Package 'SingleCellExperiment'

October 16, 2019

Version 1.6.0

Date 2018-12-16

Title S4 Classes for Single Cell Data

Depends SummarizedExperiment

Imports S4Vectors, methods, BiocGenerics, utils, stats

Suggests testthat, BiocStyle, knitr, rmarkdown, scRNAseq, magrittr, Rtsne, Matrix

biocViews ImmunoOncology, DataRepresentation, DataImport, Infrastructure, SingleCell

Description Defines a S4 class for storing data from single-cell experiments. This includes specialized methods to store and retrieve spike-in information, dimensionality reduction coordinates and size factors for each cell, along with the usual metadata for genes and libraries.

License GPL-3

VignetteBuilder knitr

RoxygenNote 6.1.0

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

git_branch RELEASE_3_9

git_last_commit baa51d7

git_last_commit_date 2019-05-02

Date/Publication 2019-10-15

Author Aaron Lun [aut, cph], Davide Risso [aut, cre, cph], Keegan Korthauer [ctb]

Maintainer Davide Risso <risso.davide@gmail.com>

R topics documented:

`Combining LEMs` *LEM combining methods*

Description

Methods to combine LinearEmbeddingMatrix objects.

Usage

```
## S4 method for signature 'LinearEmbeddingMatrix'
rbind(..., deparse.level=1)

## S4 method for signature 'LinearEmbeddingMatrix'
cbind(..., deparse.level=1)
```

Arguments

`...`	One or more LinearEmbeddingMatrix objects.
`deparse.level`	An integer scalar; see `?base::cbind` for a description of this argument.

Details

For `rbind`, LinearEmbeddingMatrix objects are combined row-wise, i.e., rows in successive objects are appended to the first object. This corresponds to adding more samples to the first object. Note that `featureLoadings` and `factorData` will only be taken from the first element in the list; no checks are performed to determine whether they are consistent or not across objects.

For `cbind`, LinearEmbeddingMatrix objects are combined columns-wise, i.e., columns in successive objects are appended to the first object. This corresponds to adding more factors to the first object. `featureLoadings` will also be combined column-wise across objects, provided that the number of features is the same across objects. Similarly, `factorData` will be combined row-wise across objects.

Combining objects with and without row names will result in the removal of all row names; similarly for column names. Duplicate row names are currently supported by duplicate column names are not, and will be de-duplicated appropriately.

Value

A LinearEmbeddingMatrix object containing all rows/columns of the supplied objects.

Author(s)

Aaron Lun

Examples

```
example(LinearEmbeddingMatrix, echo=FALSE) # using the class example
rbind(lem, lem)
cbind(lem, lem)
```

`Combining SCEs` *SCE combining methods*

Description

Methods to combine SingleCellExperiment objects.

Usage

```
## S4 method for signature 'SingleCellExperiment'
rbind(..., deparse.level=1)

## S4 method for signature 'SingleCellExperiment'
cbind(..., deparse.level=1)
```

Arguments

`...`	One or more SingleCellExperiment objects.
`deparse.level`	An integer scalar; see `?base::cbind` for a description of this argument.

Details

For `rbind`, SingleCellExperiment objects are combined row-wise, i.e., rows in successive objects are appended to the first object. Column metadata, experimental metadata and `reducedDims` coordinates will only be taken from the first element in the list.

For `cbind`, SingleCellExperiment objects are combined columns-wise, i.e., columns in successive objects are appended to the first object. `reducedDims` coordinates will be combined row-wise to reflect the addition or more cells. Row and experimental metadata will only be taken from the first element in the list.

Value

A SingleCellExperiment object containing all rows/columns of the supplied objects.

Author(s)

Aaron Lun

Examples

```
example(SingleCellExperiment, echo=FALSE) # using the class example
rbind(sce, sce)
cbind(sce, sce)
dim(reducedDims(sce)[[1]])
```

`Getter/setter methods` *LinearEmbeddingMatrix getters/setters*

Description

Getter/setter methods for the LinearEmbeddingMatrix class.

Usage

```
## S4 method for signature 'LinearEmbeddingMatrix'
sampleFactors(x, withDimnames=TRUE)

## S4 replacement method for signature 'LinearEmbeddingMatrix'
sampleFactors(x) <- value

## S4 method for signature 'LinearEmbeddingMatrix'
featureLoadings(x, withDimnames=TRUE)

## S4 replacement method for signature 'LinearEmbeddingMatrix'
featureLoadings(x) <- value

## S4 method for signature 'LinearEmbeddingMatrix'
factorData(x)

## S4 replacement method for signature 'LinearEmbeddingMatrix'
factorData(x) <- value

## S4 method for signature 'LinearEmbeddingMatrix'
as.matrix(x, ...)

## S4 method for signature 'LinearEmbeddingMatrix'
dim(x)

## S4 method for signature 'LinearEmbeddingMatrix'
dimnames(x)

## S4 replacement method for signature 'LinearEmbeddingMatrix'
dimnames(x) <- value

## S4 method for signature 'LinearEmbeddingMatrix'
x$name

## S4 replacement method for signature 'LinearEmbeddingMatrix'
x$name <- value
```

Arguments

`x`	A LinearEmbeddingMatrix object.
`value`	An appropriate value to assign to the relevant slot.
`withDimnames`	A logical scalar indicating whether dimension names should be attached to the returned object.

name	A string specifying a field of the factorData slot.
...	Further arguments, ignored.

Details

Any value to assign to sampleFactors and featureLoadings should be matrix-like objects, while factorData should be a DataFrame - ee LinearEmbeddingMatrix for details.

The as.matrix method will return the matrix of sample factors, consistent with the fact that the LinearEmbeddingMatrix mimics a sample-factor matrix. However, unlike the sampleFactors method, this is always guaranteed to return an ordinary R matrix, even if an alternative representation was stored in the slot. This ensures consistency with as.matrix methods for other matrix-like S4 classes.

For assignment to dimnames, a list of length 2 should be used containing vectors of row and column names.

Value

For the getter methods sampleFactors, featureLoadings and factorData, the value of the slot with the same name is returned. For the corresponding setter methods, a LinearEmbeddingMatrix is returned with modifications to the named slot.

For dim, the dimensions of the sampleFactors slot are returned in an integer vector of length 2. For dimnames, a list of length 2 containing the row and column names is returned. For as.matrix, an ordinary matrix derived from sampleFactors is returned.

For $, the value of the named field of the factorData slot is returned. For $<-, a LinearEmbeddingMatrix is returned with the modified field in factorData.

Author(s)

Keegan Korthauer, Davide Risso and Aaron Lun

See Also

LinearEmbeddingMatrix

Examples

```
example(LinearEmbeddingMatrix, echo=FALSE) # Using the class example

sampleFactors(lem)
sampleFactors(lem) <- sampleFactors(lem) * -1

featureLoadings(lem)
featureLoadings(lem) <- featureLoadings(lem) * -1

factorData(lem)
factorData(lem)$whee <- 1

nrow(lem)
ncol(lem)
colnames(lem) <- LETTERS[seq_len(ncol(lem))]
as.matrix(lem)
```

`LinearEmbeddingMatrix` *LinearEmbeddingMatrix class*

Description

A description of the LinearEmbeddingMatrix class for storing low-dimensional embeddings from linear dimensionality reduction methods.

Usage

```
LinearEmbeddingMatrix(sampleFactors = matrix(nrow = 0, ncol = 0),
    featureLoadings = matrix(nrow = 0, ncol = 0), factorData = NULL,
    metadata = list())
```

Arguments

`sampleFactors`	A matrix-like object of sample embeddings, where rows are samples and columns are factors.
`featureLoadings`	A matrix-like object of feature loadings, where rows are features and columns are factors.
`factorData`	A DataFrame containing factor-level information, with one row per factor.
`metadata`	An optional list of arbitrary content describing the overall experiment.

Details

The LinearEmbeddingMatrix class is a matrix-like object that supports `dim`, `dimnames` and `as.matrix`. It is designed for the storage of results from linear dimensionality reduction methods like principal components analysis (PCA), factor analysis and non-negative matrix factorization.

The `sampleFactors` slot is intended to store The low-dimensional representation of the samples, such as the principal coordinates from PCA. The feature loadings contributing to each factor are stored in `featureLoadings`, and should have the same number of columns as `sampleFactors`. The `factorData` stores additional factor-level information, such as the percentage of variance explained by each factor, and should have the same number of rows as `sampleFactors`.

The intended use of this class is to allow PCA and other results to be stored in the `reducedDims` slot of a SingleCellExperiment object. This means that feature loadings remain attached to the embedding, allowing it to be used in downstream analyses.

Value

A LinearEmbeddingMatrix object is returned from the constructor.

Author(s)

Aaron Lun, Davide Risso and Keegan Korthauer

Examples

```
lem <- LinearEmbeddingMatrix(matrix(rnorm(1000), ncol=5),
    matrix(runif(20000), ncol=5))
lem
```

Miscellaneous LEM *Miscellaneous LEM methods*

Description

Various methods for the LinearEmbeddingMatrix class.

Usage

```
## S4 method for signature 'LinearEmbeddingMatrix'
show(object)
```

Arguments

object	A LinearEmbeddingMatrix object.

Details

The show method will print out information about the data contained in object. This includes the number of samples, the number of factors, the number of genes and the fields available in factorData.

Value

A message is printed to screen describing the data stored in object.

Author(s)

Davide Risso

See Also

LinearEmbeddingMatrix

Examples

```
example(LinearEmbeddingMatrix, echo=FALSE) # Using the class example
show(lem)
```

Miscellaneous SCE *Miscellaneous SCE methods*

Description

Various methods for the SingleCellExperiment class.

Usage

```
## S4 method for signature 'SingleCellExperiment'
show(object)

## S4 method for signature 'SingleCellExperiment'
objectVersion(x)
```

Arguments

x, object	A SingleCellExperiment object.

Details

The show method will print out information about the data contained in object. This describes the stored assays and row/column metadata, as done in show,SummarizedExperiment-method. The names of the reducedDims slot and the names of the spike-ins (see spikeNames) are also reported.

The objectVersion method will return the version of the package with which x was constructed. This is useful for checking if the object is up to date relative to the latest versions of the package.

Value

For show, a message is printed to screen describing the data stored in object.

For objectVersion, an object of class package_version is returned.

See Also

SingleCellExperiment

Examples

```
example(SingleCellExperiment, echo=FALSE) # Using the class example
show(sce)
objectVersion(sce)
```

---

namedAssays *Named assay fields*

Description

Convenience methods to get or set named assay fields.

Usage

```
## S4 method for signature 'SingleCellExperiment'
counts(object, ...)
## S4 replacement method for signature 'SingleCellExperiment'
counts(object, ...) <- value

## S4 method for signature 'SingleCellExperiment'
normcounts(object, ...)
## S4 replacement method for signature 'SingleCellExperiment'
normcounts(object, ...) <- value

## S4 method for signature 'SingleCellExperiment'
logcounts(object, ...)
## S4 replacement method for signature 'SingleCellExperiment'
logcounts(object, ...) <- value

## S4 method for signature 'SingleCellExperiment'
```

```
cpm(object, ...)
## S4 replacement method for signature 'SingleCellExperiment'
cpm(object, ...) <- value

## S4 method for signature 'SingleCellExperiment'
tpm(object, ...)
## S4 replacement method for signature 'SingleCellExperiment'
tpm(object, ...) <- value

## S4 method for signature 'SingleCellExperiment'
weights(object, ...)
## S4 replacement method for signature 'SingleCellExperiment'
weights(object, ...) <- value
```

Arguments

object	A SingleCellExperiment object.
value	A numeric matrix of the same dimensions as object.
...	May contain withDimnames, which is forwarded to assays.

Details

These are wrapper methods for getting or setting assay(object,i=X,...) where X is the name of the method. For example, counts will get or set X="counts". This provide some convenience for users as well as encouraging standardization of naming across packages.

Our suggested interpretation of the fields are as follows:

counts: Raw count data, e.g., number of reads or transcripts.

normcounts: Normalized values on the same scale as the original counts. For example, counts divided by cell-specific size factors that are centred at unity.

logcounts: Log-transformed counts or count-like values. In most cases, this will be defined as log-transformed normcounts, e.g., using log base 2 and a pseudo-count of 1.

cpm: Counts-per-million. This is the read count for each gene in each cell, divided by the library size of each cell in millions.

tpm: Transcripts-per-million. This is the number of transcripts for each gene in each cell, divided by the total number of transcripts in that cell (in millions).

weights: A matrix of weights, e.g., observational weights to be used in differential expression analysis.

Value

Each method returns a matrix from the correspondingly named field in the assays slot.

Author(s)

Aaron Lun

See Also

SingleCellExperiment

Examples

```
example(SingleCellExperiment, echo=FALSE) # Using the class example
counts(sce) <- matrix(rnorm(nrow(sce)*ncol(sce)), ncol=ncol(sce))
dim(counts(sce))

# One possible way of computing normalized "counts"
sf <- 2^rnorm(ncol(sce))
sf <- sf/mean(sf)
normcounts(sce) <- t(t(counts(sce))/sf)
dim(normcounts(sce))

# One possible way of computing log-counts
logcounts(sce) <- log2(normcounts(sce)+1)
dim(normcounts(sce))
```

Reduced dimensions *Reduced dimensions methods*

Description

Methods to get or set the dimensionality reduction results.

Usage

```
## S4 method for signature 'SingleCellExperiment'
reducedDim(x, type=1, withDimnames=TRUE)

## S4 replacement method for signature 'SingleCellExperiment'
reducedDim(x, type=1) <- value

## S4 method for signature 'SingleCellExperiment'
reducedDims(x, withDimnames=TRUE)

## S4 replacement method for signature 'SingleCellExperiment'
reducedDims(x) <- value

## S4 method for signature 'SingleCellExperiment'
reducedDimNames(x)

## S4 replacement method for signature 'SingleCellExperiment,character'
reducedDimNames(x) <- value
```

Arguments

x	A SingleCellExperiment object.
type	A string containing the name for the dimensionality reduction results or a numeric index containing the position of the desired dimenionality reduction result.
withDimnames	A logical scalar indicating whether each set of results should be returned with row names matching `colnames(x)`. This can be turned off for greater efficiency.

value — For reducedDim<-, a matrix (usually double-precision) of coordinates, for each cell (row) and dimension (column).

For reducedDims<-, a named SimpleList object containing such matrices.

For reducedDimNames<-, a character vector containing the names of all dimensionality reduction results.

Details

Dimensionality reduction is often used to interpreting the results of single-cell data analysis. These methods allow the results of dimensionality reduction methods to be stored in a SingleCellExperiment object. Multiple results can be stored in a single object by assigning to different type in reducedDim<-.

If value is NULL for reducedDim<-, the set of results corresponding to type is removed from the object. If value is NULL for reducedDims<-, all dimensionality reduction results are removed.

Note that the reducedDims slot must *always* be named for consistency. Unnamed results assigned via reducedDim<- or reducedDims<- will be assigned empty names.

Value

For reducedDim, a numeric matrix is returned containing coordinates for cells (rows) and dimensions (columns).

For reducedDims, a named SimpleList of matrices is returned, with one matrix for each type of dimensionality reduction method.

For reducedDimNames, a character vector containing the names of the elements in reducedDims.

For reducedDim<- and reducedDims<-, a SingleCellExperiment object is returned with updated results in the reducedDims slot.

Author(s)

Aaron Lun

See Also

SingleCellExperiment-class

Examples

```
example(SingleCellExperiment, echo=FALSE)
reducedDim(sce, "PCA")
reducedDim(sce, "tSNE")
reducedDims(sce)

reducedDim(sce, "PCA") <- NULL
reducedDims(sce)

reducedDims(sce) <- SimpleList()
reducedDims(sce)
```

SCE colData *SCE col/rowData methods*

Description

Methods to obtain column- or row-level metadata from the SingleCellExperiment class.

Usage

```
## S4 method for signature 'SingleCellExperiment'
colData(x, ..., internal=FALSE)

## S4 method for signature 'SingleCellExperiment'
rowData(x, ..., internal=FALSE)
```

Arguments

x	A SingleCellExperiment object.
...	Further arguments to be passed to the methods for SummarizedExperiment objects, such as use.names.
internal	Whether the information contained in the internal slots should be returned.

Details

It may sometimes be useful to return both the visible and the internal colData in a single DataFrame (see SingleCellExperiment-class). This can be achieved by using colData(x,internal=TRUE), which will return the stored colData along with the int_colData (currently the sizeFactors). Similarly, rowData(x,internal=TRUE) will return the stored rowData along with the int_rowData (currently the columns corresponding to isSpike). Warnings will be raised in the event of any name clashes.

Value

colData and rowData return a DataFrame with number of rows equal to ncol(x) and nrow(x), respectively.

See Also

SingleCellExperiment

Examples

```
example(SingleCellExperiment, echo=FALSE) # Using the class example
sizeFactors(sce) <- runif(ncol(sce))
isSpike(sce, "ERCC") <- rbinom(nrow(sce), 1, 0.2)==1
rowData(sce, internal=TRUE)
colData(sce, internal=TRUE)
```

SCE internals *SCE internal methods*

Description

Methods to get or set internal fields from the SingleCellExperiment class.

Usage

```
## S4 method for signature 'SingleCellExperiment'
int_elementMetadata(x)

## S4 replacement method for signature 'SingleCellExperiment'
int_elementMetadata(x) <- value

## S4 method for signature 'SingleCellExperiment'
int_colData(x)

## S4 replacement method for signature 'SingleCellExperiment'
int_colData(x) <- value

## S4 method for signature 'SingleCellExperiment'
int_metadata(x)

## S4 replacement method for signature 'SingleCellExperiment'
int_metadata(x) <- value
```

Arguments

x	A SingleCellExperiment object.
value	For `int_elementMetadata`, a DataFrame with number of rows equal to `nrow(x)`. For `int_colData`, a DataFrame with number of rows equal to `ncol(x)`. For `int_metadata`, a list.

Details

These functions are intended for package developers who want to add protected fields to a SingleCellExperiment. They should *not* be used by ordinary users of the **SingleCellExperiment** package.

Package developers intending to use these methods should read the development vignette for instructions.

Value

A SingleCellExperiment object equivalent to `x` but with modified internal fields.

See Also

SingleCellExperiment

Examples

```
example(SingleCellExperiment, echo=FALSE) # Using the class example
int_metadata(sce)$whee <- 1
```

`SingleCellExperiment` *SingleCellExperiment class*

Description

A description of the SingleCellExperiment class for storing single-cell sequencing data.

Usage

```
SingleCellExperiment(..., reducedDims=SimpleList())
```

Arguments

`...`	Arguments to pass to the `SummarizedExperiment` constructor.
`reducedDims`	A SimpleList object containing matrices of cell coordinates in reduced space.

Details

The SingleCellExperiment class inherits from the SummarizedExperiment class, with several additional slots:

`reducedDims`: A SimpleList containing matrices of cell coordinates.

`int_elementMetadata`: A DataFrame containing internal row metadata (for each genomic feature).

`int_colData`: A DataFrame containing internal column metadata (for each cell).

`int_metadata`: A list containing internal experiment metadata.

The intended use of this class is the same as that for SummarizedExperiment instances. Rows should represent genomic features such as genes, while columns represent samples - in this case, single cells. Different quantifications (e.g., counts, CPMs, log-expression) can be stored simultaneously in the `assays` slot. Row and column metadata can be attached using `rowData` and `colData`, respectively.

The additional `reducedDims` slot allows storage of results from multiple dimensionality reduction methods, e.g., PCA or t-SNE. Each element of the SimpleList should be a matrix of coordinates for all cells from one reduction method. The number of rows of each matrix should be equal to the number of cells in the SingleCellExperiment object.

The internal metadata slots are not intended for external use. Please use the appropriate getter/setter functions instead, such as `isSpike` or `sizeFactors`. Package developers should refer to the suggestions in `?int_metadata`.

Value

A SingleCellExperiment object is returned from the constructor.

Author(s)

Aaron Lun and Davide Risso

See Also

isSpike, sizeFactors, reducedDims

Examples

```
ncells <- 100
u <- matrix(rpois(20000, 5), ncol=ncells)
v <- log2(u + 1)

pca <- matrix(runif(ncells*5), ncells)
tsne <- matrix(rnorm(ncells*2), ncells)

sce <- SingleCellExperiment(assays=list(counts=u, logcounts=v),
    reducedDims=SimpleList(PCA=pca, tSNE=tsne))
sce

## coercion from SummarizedExperiment
se <- SummarizedExperiment(assays=list(counts=u, logcounts=v))
as(se, "SingleCellExperiment")
```

Size factor methods *Size factors methods*

Description

Gets or sets the size factors for all cells.

Usage

```
## S4 method for signature 'SingleCellExperiment'
sizeFactors(object, type=NULL)

## S4 replacement method for signature 'SingleCellExperiment'
sizeFactors(object, type=NULL) <- value

## S4 method for signature 'SingleCellExperiment'
clearSizeFactors(object)

## S4 method for signature 'SingleCellExperiment'
sizeFactorNames(object)
```

Arguments

object	A SingleCellExperiment object.
type	A string specifying the *type* of size factor to get or set.
value	A numeric vector of size factors for all cells.

Details

A size factor is a scaling factor used to divide the raw counts of a particular cell to obtain normalized expression values. The `sizeFactors` methods can be used to get or set size factors for all cells.

The `type` argument allows storage of multiple vectors of size factors (e.g., different values for spike-ins versus endogenous genes). If `type` is `NULL`, a "default" set of size factors is stored or returned.

If `value` is `NULL` for `isSpike<-`, size factors of `type` will be removed from `object`. All size factors can be removed from `object` by using the `clearSizeFactors` method.

The `sizeFactorNames` method will return the names of all stored size factor sets. This does not include the default set of size factors (obtained with `isSpike(...,type=NULL)`) as these are unnamed.

Value

For `sizeFactors`, a numeric vector is returned containing size factors of the set `type` for all cells. If `type` is not available, `NULL` is returned instead.

For `sizeFactors<-`, a SingleCellExperiment is returned with size factors stored in the internal metadata fields.

For `clearSizeFactors`, a SingleCellExperiment is returned with no size factor information.

For `sizeFactorNames`, a character vector is returned containing the names of all named size factor sets.

Author(s)

Aaron Lun

See Also

`SingleCellExperiment-class`

Examples

```
example(SingleCellExperiment, echo=FALSE) # Using the class example
sizeFactors(sce) <- runif(ncol(sce))
sizeFactors(sce)

sizeFactors(sce, "ERCC") <- runif(ncol(sce))
sizeFactors(sce, "ERCC")
sizeFactors(sce) # unchanged.

sizeFactors(sce, "ERCC") <- NULL
sizeFactors(sce, "ERCC")
```

Spike-in methods *Spike-in methods*

Description

Gets or sets the rows corresponding to spike-in transcripts.

Usage

```
## S4 method for signature 'SingleCellExperiment,character'
isSpike(x, type)

## S4 method for signature 'SingleCellExperiment,missing'
isSpike(x, type)

## S4 method for signature 'SingleCellExperiment,NULL'
isSpike(x, type)

## S4 replacement method for signature 'SingleCellExperiment,character'
isSpike(x, type) <- value

## S4 method for signature 'SingleCellExperiment'
clearSpikes(x)

## S4 method for signature 'SingleCellExperiment'
spikeNames(x)
```

Arguments

x	A SingleCellExperiment object.
type	A string containing the name of the spike-in set.
value	A vector indicating which rows correspond to spike-in transcripts.

Details

Spike-in transcripts may be added during library preparation in single-cell RNA sequencing experiments. These usually need to be handled differently during data analysis, compared to the endogenous genes. Thus, it is important to indicate which rows correspond to spike-in transcripts.

The isSpike<- method accepts any value that indicates which rows correspond to spike-ins. This can be a logical or integer subsetting vector, or a vector of row names. The type should be set to the name of the spike-in set, e.g., "ERCC" or "SIRV".

In this manner, multiple types of spike-in sets are supported for a single experiment. This is useful not only when different spike-ins are used, but also for different mixtures of the same set (e.g., ERCC mixes 1 and 2). The names of all available spike-in sets can be obtained using spikeNames.

To remove spike-ins for a particular set, value should be set to NULL when using isSpike<-. To remove all spike-in information, clearSpikes should be used to obtain a new SingleCellExperiment object with no spike-ins specified.

In previous versions (<= 1.1.1), if value was NULL in isSpike<-, all existing spike-in sets would be removed. This behaviour is now deprecated, and clearSpikes should be used instead. Also,

if type was missing or NULL for isSpike<-, the spike-in set would be automatically assigned an empty name. This is also deprecated, and all spike-ins should be given a user-supplied name.

The isSpike getter methods will return a logical vector indicatng which rows represent spike-ins of the set specified by type. If type is missing or NULL, the vector will instead indicate whether each row is in *any* spike-in set. If type is specified but not available, an error will be raised.

Value

For isSpike, a logical vector is returned indicating whether each row is in the specified set type or any set.

For isSpike<-, a SingleCellExperiment is returned with spike-in information stored in the internal metadata fields.

For spikeNames, a character vector is returned containing the names of available spike-in sets.

For clearSpikes, a SingleCellExperiment is returned with no spike-in information.

Author(s)

Aaron Lun

See Also

SingleCellExperiment-class

Examples

```
example(SingleCellExperiment, echo=FALSE) # Using the class example
isSpike(sce, "ERCC") <- 1:10
isSpike(sce)

isSpike(sce, "SIRV") <- 11:20
spikeNames(sce)
which(isSpike(sce))
which(isSpike(sce, "SIRV"))

isSpike(sce, "ERCC") <- NULL
spikeNames(sce)
```

Subsetting LEMs *LEM subsetting methods*

Description

Methods to subset LinearEmbeddingMatrix objects.

Usage

```
## S4 method for signature 'LinearEmbeddingMatrix,ANY,ANY'
x[i, j, ..., drop=TRUE]

## S4 replacement method for signature
## 'LinearEmbeddingMatrix,ANY,ANY,LinearEmbeddingMatrix'
x[i, j] <- value
```

Arguments

`x`	A LinearEmbeddingMatrix object.
`i, j`	A vector of logical or integer subscripts, indicating the rows and columns to be subsetted for `i` and `j`, respectively.
`...`	Extra arguments that are ignored.
`drop`	A logical scalar indicating whether the result should be coerced to the lowest possible dimension.
`value`	A LinearEmbeddingMatrix object with number of rows equal to length of `i` (or that of `x`, if `i` is not specified). The number of columns must be equal to the length of `j` (or number of columns in `x`, if `j` is not specified).

Details

Subsetting yields a LinearEmbeddingMatrix object containing the specified rows (samples) and columns (factors). If column subsetting is performed, values of `featureLoadings` and `factorData` will be modified to retain only the selected factors.

If `drop=TRUE` and the subsetting would produce dimensions of length 1, those dimensions are dropped and a vector is returned directly from `sampleFactors`. This mimics the expected behaviour from a matrix-like object. Users should set `drop=FALSE` to ensure that a LinearEmbeddingMatrix is returned.

For subset replacement, if neither `i` or `j` are set, `x` will be effectively replaced by `value`. However, row and column names will *not* change, consistent with replacement in ordinary matrices.

Value

For `[`, a subsetted LinearEmbeddingMatrix object is returned.

For `[<-`, a modified LinearEmbeddingMatrix object is returned.

Author(s)

Aaron Lun

See Also

`LinearEmbeddingMatrix-class`

Examples

```
example(LinearEmbeddingMatrix, echo=FALSE) # using the class example

lem[1:10,]
lem[,1:5]

lem2 <- lem
lem2[1:10,] <- lem[11:20,]
```

Subsetting SCEs *SCE subsetting methods*

Description

Methods to subset SingleCellExperiment objects.

Usage

```
## S4 method for signature 'SingleCellExperiment,ANY,ANY'
x[i, j, ..., drop=TRUE]

## S4 replacement method for signature 'SingleCellExperiment,ANY,ANY,SingleCellExperiment'
x[i, j] <- value
```

Arguments

x	A SingleCellExperiment object.
i, j	A vector of logical or integer subscripts, indicating the rows and columns to be subsetted for i and j, respectively.
...	Extra arguments to be passed to [,SummarizedExperiment-method.
drop	A logical scalar that is ignored.
value	A SingleCellExperiment object with number of rows equal to length of i (or that of x, if i is not specified). The number of columns must be equal to the length of j (or number of columns in x, if j is not specified).

Details

Subsetting yields a SingleCellExperiment object containing the specified rows (features) and columns (cells). Internal row and column metadata fields will also be subsetted so that methods such as isSpike are still valid. If column subsetting is performed, values of the reducedDims will be modified to retain only the selected cells.

Subset assignment will replace the assay values and metadata of the specified rows or columns in x with those in value. If both i and j are set, the relevant block of assay values will be replaced, along with the metadata for the affected rows and columns. If neither i or j are set, x will be turned into value.

Value

For [and subset, a subsetted SingleCellExperiment object is returned.

For [<-, a modified SingleCellExperiment object is returned.

Author(s)

Aaron Lun

See Also

SingleCellExperiment-class

Examples

```
example(SingleCellExperiment, echo=FALSE) # using the class example

sce[1:10,]
sce[,1:5]

sce2 <- sce
sce2[1:10,] <- sce[11:20,]

# Can also use subset()
sce$WHEE <- sample(LETTERS, ncol(sce), replace=TRUE)
subset(sce, , WHEE=="A")

# Can also use split()
split(sce, sample(LETTERS, nrow(sce), replace=TRUE))
```

Index