

Package ‘SpatialExperiment’

April 3, 2022

Type Package

Title S4 Class for Spatial Experiments handling

Version 1.4.0

Date 2021-08-23

Description Defines S4 classes for storing data for spatial experiments.

Main examples are reported by using seqFISH and 10x-Visium Spatial Gene Expression data.

This includes specialized methods for storing, retrieving spatial coordinates, 10x dedicated parameters and their handling.

License GPL-3

BugReports <https://github.com/drighelli/SpatialExperiment/issues>

Encoding UTF-8

LazyData false

biocViews DataRepresentation, DataImport, ImmunoOncology,
DataRepresentation, Infrastructure, SingleCell, GeneExpression

Depends methods, SingleCellExperiment

Imports BiocFileCache, DropletUtils, rjson, magick, grDevices,
S4Vectors, SummarizedExperiment, BiocGenerics, utils

Suggests knitr, rmarkdown, testthat, BiocStyle, BumpyMatrix

VignetteBuilder knitr

RoxygenNote 7.1.1

git_url <https://git.bioconductor.org/packages/SpatialExperiment>

git_branch RELEASE_3_14

git_last_commit fd11626

git_last_commit_date 2021-10-26

Date/Publication 2022-04-03

Author Dario Righelli [aut, cre],
Davide Risso [aut],
Helena L. Crowell [aut],
Lukas M. Weber [aut]

Maintainer Dario Righelli <dario.righelli@gmail.com>

R topics documented:

imgData-methods	2
read10xVisium	4
readImgData	6
SpatialExperiment-assays	7
SpatialExperiment-class	8
SpatialExperiment-colData	11
SpatialExperiment-combine	12
SpatialExperiment-methods	14
SpatialExperiment-misc	17
SpatialExperiment-subset	18
SpatialImage-class	18
SpatialImage-misc	20
Index	21

imgData-methods	<i>Methods for handling image-related data</i>
-----------------	--

Description

The set of functions described below is designed to handle the image-related data stored inside a `SpatialExperiment`'s `imgData` `int_metadata` field. These include:

- `getImg`, `addImg`, `rmvImg` to retrieve/add/remove an image entry to/from the `imgData` `DataFrame`
- `imgSource`, `imgRaster` to retrieve the path/URL and raster object, respectively, associated with an image or set of images

Usage

```
## S4 method for signature 'SpatialExperiment'
getImg(x, sample_id = NULL, image_id = NULL)
```

```
## S4 method for signature 'SpatialExperiment'
addImg(x, imageSource, scaleFactor, sample_id, image_id, load = TRUE)
```

```
## S4 method for signature 'SpatialExperiment'
rmvImg(x, sample_id = NULL, image_id = NULL)
```

```
## S4 method for signature 'SpatialExperiment'
imgSource(x, sample_id = NULL, image_id = NULL)
```

```
## S4 method for signature 'SpatialExperiment'
imgRaster(x, sample_id = NULL, image_id = NULL)
```

Arguments

x	a SpatialExperiment
sample_id	character string, TRUE or NULL specifying sample/image identifier(s); here, TRUE is equivalent to all samples/images and NULL specifies the first available entry (see details)
image_id	see sample_id
imageSource	a character string specifying an image file name (.png, .jpg or .tif) or URL to source the image from
scaleFactor	single numeric scale factor used to rescale spatial coordinates according to the image's resolution
load	logical; should the image(s) be loaded into memory as a raster object? if FALSE, will store the path/URL instead

Value

getImg() returns a single or list of SpatialImage(s).

add/rmvImg() return a [SpatialExperiment](#) with modified imgData; specifically, they create/remove an image entry (row) in the imgData DataFrame.

imgRaster/Source() access relevant data in the SpatialImage(s) stored inside the imgData's data field. Depending on whether or not multiple entries are accessed, a character string or vector is returned by imgSource(), and a single or list of raster object(s) is returned by imgRaster().

Author(s)

Helena L. Crowell

Examples

```
example(read10xVisium)

# 'SpatialImage' accession
(spi <- getImg(spe))
plot(imgRaster(spi))

# remove an image
imgData(spe)
spe <- rmvImg(spe,
  sample_id = "section1",
  image_id = "lowres")
imgData(spe)

# add an image
url <- "https://i.redd.it/3pw5uah7xo041.jpg"
spe <- addImg(spe,
  sample_id = "section1",
  image_id = "pomeranian",
  imageSource = url,
  scaleFactor = NA_real_,
```

```

load = FALSE)

# extract image
img <- imgRaster(spe,
  sample_id = "section1",
  image_id = "pomeranian")
plot(img)

```

read10xVisium

Load data from a 10x Genomics Visium experiment

Description

Creates a [SpatialExperiment](#) from the Space Ranger output directories for 10x Genomics Visium spatial gene expression data.

Usage

```

read10xVisium(
  samples = "",
  sample_id = paste0("sample", sprintf("%02d", seq_along(samples))),
  type = c("HDF5", "sparse"),
  data = c("filtered", "raw"),
  images = "lowres",
  load = TRUE
)

```

Arguments

samples	a character vector specifying one or more directories, each corresponding to a 10x Genomics Visium sample (see details); if provided, names will be used as sample identifiers
sample_id	character string specifying unique sample identifiers, one for each directory specified via samples; ignored if <code>!is.null(names(samples))</code>
type	character string specifying the type of format to read count data from (see read10xCounts)
data	character string specifying whether to read in filtered (spots mapped to tissue) or raw data (all spots).
images	character vector specifying which images to include. Valid values are "lowres", "hires", "fullres", "d"
load	logical; should the image(s) be loaded into memory as a grob? If FALSE, will store the path/URL instead.

Details

The constructor assumes data from each sample are located in a single output directory as returned by Space Ranger, thus having the following file organization:

```
sample
├──outs
│   ├──raw/filtered_feature_bc_matrix.h5
│   ├──raw/filtered_feature_bc_matrix
│   ├──barcodes.tsv
│   ├──features.tsv
│   ├──matrix.mtx
│   └──spatial
│       ├──scalefactors_json.json
│       ├──tissue_lowres_image.png
│       └──tissue_positions_list.csv
```

Value

a [SpatialExperiment](#) object

Author(s)

Helena L. Crowell

Examples

```
dir <- system.file(
  file.path("extdata", "10xVisium"),
  package = "SpatialExperiment")

sample_ids <- c("section1", "section2")
samples <- file.path(dir, sample_ids)

list.files(samples[1])
list.files(file.path(samples[1], "spatial"))
file.path(samples[1], "raw_feature_bc_matrix")

(spe <- read10xVisium(samples, sample_ids,
  type = "sparse", data = "raw",
  images = "lowres", load = FALSE))

# tabulate number of spots mapped to tissue
cd <- colData(spe, spatialData = TRUE)
table(
  in_tissue = cd$in_tissue,
  sample_id = cd$sample_id)

# view available images
imgData(spe)
```

readImgData

*Read images & scale factors for 10x Genomics Visium***Description**

Function to read in images and scale factors for 10x Genomics Visium data, and return as a valid [imgData](#) DataFrame.

Usage

```
readImgData(
  path = ".",
  sample_id = names(path),
  imageSources = file.path(path, "tissue_lowres_image.png"),
  scaleFactors = file.path(path, "scalefactors_json.json"),
  load = TRUE
)
```

Arguments

path	a path where to find one or more images
sample_id	the sample_id for the SpatialExperiment object
imageSources	the images source path(s)
scaleFactors	the .json file where to find the scale factors
load	logical; should the image(s) be loaded into memory as a grob? If FALSE, will store the path/URL instead.

Value

a [DataFrame](#)

Author(s)

Helena L. Crowell

Examples

```
dir <- system.file(
  file.path("extdata", "10xVisium", "section1", "spatial"),
  package = "SpatialExperiment")

# base directory contains
# - scale factors (scalefactors_json.json)
# - one image (tissue_lowres_image.png)
list.files(dir)

# read in images & scale factors
```

```
# as valid 'imgData' 'DFrame'  
readImgData(dir, sample_id = "foo")
```

SpatialExperiment-assays

Methods for named assays

Description

The `SpatialExperiment` class provides methods for getting or setting named `assays`. For example, `molecules(spe)` will get or set an assay named `molecules` from object `spe`, equivalent to `assay(spe, i = "molecules")`. This provides a convenient interface for users and encourages standardization of assay names across packages.

Available methods

In the following code, `spe` is a `SpatialExperiment` object, `value` is a `BumpyMatrix`-like object with the same dimensions as `spe`, and `...` are further arguments passed to `assay` (for the getter) or `assay<-` (for the setter).

```
molecules(x, ...), molecules(x, ...) <- value: Get or set an assay named molecules, which is usually assumed to be a BumpyMatrix-formatted object containing spatial coordinates (and any other information) of the individual molecules per gene per cell.
```

Author(s)

Dario Righelli

See Also

`assay` and `assay<-`

Examples

```
example(SpatialExperiment)  
molecules(spe_mol)
```

SpatialExperiment-class

The SpatialExperiment class

Description

The `SpatialExperiment` class is designed to represent spatially resolved transcriptomics (ST) data. It inherits from the `SingleCellExperiment` class and is used in the same manner. In addition, the class supports storage of spatial information via `spatialData` and `spatialCoords`, and storage of images via `imgData`.

Arguments

<code>...</code>	Arguments passed to the <code>SingleCellExperiment</code> constructor to fill the slots of the base class.
<code>sample_id</code>	A character sample identifier, which matches the <code>sample_id</code> in <code>imgData</code> . The <code>sample_id</code> will also be stored in a new column in <code>colData</code> , if not already present. Default = <code>sample01</code> .
<code>spatialDataNames</code>	A character vector of column names from <code>colData</code> to include in <code>spatialData</code> . Alternatively, the <code>spatialData</code> argument may be provided. If both are provided, <code>spatialDataNames</code> is given precedence, and a warning is returned.
<code>spatialCoordsNames</code>	A character vector of column names from <code>colData</code> or <code>spatialData</code> containing spatial coordinates, which will be accessible with <code>spatialCoords</code> . Alternatively, the <code>spatialCoords</code> argument may be provided. If both are provided, <code>spatialCoordsNames</code> is given precedence, and a warning is returned. Default = <code>c("x", "y")</code> .
<code>spatialData</code>	A <code>DataFrame</code> containing columns to store in <code>spatialData</code> , which must contain at least the columns of spatial coordinates. Alternatively, <code>spatialDataNames</code> may be provided. If both are provided, <code>spatialDataNames</code> is given precedence, and a warning is returned.
<code>spatialCoords</code>	A numeric matrix containing columns of spatial coordinates, which will be accessible with <code>spatialCoords</code> . Alternatively, <code>spatialCoordsNames</code> may be provided. If both are provided, <code>spatialCoordsNames</code> is given precedence, and a warning is returned.
<code>scaleFactors</code>	Optional scale factors associated with the image(s). This can be provided as a numeric value, numeric vector, list, or file path to a JSON file for the 10x Genomics Visium platform. For 10x Genomics Visium, the correct scale factor will automatically be selected depending on the resolution of the image from <code>imageSources</code> . Default = 1.
<code>imgData</code>	Optional <code>DataFrame</code> containing the image data. Alternatively, this can be built from the arguments <code>imageSources</code> and <code>image_id</code> (see Details).
<code>imageSources</code>	Optional file path(s) or URL(s) for one or more image sources.

image_id	Optional character vector (same length as imageSources) containing unique image identifiers.
loadImage	Logical indicating whether to load image into memory. Default = FALSE.

Details

In this class, rows represent genes, and columns represent spots (for spot-based ST platforms) or cells (for molecule-based ST platforms). As for [SingleCellExperiment](#), counts and logcounts can be stored in the [assays](#) slot, and row and column metadata in [rowData](#) and [colData](#). For molecule-based ST data, the additional measurements per molecule per cell can be stored in a BumpyMatrix-formatted assay named [molecules](#).

The additional arguments in the constructor documented above (e.g. [spatialData](#), [spatialCoords](#), [imgData](#), and others) represent the main extensions to the [SingleCellExperiment](#) class to store associated spatial and imaging information for ST data.

The constructor expects [colData](#) to contain a column named `sample_id`. If this is not present, it will assign the value from the `sample_id` argument. If the [imgData](#) argument is provided, the constructor expects the [imgData](#) [DataFrame](#) to already be built. Otherwise, it will build it from the [imageSources](#) and (optional) `image_id` arguments. If `image_id` is not provided, this will be assumed from `sample_id` and [imageSources](#) instead. To combine multiple samples within a single object, see [combine](#).

For 10x Genomics Visium datasets, the function [read10xVisium](#) can be used to load data and create a [SpatialExperiment](#) object directly from Space Ranger output files.

Value

A [SpatialExperiment](#) object.

Author(s)

Dario Righelli and Helena L. Crowell

See Also

?["SpatialExperiment-methods"](#), which includes: [spatialData](#), [spatialDataNames](#), [spatialCoords](#), [spatialCoordsNames](#), [imgData](#), [scaleFactors](#)

?["SpatialExperiment-assays"](#), which includes: [molecules](#)

?["SpatialExperiment-colData"](#)

?["SpatialExperiment-combine"](#)

?["SpatialExperiment-subset"](#)

?["SpatialExperiment-misc"](#)

[readImgData](#)

?["imgData-methods"](#)

[SpatialImage](#)

[read10xVisium](#)

Examples

```
#####
# Example 1: Spot-based ST (10x Visium) using constructor
#####

dir <- system.file(
  file.path("extdata", "10xVisium", "section1"),
  package = "SpatialExperiment")

# read in counts
fnm <- file.path(dir, "raw_feature_bc_matrix")
sce <- DropletUtils::read10xCounts(fnm)

# read in image data
img <- readImgData(
  path = file.path(dir, "spatial"),
  sample_id="foo")

# read in spatial coordinates
fnm <- file.path(dir, "spatial", "tissue_positions_list.csv")
xyz <- read.csv(fnm, header = FALSE,
  col.names = c(
    "barcode", "in_tissue", "array_row", "array_col",
    "pxl_row_in_fullres", "pxl_col_in_fullres"))

# construct observation & feature metadata
rd <- S4Vectors::DataFrame(
  symbol = rowData(sce)$Symbol)

# construct 'SpatialExperiment'
(spe <- SpatialExperiment(
  assays = list(counts = assay(sce)),
  colData = colData(sce), rowData = rd, imgData = img,
  spatialData=DataFrame(xyz),
  spatialCoordsNames=c("pxl_col_in_fullres", "pxl_row_in_fullres"),
  sample_id="foo"))

#####
# Example 2: Spot-based ST (10x Visium) using 'read10xVisium'
#####

# see ?read10xVisium for details
example(read10xVisium)

#####
# Example 3: Molecule-based ST
#####

# create simulated data
n <- 1000; ng <- 50; nc <- 20
# sample xy-coordinates in [0,1]
x <- runif(n)
```

```

y <- runif(n)
# assign each molecule to some gene-cell pair
gs <- paste0("gene", seq(nc))
cs <- paste0("cell", seq(nc))
gene <- sample(gs, n, TRUE)
cell <- sample(cs, n, TRUE)
# construct data.frame of molecule coordinates
df <- data.frame(gene, cell, x, y)

# (assure gene & cell are factor so that
# missing observations aren't dropped)
df$gene <- factor(df$gene, gs)
df$cell <- factor(df$cell, cs)

# construct BumpyMatrix
mol <- BumpyMatrix::splitAsBumpyMatrix(
  df[, c("x", "y")],
  row = df$gene, column = df$cell)

# get count matrix
y <- with(df, table(gene, cell))
y <- as.matrix(unclass(y))

# construct SpatialExperiment
spe_mol <- SpatialExperiment(
  assays = list(
    counts = y,
    molecules = mol))

```

SpatialExperiment-colData

SpatialExperiment colData

Description

The `SpatialExperiment` class provides a modified `colData` setter, which ensures that the `SpatialExperiment` object remains valid.

Usage

```

## S4 replacement method for signature 'SpatialExperiment,DataFrame'
colData(x) <- value

## S4 replacement method for signature 'SpatialExperiment,`NULL`'
colData(x) <- value

```

Arguments

x	a <code>SpatialExperiment</code>
value	a <code>DataFrame</code>

Details

The `colData` setter performs several checks to ensure validity. If the replacement `colData` does not contain a `sample_id` column, the existing `sample_ids` will be retained. If the replacement `colData` contains `sample_ids`, a check is performed to ensure the number of unique `sample_ids` is the same, i.e. a one-to-one mapping is possible. If the replacement is `NULL`, the `sample_ids` are retained. In addition, checks are performed against the `sample_ids` in `imgData`.

Value

a `SpatialExperiment` object with updated `colData`

Examples

```
example(read10xVisium)

# return additional columns for colData
head(colData(spe, spatialData = TRUE, spatialCoords = TRUE))

# empty replacement retains sample identifiers
colData(spe) <- NULL
names(colData(spe))

# replacement of sample identifiers
# requires one-to-one mapping

## invalid replacement

tryCatch(
  spe$sample_id <- seq(ncol(spe)),
  error = function(e) message(e))

## valid replacement

old <- c("section1", "section2")
new <- c("sample_A", "sample_B")
idx <- match(spe$sample_id, old)

tmp <- spe
tmp$sample_id <- new[idx]
table(spe$sample_id, tmp$sample_id)
```

SpatialExperiment-combine

Combining SpatialExperiment objects

Description

The `SpatialExperiment` class provides modified methods to combine multiple `SpatialExperiment` objects by column, for example from multiple samples. These methods ensure that all data fields remain synchronized when samples are added or removed.

Usage

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

Arguments

```
...          a list of SpatialExperiment objects
deparse.level refer to ?rbind
```

Value

a combined [SpatialExperiment](#) object

Combining

The ... argument is assumed to contain one or more [SpatialExperiment](#) objects.

`cbind(..., deparse.level=1)`: Returns a [SpatialExperiment](#) where all objects in ... are combined column-wise, i.e., columns in successive objects are appended to the first object.

Each [SpatialExperiment](#) object in ... must have the same `colData` (with the same [spatialCoords](#)). If multiple objects use the same `sample_id`, the method will proceed by assigning unique `sample_ids`.

Additionally, the method combines `imgData` by row using `rbind`.

Refer to [?"cbind,SingleCellExperiment-method"](#) for details on how metadata and other inherited attributes are combined in the output object.

Refer to [?cbind](#) for the interpretation of `deparse.level`.

Author(s)

Dario Righelli

Examples

```
example(read10xVisium, echo = FALSE)

# merging with duplicated 'sample_id's
# will automatically assign unique identifiers
spe1 <- spe2 <- spe
spe3 <- cbind(spe1, spe2)
unique(spe3$sample_id)

# assign unique sample identifiers
spe1 <- spe2 <- spe
spe1$sample_id <- paste(spe1$sample_id, "sample1", sep = ".")
spe2$sample_id <- paste(spe2$sample_id, "sample2", sep = ".")

# combine into single object
spe <- cbind(spe1, spe2)

# view joint 'imgData'
```

```
imgData(spe)

# tabulate number of spots mapped to tissue
cd <- colData(spe, spatialData = TRUE)
table(
  in_tissue = cd$in_tissue,
  sample_id = cd$sample_id)
```

SpatialExperiment-methods

Methods for spatial attributes

Description

The [SpatialExperiment](#) class provides a family of methods to get and set spatial data attributes in [SpatialExperiment](#) objects. Spatial attributes include `spatialData`, `spatialCoords`, `imgData`, and `scaleFactors`.

Usage

```
## S4 method for signature 'SpatialExperiment'
spatialData(x, spatialCoords = FALSE)

## S4 replacement method for signature 'SpatialExperiment,DFrame'
spatialData(x) <- value

## S4 replacement method for signature 'SpatialExperiment,`NULL`'
spatialData(x) <- value

## S4 method for signature 'SpatialExperiment'
spatialDataNames(x)

## S4 replacement method for signature 'SpatialExperiment,character'
spatialDataNames(x) <- value

## S4 method for signature 'SpatialExperiment'
spatialCoords(x)

## S4 replacement method for signature 'SpatialExperiment,matrix'
spatialCoords(x) <- value

## S4 replacement method for signature 'SpatialExperiment,`NULL`'
spatialCoords(x) <- value

## S4 method for signature 'SpatialExperiment'
spatialCoordsNames(x)
```

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

## S4 replacement method for signature 'SpatialExperiment,NULL`'
spatialCoordsNames(x) <- value

## S4 method for signature 'SpatialExperiment'
scaleFactors(x, sample_id = TRUE, image_id = TRUE)

## S4 method for signature 'SpatialExperiment'
imgData(x)

## S4 replacement method for signature 'SpatialExperiment,DataFrame'
imgData(x) <- value

## S4 replacement method for signature 'SpatialExperiment,NULL`'
imgData(x) <- value
```

Arguments

<code>x</code>	A SpatialExperiment object.
<code>spatialCoords</code>	Logical specifying whether to include columns from <code>spatialCoords</code> in the output DataFrame from <code>spatialData</code> . Default = FALSE.
<code>value</code>	Replacement value for replacement methods.
<code>sample_id</code>	Logical value or character vector specifying sample identifier(s) for <code>scaleFactors</code> . Default = TRUE (all samples).
<code>image_id</code>	Logical value or character vector specifying image identifier(s) for <code>scaleFactors</code> . Default = TRUE (all images).
<code>spatialData</code>	Logical specifying whether to include columns from <code>spatialData</code> in the output DataFrame from <code>colData</code> . Default = FALSE

Details

Additional details for each type of data attribute are provided below.

[spatialData](#) and [spatialCoords](#) are distinguished as follows: `spatialData` is a `DataFrame` containing all the data associated with the spatial information (optionally including spatial coordinates from `spatialCoords`), while `spatialCoords` is a numeric matrix containing only the defined spatial coordinates (e.g. columns `x` and `y`).

Value

Return value varies depending on method, as described below.

`spatialData` and `spatialCoords` methods

`spatialData(x)`: The `spatialData` getter provides the optional argument `spatialCoords`, which can be used to include the columns of spatial coordinates (`spatialCoords`) in the output `DataFrame`.

`spatialData(x) <- value`: The `spatialData` setter expects a `DataFrame`. If the input does not contain an `in_tissue` column, this will be included with a default value of 1.

`spatialCoords(x)`: Getter for numeric matrix of spatial coordinates.

`spatialCoords(x) <- value`: Setter for numeric matrix of spatial coordinates.

spatialDataNames and spatialCoordsNames methods

`spatialDataNames(x)`: Returns the column names of the `spatialData` `DataFrame`.

`spatialDataNames(x) <- value`: Setter to replace column names in the `spatialData` `DataFrame`.

`spatialCoordsNames(x)`: Returns the defined names of the spatial coordinates (e.g. `c("x", "y")`).

`spatialCoordsNames(x) <- value`: Setter to define the names of the spatial coordinate columns.

imgData methods

`imgData(x)`: Getter to return the `imgData` `DataFrame`.

`imgData(x) <- value`: Setter to provide a `DataFrame` object as `imgData` of the `SpatialExperiment` object.

Other methods

`scaleFactors(x, sample_id, image_id)`: Getter to return the scale factors associated with the `sample_id(s)` and `image_id(s)` provided. This is related to the stored image(s) in the `SpatialExperiment` `imgData` structure. See argument descriptions for further details.

Examples

```
example(read10xVisium)

# spatialData returns a DataFrame
spatialData(spe)

# spatialCoords returns a numeric matrix
head(spatialCoords(spe))

# spatialData replacement method
spdata <- spatialData(spe)
spdata$array_col <- spdata$array_row
spatialData(spe) <- spdata

# return additional columns for spatialData
spatialData(spe, spatialCoords=TRUE)

# change spatial coordinate names
spatialCoordsNames(spe)
spatialCoordsNames(spe) <- c("x", "y")
head(spatialCoords(spe))

# imgData and scale factors
imgData(spe)
scaleFactors(spe)
```

```
# tabulate number of spots mapped to tissue
cd <- colData(spe, spatialData = TRUE)
table(
  in_tissue = cd$in_tissue,
  sample_id = cd$sample_id)
```

SpatialExperiment-misc

Miscellaneous SpatialExperiment methods

Description

Miscellaneous methods for the [SpatialExperiment](#) class and its descendants that do not fit into any other documentation category such as, for example, show methods.

Usage

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

Arguments

object a [SpatialExperiment](#) object

Value

Returns NULL

Author(s)

Dario Righelli and Helena L. Crowell

Examples

```
example(read10xVisium)
spe
```

SpatialExperiment-subset

Subsetting SpatialExperiment objects

Description

The subsetting method for `SpatialExperiment` objects ensures that spatial data attributes (`spatialData`, `spatialCoords`, `imgData`) are subsetting correctly to match rows and columns with the remainder of the object.

Arguments

<code>x</code>	a <code>SpatialExperiment</code> object
<code>i</code>	row indices for subsetting
<code>j</code>	column indices for subsetting

Value

a `SpatialExperiment` object

subset

[`:` subsetting method

Examples

```
example(read10xVisium)

dim(spe)

set.seed(123)
idx <- sample(ncol(spe), 10)
sub <- spe[, idx]
dim(sub)
spatialData(sub, spatialCoords = TRUE)
```

SpatialImage-class

The SpatialImage class

Description

The `SpatialImage` class hierarchy provides representations of images from a variety of sources. It is used by the `SpatialExperiment` class to manage the loading of images across multiple studies.

Constructor

`SpatialImage(x, is.url)` will return a `SpatialImage` object. The class of the object depends on the type of `x`:

- If `x` is a raster object, a `LoadedSpatialImage` is returned. This represents an image that is fully realized into memory, where the raster representation is stored inside the output object.
- If `x` is a string and `is.url=TRUE` or it starts with "`http://`", "`https://`" or "`ftp://`", a `RemoteSpatialImage` is returned. This represents an image that is remotely hosted and retrieved only on request.
- If `x` is a string and `is.url=TRUE` or it does not start with a URL-like prefix, a `StoredSpatialImage` is returned. This represents an image that is stored in a local file and is loaded into memory only on request.

Getting the raster image

For a `SpatialImage` object `x`, `imgRaster(x, ...)` will return a raster object (see [?as.raster](#)). This is effectively a matrix of RGB colors for each pixel in the image.

For a `StoredSpatialImage` object `x`, additional arguments in `...` are passed to `image_read`. This controls how the image is read into memory.

For a `RemoteSpatialImage` object `x`, the image file is first downloaded before the raster is returned. Here, `...` may contain an extra cache argument, which should be a `BiocFileCache` object (from the **BiocFileCache** package) specifying the file cache location. The default location is determined by `options("SpatialExperiment.remote.cache.path")`, otherwise it defaults to a subdirectory in the R temporary directory. Any further named arguments in `...` are passed to `image_read`.

`as.raster(x, ...)` is the same as `imgRaster(x, ...)`.

In-memory caching

For `StoredSpatialImage` and `RemoteSpatialImage` objects, loading the image with `imgRaster` will automatically store the loaded raster object in an in-memory cache. Any subsequent `imgRaster` call will retrieve the raster from the cache, avoiding costly retrieval from the file system.

The cache policy is to evict the least recently used images when a new image would be added that exceeds the maximum cache size. If the new image by itself exceeds the maximum cache size, all images are evicted from the cache to trigger garbage collection and free up memory.

By default, the maximum size of the cache is 4 GB. This can be modified by setting `options("SpatialExperiment.cache.size")` to some number of bytes, e.g., `2^32`.

Other methods

`dim(x)` will return an integer vector of length 2, containing the width and height of the image in pixels. Note that this calls `imgRaster` under the hood and thus may interact with the file and memory caches as described above.

For any `SpatialImage` `x`, `as(x, "LoadedSpatialImage")` will create a `LoadedSpatialImage` containing an in-memory raster object.

For a `RemoteSpatialImage` `x`, `as(x, "StoredSpatialImage")` will create a `StoredSpatialImage` pointing to the file cache location.

Author(s)

Aaron Lun

Examples

```
path <- system.file(
  "extdata", "10xVisium", "section1", "spatial",
  "tissue_lowres_image.png", package="SpatialExperiment")

spi <- SpatialImage(path)
plot(imgRaster(spi))

# the following operations all use the cache
# so there is no need to reload the image
nrow(spi)
ncol(spi)
plot(as.raster(spi))

# coercing to an explicitly in-memory raster
spi <- as(spi, "LoadedSpatialImage")
plot(as.raster(spi))
```

SpatialImage-misc*Miscellaneous SpatialImage methods*

Description

Miscellaneous methods for the [SpatialImage](#) class that do not fit into any other documentation category such as, for example, show methods.

Usage

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

Arguments

object a SpatialImage object

Value

none

Author(s)

Helena L. Crowell

Index

[, SpatialExperiment, ANY, ANY, ANY-method
(SpatialExperiment-subset), 18

addImg (imgData-methods), 2
addImg, SpatialExperiment-method
(imgData-methods), 2

as.raster, 19

assay, 7

assays, 7, 9

cbind, 13

cbind, SingleCellExperiment-method
(SpatialExperiment-combine), 12

cbind, SpatialExperiment-method
(SpatialExperiment-combine), 12

coerce, RemoteSpatialImage, StoredSpatialImage-method
(SpatialImage-class), 18

coerce, SpatialImage, LoadedSpatialImage-method
(SpatialImage-class), 18

colData, 8, 9

colData (SpatialExperiment-colData), 11

colData<- (SpatialExperiment-colData),
11

colData<-, SpatialExperiment, DataFrame-method
(SpatialExperiment-colData), 11

colData<-, SpatialExperiment, NULL-method
(SpatialExperiment-colData), 11

combine, 9

DataFrame, 6, 8, 9, 11, 15

dim, SpatialImage-method
(SpatialImage-class), 18

getImg (imgData-methods), 2
getImg, SpatialExperiment-method
(imgData-methods), 2

image_read, 19

imgData, 6, 8, 9, 12, 18

imgData (SpatialExperiment-methods), 14

imgData, SpatialExperiment-method
(SpatialExperiment-methods), 14

imgData-methods, 2

imgData<- (SpatialExperiment-methods),
14

imgData<-, SpatialExperiment, DataFrame-method
(SpatialExperiment-methods), 14

imgData<-, SpatialExperiment, NULL-method
(SpatialExperiment-methods), 14

imgRaster (SpatialImage-class), 18

imgRaster, LoadedSpatialImage-method
(SpatialImage-class), 18

imgRaster, RemoteSpatialImage-method
(SpatialImage-class), 18

imgRaster, SpatialExperiment-method
(imgData-methods), 2

imgRaster, StoredSpatialImage-method
(SpatialImage-class), 18

imgRaster<- (SpatialImage-class), 18

imgRaster<-, LoadedSpatialImage-method
(SpatialImage-class), 18

imgSource (SpatialImage-class), 18

imgSource, LoadedSpatialImage-method
(SpatialImage-class), 18

imgSource, RemoteSpatialImage-method
(SpatialImage-class), 18

imgSource, SpatialExperiment-method
(imgData-methods), 2

imgSource, StoredSpatialImage-method
(SpatialImage-class), 18

imgSource<- (SpatialImage-class), 18

imgSource<-, RemoteSpatialImage, character-method
(SpatialImage-class), 18

imgSource<-, StoredSpatialImage, character-method
(SpatialImage-class), 18

LoadedSpatialImage-class
(SpatialImage-class), 18

molecules, 9

molecules (SpatialExperiment-assays), 7
 molecules, SpatialExperiment-method
 (SpatialExperiment-assays), 7
 molecules<- (SpatialExperiment-assays),
 7
 molecules<-, SpatialExperiment-method
 (SpatialExperiment-assays), 7

 rbind, 13
 read10xCounts, 4
 read10xVisium, 4, 9
 readImgData, 6, 9
 RemoteSpatialImage-class
 (SpatialImage-class), 18
 rmvImg (imgData-methods), 2
 rmvImg, SpatialExperiment-method
 (imgData-methods), 2
 rowData, 9

 scaleFactors, 9
 scaleFactors
 (SpatialExperiment-methods), 14
 scaleFactors, SpatialExperiment-method
 (SpatialExperiment-methods), 14
 show, SpatialExperiment-method
 (SpatialExperiment-misc), 17
 show, SpatialImage-method
 (SpatialImage-misc), 20
 SingleCellExperiment, 8, 9
 spatialCoords, 8, 9, 13, 15, 18
 spatialCoords
 (SpatialExperiment-methods), 14
 spatialCoords, SpatialExperiment-method
 (SpatialExperiment-methods), 14
 spatialCoords<-
 (SpatialExperiment-methods), 14
 spatialCoords<-, SpatialExperiment, matrix-method
 (SpatialExperiment-methods), 14
 spatialCoords<-, SpatialExperiment, NULL-method
 (SpatialExperiment-methods), 14
 spatialCoordsNames, 9
 spatialCoordsNames
 (SpatialExperiment-methods), 14
 spatialCoordsNames, SpatialExperiment-method
 (SpatialExperiment-methods), 14
 spatialCoordsNames<-
 (SpatialExperiment-methods), 14
 spatialCoordsNames<-, SpatialExperiment, character-method
 (SpatialExperiment-methods), 14

 spatialCoordsNames<-, SpatialExperiment, NULL-method
 (SpatialExperiment-methods), 14
 spatialData, 8, 9, 15, 18
 spatialData
 (SpatialExperiment-methods), 14
 spatialData, SpatialExperiment-method
 (SpatialExperiment-methods), 14
 spatialData<-
 (SpatialExperiment-methods), 14
 spatialData<-, SpatialExperiment, DFrame-method
 (SpatialExperiment-methods), 14
 spatialData<-, SpatialExperiment, NULL-method
 (SpatialExperiment-methods), 14
 spatialDataNames, 9
 spatialDataNames
 (SpatialExperiment-methods), 14
 spatialDataNames, SpatialExperiment-method
 (SpatialExperiment-methods), 14
 spatialDataNames<-
 (SpatialExperiment-methods), 14
 spatialDataNames<-, SpatialExperiment, character-method
 (SpatialExperiment-methods), 14
 SpatialExperiment, 3–7, 11–15, 17, 18
 SpatialExperiment
 (SpatialExperiment-class), 8
 SpatialExperiment-assays, 7
 SpatialExperiment-class, 8
 SpatialExperiment-colData, 11
 SpatialExperiment-combine, 12
 SpatialExperiment-methods, 14
 SpatialExperiment-misc, 17
 SpatialExperiment-subset, 18
 SpatialImage, 9, 20
 SpatialImage (SpatialImage-class), 18
 SpatialImage-class, 18
 SpatialImage-misc, 20
 StoredSpatialImage-class
 (SpatialImage-class), 18