Package 'SpatialExperiment'

April 1, 2025

Version 1.16.0

Title S4 Class for Spatially Resolved -omics Data

Description Defines an S4 class for storing data from spatial -omics experiments. The class extends SingleCellExperiment to support storage and retrieval of additional information from spot-based and molecule-based platforms, including spatial coordinates, images, and image metadata. A specialized constructor function is included for data from the 10x Genomics Visium platform.

URL https://github.com/drighelli/SpatialExperiment

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

License GPL-3

Encoding UTF-8

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

Depends methods, SingleCellExperiment

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

Suggests knitr, rmarkdown, testthat, BiocStyle, BumpyMatrix, DropletUtils

VignetteBuilder knitr

RoxygenNote 7.2.3

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

git_branch RELEASE_3_20

git_last_commit 8798cb0

git_last_commit_date 2024-10-29

Repository Bioconductor 3.20

Date/Publication 2025-03-31

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

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

Contents

`imgData-methods` *Methods for handling image-related data*

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, path = FALSE)

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

## S4 method for signature 'SpatialExperiment'
rotateImg(x, sample_id = NULL, image_id = NULL, degrees = 90)

## S4 method for signature 'SpatialExperiment'
mirrorImg(x, sample_id = NULL, image_id = NULL, axis = c("h", "v"))
```

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
path	logical; for RemoteSpatialImages, TRUE returns the path to the image's cached file, and FALSE its URL. For Stored/LoadedSpatialImages, a path/NA is returned, irrespective of path.
degrees	single numeric in +/-[0,90,...,360] specifying how many degrees to rotate. A negative/positive value corresponds to counter-/clockwise rotation
axis	character string specifying whether to mirror horizontally ("h") or vertically ("v")

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().

rotate/mirrorImg() return a LoadedSpatialImage with modified a raster matrix.

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)

###################
# transformations #
###################

# clockwise rotation
spe1 <- rotateImg(spe,
  degrees = 90) # first image

spe2 <- rotateImg(spe,
  sample_id = TRUE,
  image_id = TRUE,
  degrees = 90) # all images

par(mfrow = c(1, 3))
plot(imgRaster(spe))
plot(imgRaster(spe1))
plot(imgRaster(spe2))

# horizontal/vertical mirroring
spe1 <- mirrorImg(spe, axis = "h")
spe2 <- mirrorImg(spe, axis = "v")

par(mfrow = c(1, 3))
plot(imgRaster(spe))
plot(imgRaster(spe1))
plot(imgRaster(spe2))
```

`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 !is.null(names(samples))
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", "detected", "aligned"
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 (where "raw/filtered" refers to either "raw" or "filtered" to match the ‘data‘ argument.) The base directory "outs/" from Space Ranger can either be included manually in the paths provided in ‘samples‘, or can be ignored; if ignored, it will be added automatically. The ‘.h5‘ files are used if ‘type = "HDF5"‘. (Note that ‘tissue_positions.csv‘ was renamed in Space Ranger v2.0.0.)

sample
· | — outs
· · | — raw/filtered_feature_bc_matrix.h5
· · | — raw/filtered_feature_bc_matrix
· · · | — barcodes.tsv.gz
· · · | — features.tsv.gz
· · · | — matrix.mtx.gz
· · | — spatial
· · · | — scalefactors_json.json
· · · | — tissue_lowres_image.png
· · · | — tissue_positions.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, "outs")

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))

# base directory 'outs/' from Space Ranger can also be omitted
samples2 <- file.path(dir, sample_ids)
(spe2 <- read10xVisium(samples2, sample_ids,
  type = "sparse", data = "raw",
  images = "lowres", load = FALSE))

# tabulate number of spots mapped to tissue
cd <- colData(spe)
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", "outs", "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 spatialCoords and storage of images via imgData.

Arguments

...	Arguments passed to the SingleCellExperiment constructor to fill the slots of the base class.
sample_id	A character sample identifier, which matches the sample_id in imgData. The sample_id will also be stored in a new column in colData, if not already present. Default = sample01.
spatialCoordsNames	A character vector of column names from colData containing spatial coordinates, which will be accessible with spatialCoords. Alternatively, the spatialCoords argument may be provided. If both are provided, spatialCoordsNames is given precedence, and a warning is returned. Default = c("x", "y").
spatialCoords	A numeric matrix containing columns of spatial coordinates, which will be accessible with spatialCoords. Alternatively, spatialCoordsNames may be provided. If both are provided, spatialCoordsNames is given precedence, and a warning is returned.
scaleFactors	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 imageSources. Default = 1.
imgData	Optional DataFrame containing the image data. Alternatively, this can be built from the arguments imageSources and image_id (see Details).
imageSources	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.

spatialDataNames
(Deprecated) A character vector of column names from colData to include in spatialData. Alternatively, the spatialData argument may be provided. If both are provided, spatialDataNames is given precedence, and a warning is returned. (Note: spatialData and spatialDataNames have been deprecated; colData and spatialCoords should be used for all columns. The arguments have been retained for backward compatibility but may be removed in the future.)

spatialData (Deprecated) A DataFrame containing columns to store in spatialData, which must contain at least the columns of spatial coordinates. Alternatively, spatialDataNames may be provided. If both are provided, spatialDataNames is given precedence, and a warning is returned. (Note: spatialData and spatialDataNames have been deprecated; colData and spatialCoords should be used for all columns. The arguments have been retained for backward compatibility but may be removed in the future.)

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. spatialCoords, imgData, and others) represent the 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: 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", "outs"),
    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)),
    rowData = rd,
    colData = DataFrame(xyz),
    spatialCoordsNames = c("pxl_col_in_fullres", "pxl_row_in_fullres"),
    imgData = img,
    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(ng))
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-coercion

SpatialExperiment coercion methods

Description

The SpatialExperiment class inherits from the SingleCellExperiment class making it necessary to coerce between these classes. To do so, we designed two different methods: the traditional as method and the toSpatialExperiment function (recommended). The as method checks if the SingleCellExperiment object has already populated int_colData with three elements: spatialData, spatialCoords, and imgData. It also checks if colData already contains a sample_id. In case these checks pass the new SpatialExperiment will have the same values as the SingleCellExperiment passed object. Otherwise a SpatialExperiment with default values for these slots will be created.

The toSpatialExperiment method expects a SingleCellExperiment object and additional arguments as explained in the related section of this documentation. In case the SingleCellExperiment object has already populated int_colData with spatialData and/or spatialCoords and/or imgData, these will be respectively overwritten in case the arguments spatialData/spatialDataNames and/or spatialCoords/spatialCoordsNames and/or imgData are not NULL.

Arguments

sce	A SingleCellExperiment object.
sample_id	A character sample identifier, which matches the sample_id in imgData. The sample_id will also be stored in a new column in colData, if not already present. Default = sample01.
spatialCoordsNames	A character vector of column names from colData containing spatial coordinates, which will be accessible with spatialCoords. Alternatively, the spatialCoords argument may be provided. If both are provided, spatialCoordsNames is given precedence, and a warning is returned. Default = c("x", "y").
spatialCoords	A numeric matrix containing columns of spatial coordinates, which will be accessible with spatialCoords. Alternatively, spatialCoordsNames may be provided. If both are provided, spatialCoordsNames is given precedence, and a warning is returned.
scaleFactors	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 imageSources. Default = 1.
imgData	Optional DataFrame containing the image data. Alternatively, this can be built from the arguments imageSources and image_id (see Details).
imageSources	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.
spatialDataNames	(Deprecated) A character vector of column names from colData to include in spatialData. Alternatively, the spatialData argument may be provided. If both are provided, spatialDataNames is given precedence, and a warning is returned. (Note: spatialData and spatialDataNames have been deprecated; colData and spatialCoords should be used for all columns. The arguments have been retained for backward compatibility but may be removed in the future.)
spatialData	(Deprecated) A DataFrame containing columns to store in spatialData, which must contain at least the columns of spatial coordinates. Alternatively, spatialDataNames may be provided. If both are provided, spatialDataNames is given precedence, and a warning is returned. (Note: spatialData and spatialDataNames have been deprecated; colData and spatialCoords should be used for all columns. The arguments have been retained for backward compatibility but may be removed in the future.)

Examples

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

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

```
# 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"))

# read in image data
img <- readImgData(
    path = file.path(dir, "spatial"),
    sample_id = "sample01")

## as method
(spe <- as(sce, "SpatialExperiment"))

colData(sce) <- DataFrame(xyz[,c(1:4)])
int_colData(sce)$spatialCoords <- as.matrix(xyz[,c(5,6)])

## Coercing an sce without imgData
(spe <- as(sce, "SpatialExperiment"))

## Coercing an sce with imgData
int_colData(sce)$imgData <- img
(spe <- as(sce, "SpatialExperiment"))

## toSpatialExperiment method
colData(sce) <- DataFrame(xyz)
(spe <- toSpatialExperiment(sce,
    imgData = img,
    spatialCoordsNames = c("pxl_col_in_fullres", "pxl_row_in_fullres"),
    sample_id = "sample01"))
```

`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 `SpatialExperiment`
value	a `DataFrame`

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)

# 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)
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 spatialCoords, imgData, and scaleFactors, as well as methods to rotate and mirror SpatialExperiment objects and their spatial coordinates.

Usage

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

## 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 replacement method for signature 'SpatialExperiment,`NULL`'
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'
x$name

## 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

x	A SpatialExperiment object.
value	Replacement value for replacement methods.
sample_id	Logical value or character vector specifying sample identifier(s) for scaleFactors. Default = TRUE (all samples).
image_id	Logical value or character vector specifying image identifier(s) for scaleFactors. Default = TRUE (all images).
name	The name of the colData column to extract.

Details

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

Note: spatialData and spatialDataNames (previously used to store a subset of columns from colData) have been deprecated. All columns should be stored in either spatialCoords (numeric matrix containing spatial coordinates) or colData (all other columns). The spatialData/spatialDataNames functionality has been retained for backward compatibility but may be removed in the future.

See rotateCoords, mirrorCoords, rotateObject, or mirrorObject for details on methods to rotate and mirror SpatialExperiment objects and their spatialCoords.

Value

Return value varies depending on method, as described below.

spatialData and spatialCoords methods

`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 names of the colData associated with the spatial information, which are stored in the int_metadata.

`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)

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

# 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)
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-rotate-mirror

Methods for spatial attributes

Description

The SpatialExperiment class provides methods to rotate and mirror SpatialExperiment objects and their spatialCoords.

Usage

```
## S4 method for signature 'SpatialExperiment'
rotateCoords(x, sample_id = NULL, degrees = 90, warn = TRUE)

## S4 method for signature 'SpatialExperiment'
mirrorCoords(x, sample_id = NULL, axis = c("h", "v"), warn = TRUE)

## S4 method for signature 'SpatialExperiment'
rotateObject(x, sample_id = NULL, image_id = NULL, degrees = 90)

## S4 method for signature 'SpatialExperiment'
mirrorObject(x, sample_id = NULL, image_id = NULL, axis = c("h", "v"))
```

Arguments

x	A SpatialExperiment object.
sample_id	Logical value or character vector specifying sample identifier(s) for scaleFactors. Default = TRUE (all samples).
degrees	single numeric in +/-[0,90,...,360] specifying how many degrees to rotate. A negative/positive value corresponds to counter-/clockwise rotation. Applicable for rotateCoords and rotateObject methods.
warn	Logical value indicating whether to print a warning about mismatches between coordinates and images, possible with the spatialCoords transformation methods rotateCoords and mirrorCoords.
axis	character string specifying whether to mirror horizontally ("h") or vertically ("v"). Applicable for mirrorCoords and mirrorObject methods.
image_id	Logical value or character vector specifying image identifier(s) for scaleFactors. Default = TRUE (all images).

Details

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

Value

Return value varies depending on method, as described below.

spatialCoords transformation methods

`rotateCoords(x, sample_id, degrees, warn)`: Apply a rotation to the `spatialCoords` of `x`, potentially subsetted to sample `sample_id` (or without subsetting if `sample_id` is `NULL`), by the specified number of `degrees` clockwise. Warn about mismatches with images if `warn`.

`mirrorCoords(x, sample_id, axis, warn)`: Reflect the `spatialCoords` of `x` across either the horizontal or vertical axis, specified by supplying "h" or "v" to the `axis` argument, respectively. Subset `x` to just the sample `sample_id`, if not `NULL`. Warn about mismatches with images if `warn`.

SpatialExperiment transformation wrapper methods

`rotateObject(x, sample_id, image_id, degrees)`: Apply a rotation to the `spatialCoords` and `imgData` of `x`, potentially subsetted to sample `sample_id` (or without subsetting if `sample_id` is `NULL`), by the specified number of `degrees` clockwise. Wrapper around `rotateCoords` and `rotateImg`.

`mirrorObject(x, sample_id, image_id, axis)`: Reflect the `spatialCoords` and `imgData` of `x` across either the horizontal or vertical axis, specified by supplying "h" or "v" to the `axis` argument, respectively. Subset `x` to just the sample `sample_id`, if not `NULL`. Wrapper around `mirrorCoords` and `mirrorImg`.

Author(s)

Nicholas J. Eagles

Examples

```
example(read10xVisium)

# rotateCoords(), mirrorCoords(), rotateObject(), and mirrorObject() return a
# SpatialExperiment, potentially subsetted by sample.

# Subset to just "section1"; rotate coordinates 90 degrees clockwise followed
# by a reflection across the vertical axis
spe_coords <- rotateCoords(spe, sample_id = "section1", degrees = 90)
spe_coords <- mirrorCoords(spe_coords, axis = "v")

# Subset to just "section2"; transform both the imgData() and spatialCoords()
# by a 180-degree rotation then reflection across the vertical axis
spe_wrapper <- rotateObject(spe, sample_id = "section2", degrees = 180)
spe_wrapper <- mirrorObject(spe_wrapper, axis = "v")
```

SpatialExperiment-subset *Subsetting SpatialExperiment objects*

Description

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

Arguments

x	a SpatialExperiment object
i	row indices for subsetting
j	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)
colData(sub)
spatialCoords(sub)
```

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://"`, `"http://"` 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.cach` to some number of bytes, e.g., `2^32`.

Transformations

Two basic image transformations are currently supported for any `SpatialImage` `x`, namely, `rotateImg(x, degrees)` for clockwise (`degrees > 0`) and counter-clockwise (`degrees < 0`) rotation, and `mirrorImg(x, axis)` for horizontal (`axis = "h"`) and vertical (`axis = "v"`) mirroring.

Note that, both `rotateImg()` and `mirrorImg()` operate on the `raster` matrix of the input `SpatialImage`. Thus, any `SpatialImage` will automatically be coerced into a `LoadedSpatialImage` upon rotation/mirroring.

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", "outs", "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))

###################
# transformations #
###################

# (counter-)clockwise rotation
spi1 <- rotateImg(spi, degrees = +90)
spi2 <- rotateImg(spi, degrees = -90)

par(mfrow = c(1, 3))
plot(as.raster(spi))
plot(as.raster(spi1))
plot(as.raster(spi2))

# horizontal/vertical mirroring
spi1 <- mirrorImg(spi, axis = "h")
spi2 <- mirrorImg(spi, axis = "v")

par(mfrow = c(1, 3))
plot(as.raster(spi))
plot(as.raster(spi1))
plot(as.raster(spi2))
```

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 'VirtualSpatialImage'
show(object)
```

Arguments

object	a SpatialImage object

Value

none

Author(s)

Helena L. Crowell

Index