined</code>	A <code>SingleCellExperiment</code> object with combined data.
<code>group</code>	Character. Name of group/batch variable that separates elements of <code>sce_pre_list</code> . Needs to be one of <code>names(colData(sce_combined))</code> .
<code>k</code>	Numeric. Number of k-nearest neighbours (knn) to use.
<code>dim_red</code>	Character. Name of embeddings to use as subspace to calculate LDF before integration. Default is "PCA".
<code>dim_combined</code>	Character. Name of embeddings to use as subspace to calculate LDF after integration. Default is <code>dim_red</code> .



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.

## Details

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 test 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

Latecki, Longin Jan and Lazarevic, Aleksandar and Pokrajac, Dragoljub (2007). Outlier Detection with Kernel Density Functions. Mach. Learn. Data Min. Pattern Recognit.. Springer Berlin Heidelberg.

## See Also

[ldfDiff](#), [.ldfKnn](#).

Other ldf functions: [ldfDiff\(\)](#)

## 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_pre_list <- list("1" = sce_batch1)

ldf_1 <- ldfSce("1", sce_pre_list, sce, k = 10, group = "batch",
dim_combined = "MNN", n_dim = 5)
```

---

locStructure	<i>locStructure</i>
--------------	---------------------

---

## 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

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

## 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.

## Value

A SingleCellExperiment with the mixing metric within colData.

## References

Stuart T Butler A Hoffman P Hafemeister C Papalexi E et. al. (2019) Comprehensive Integration of Single-Cell Data. Cell.

## 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

*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.
k_pos	Position of the cell, which rank to use for scoring, defaults to 5.
res_name	Character. Appendix of the result score's name (e.g. method used to combine batches).

**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**

Stuart T Butler A Hoffman P Hafemeister C Papalexi E et. al. (2019) Comprehensive Integration of Single-Cell Data. Cell.

**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

*visCluster*

---

**Description**

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

**Usage**

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

**Arguments**

<code>sce_cms</code>	A <code>SingleCellExperiment</code> object with the result scores (e.g. <code>cms</code> ) to plot within <code>colData(res_object)</code> .
<code>cluster_var</code>	Character. Name of the factor level variable to summarize metric scores on.
<code>metric_var</code>	Character Name of the metric scores to use. Default is <code>"cms"</code> .
<code>violin</code>	A logical. If true violin plots are plotted, while the default ( <code>FALSE</code> ) will plot ridge plots.

**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")
```

---

`visGroup`*visGroup*

---

**Description**

Plot group label in a reduced dimensional plot.

**Usage**

```
visGroup(sce, group, dim_red = "TSNE")
```

**Arguments**

<code>sce</code>	A <code>SingleCellExperiment</code> object.
<code>group</code>	Character. Name of group/batch variable. Needs to be one of <code>names(colData(sce))</code> .
<code>dim_red</code>	Character. Name of embeddings to use as subspace for plotting. Default is "TSNE".

**Details**

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

**Value**

a `ggplot` object.

**See Also**[visOverview](#), [visMetric](#)Other visualize functions: [visCluster\(\)](#)

**Examples**

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

visGroup(sce, "batch")
```

visHist

*visHist***Description**

Plot pvalue histograms of metric score distributions

**Usage**

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

**Arguments**

<code>res_object</code>	SingleCellExperiment object, matrix or data.frame. The SingleCellExperiment object should contain the result scores (e.g. cms) to plot in <code>colData(res_object)</code> . Matrix or data frame should have result scores in columns and cells in rows.
<code>metric</code>	Character vector. Specify names of <code>colData(sce)</code> 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.
<code>prefix</code>	Boolean. Is 'metric' used to specify column's prefix(true) or complete column names (False).
<code>n_col</code>	Numeric. Number of columns of the pval histogram.
<code>metric_prefix</code>	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**

```
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)
```

---

visIntegration

*visIntegration*


---

**Description**

Creates a summary plot of metric scores (for different integration methods).

**Usage**

```
visIntegration(
  res_object,
  metric = "cms",
  prefix = TRUE,
  violin = FALSE,
  metric_name = "metric",
  metric_prefix = NULL
)
```

**Arguments**

res_object	SingleCellExperiment object, list, matrix or data.frame. The SingleCellExperiment object should contain the result scores (cms) to compare within colData(res_object). List, matrix or data frame should have result scores in list elements resp. columns.
metric	Character vector. Specify names of colData(sce) to be compared. Applies only if 'res_object' is a SingleCellExperiment object. Default is 'cms'. If prefix is TRUE all columns starting with 'metric' will be compared and plotted.
prefix	Boolean. Is 'metric' used to specify column's prefix(true) or complete column names (False).
violin	A logical. If true violin plots are plotted, while the default (FALSE) will plot ridge plots.
metric_name	Character. Name of the score metric.
metric_prefix	Former parameter to define prefix of the metric to be plotted. Will stop and ask for the new syntax.

**Details**

Plots summarized cms scores from an `SingleCellExperiment` object, list or dataframe. This function is intended to visualize and compare different methods and views of the same dataset, not to compare different datasets.

**Value**

a ggplot object.

**See Also**

[visCluster](#), [ggridges](#)

**Examples**

```
library(SingleCellExperiment)

sim_list <- readRDS(system.file("extdata/sim50.rds", package = "CellMixS"))

sce <- sim_list[["batch20"]][, c(1:30,300:320)]
sce_mnn <- cms(sce,"batch", k = 20, dim_red = "MNN", res_name = "MNN",
n_dim = 2)

visIntegration(sce_mnn, metric = "cms.", violin = TRUE)
```

---

visMetric

*visMetric*


---

**Description**

Plot metric scores in a reduced dimensional plot.

**Usage**

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

**Arguments**

sce_cms	A <code>SingleCellExperiment</code> object with the result scores (e.g. cms) to plot within <code>colData(res_object)</code> .
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 $-\log_{10}(\text{metric})$ 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)
```

---

visOverview

*visOverview*

---

**Description**

Plot an overview of metric results, group label and any colData variable in a reduced dimensional representation.

**Usage**

```
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 $-\log_{10}(\text{metric})$ should be plotted.
other_var	Character string. Name(s) of other variables to be plotted asided. Need correspond to one of <code>colData(sce)</code> .
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 smoothed scores can be plotted asided. 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, 14
  - ldfSce, 16
- \* **visualize functions**
  - visCluster, 20
  - visGroup, 21
  - visIntegration, 23
- \* **visualize metric functions**
  - visHist, 22
  - visMetric, 24
  - visOverview, 25
- .cmsCell, 3, 4–7, 9
- .defineSubspace, 4, 4, 5–7
- .filterKnn, 4, 5, 6, 7
- .filterLocMin, 4, 5, 5, 6, 7
- .ldfKnn, 4–6, 6, 7, 15, 17
- .smoothCms, 4–6, 7, 9
  
- ad.test, 4
  
- BiocParallelParam, 8, 12
  
- CellMixS-package, 2
- cms, 2, 4, 5, 7, 8, 11, 12
  
- entropy, 9, 12
- evalIntegration, 10
  
- isi, 13
  
- ldfDiff, 2, 12, 14, 17
- ldfSce, 4, 6, 15, 16
- LocalStruct, 12, 13, 18
- locStructure, 13, 18
  
- MixingMetric, 11, 12, 20
  
- mixMetric, 12, 19
  
- RunPCA, 18
- runPCA, 18
  
- visCluster, 20, 21, 24
- visGroup, 21, 21, 25, 26
- visHist, 22, 25, 26
- visIntegration, 21, 23
- visMetric, 21, 23, 24, 26
- visOverview, 21, 23, 25, 25