

Package ‘xenLite’

January 4, 2025

Date 2024-09-19

Title Simple classes and methods for managing Xenium datasets

Version 1.1.0

Description Define a relatively light class for managing Xenium data using Bioconductor. Address use of parquet for coordinates, SpatialExperiment for assay and sample data. Address serialization and use of cloud storage.

License Artistic-2.0

Encoding UTF-8

Depends R (>= 4.1)

Suggests knitr, testthat, BiocStyle, yesno, terra,
SpatialFeatureExperiment, SFEDData, tiff

Imports SpatialExperiment, BiocFileCache, Matrix, S4Vectors,
SummarizedExperiment, methods, utils, EBImage, shiny,
HDF5Array, arrow, ggplot2, SingleCellExperiment, TENxIO, dplyr,
graphics, stats

VignetteBuilder knitr

biocViews Infrastructure

RoxygenNote 7.3.2

URL <https://github.com/vjcitn/xenLite>

BugReports <https://github.com/vjcitn/xenLite/issues>

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

git_branch devel

git_last_commit 921c057

git_last_commit_date 2024-10-29

Repository Bioconductor 3.21

Date/Publication 2025-01-03

Author Vincent Carey [aut, cre] (ORCID:
<<https://orcid.org/0000-0003-4046-0063>>)

Maintainer Vincent Carey <stvjc@channing.harvard.edu>

Contents

cacheMtif	2
cacheSfeLungNtx	3
cacheXenLuad	4
cacheXenPdmelLite	4
cacheXenProstLite	5
clipRect	6
demoapp	6
e2sym	7
e79sym	8
getCellBoundaries	8
getCellBoundaries,XenSPEP-method	9
getNucleusBoundaries	9
getNucleusBoundaries,XenSPEP-method	10
getTranscripts	10
getTranscripts,XenSPEP-method	11
ggprepSeg	11
ingest_xen	12
loadGeometry	13
loadGeometry,XenSPEP-method	13
plotXenGgprep	14
printXenGgprep	14
resetParqPaths	15
restoreZipXenSPEP	15
show,XenSPEP-method	16
viewSeg	16
viewSegG2	17
XenSPEP	18
XenSPEP-class	18
zipXenSPEP	19
[,XenSPEP,ANY,ANY,ANY-method	19
Index	21

cacheMtif	<i>cache and/or retrieve path to an ome.tif file for demonstration</i>
-----------	--

Description

cache and/or retrieve path to an ome.tif file for demonstration

Usage

```
cacheMtif(
  cache = BiocFileCache::BiocFileCache(),
  url =
    "https://mgph.osn.xsede.org/bir190004-bucket01/BiocXenData/morphology_focus_0001.ome.tif"
)
```

Arguments

cache defaults to BiocFileCache::BiocFileCache()
 url location where tiff file can be retrieved

Value

path to cached resource

Note

The tiff file was retrieved after running SFEData::XeniumOutput("v2"), and depicts a pancreas tissue sample.

Examples

```
pa <- cacheMtif()
if (!requireNamespace("tiff")) stop("install tiff package to run this example")
x <- tiff::readTIFF(pa)
plot(0, xlim = c(0, 1000), ylim = c(0, 1000), xlab = " ", ylab = " ")
rasterImage(x * 5.5, 0, 0, 1000, 1000)
```

cacheSfeLungNtx	<i>cache and/or retrieve path to an SFE of VI lung demo data from 10x</i>
-----------------	---

Description

cache and/or retrieve path to an SFE of VI lung demo data from 10x

Usage

```
cacheSfeLungNtx(
  cache = BiocFileCache::BiocFileCache(),
  url = "https://mghp.osn.xsede.org/bir190004-bucket01/BiocXenData/sfeLung.zip"
)
```

Arguments

cache defaults to BiocFileCache::BiocFileCache()
 url location where zip file can be retrieved

Value

path to cached resource

Note

Lacks transcript coordinates

cacheXenLuad	<i>cache and/or retrieve path to Xenium Lung Adenocarcinoma example data, zipped SPEP accompanied by parquet</i>
--------------	--

Description

cache and/or retrieve path to Xenium Lung Adenocarcinoma example data, zipped SPEP accompanied by parquet

Usage

```
cacheXenLuad(
  cache = BiocFileCache::BiocFileCache(),
  url = "https://mgdp.osn.xsede.org/bir190004-bucket01/BiocXenData/luad_lite.zip"
)
```

Arguments

cache	defaults to BiocFileCache::BiocFileCache()
url	location where zip file can be retrieved

Value

path to cached resource

Examples

```
if (interactive()) {
  pa <- cacheXenLuad()
  luad <- restoreZipXenSPEP(pa)
  print(luad)
  print(slot(luad, "cellbounds_path"))
  viewSeg(luad, xlim = c(4000, 4500), ylim = c(2000, 2500))
}
```

cacheXenPdmelLite	<i>counts-in-memory version of melanoma 5k dataset</i>
-------------------	--

Description

counts-in-memory version of melanoma 5k dataset

Usage

```
cacheXenPdmelLite(
  cache = BiocFileCache::BiocFileCache(),
  url = "https://mgdp.osn.xsede.org/bir190004-bucket01/BiocXenData/pdmel_lite.zip"
)
```

Arguments

cache defaults to BiocFileCache::BiocFileCache()
url location where zip file can be retrieved

Value

path to cached resource

Examples

```
if (interactive()) {  
  pa <- cacheXenPdmelLite()  
  pdmel_lite <- restoreZipXenSPEP(pa)  
  print(pdmel_lite)  
}
```

cacheXenProstLite *counts-in-memory version of prostate 5k dataset*

Description

counts-in-memory version of prostate 5k dataset

Usage

```
cacheXenProstLite(  
  cache = BiocFileCache::BiocFileCache(),  
  url = "https://mghp.osn.xsede.org/bir190004-bucket01/BiocXenData/prost_lite.zip"  
)
```

Arguments

cache defaults to BiocFileCache::BiocFileCache()
url location where zip file can be retrieved

Value

path to cached resource

Examples

```
if (interactive()) {  
  pa <- cacheXenProstLite()  
  prost_lite <- restoreZipXenSPEP(pa)  
  print(prost_lite)  
}
```

clipRect	<i>restrict XenSPEP to cells with centroids in specified rectangle, also restrict boundary and transcript location features</i>
----------	---

Description

restrict XenSPEP to cells with centroids in specified rectangle, also restrict boundary and transcript location features

Usage

```
clipRect(xsce, xlim, ylim)
```

Arguments

xsce	XenSPEP instance
xlim	numeric(2)
ylim	numeric(2)

Value

XenSPEP instance

Note

Could be too RAM-hungry.

Examples

```
args(clipRect)
```

demoapp	<i>simple app to explore an image</i>
---------	---------------------------------------

Description

simple app to explore an image

Usage

```
demoapp(simple = FALSE)
```

Arguments

simple	logical(1) if TRUE, use a cached tiff for illustration
--------	--

Value

No value returned, run for side effect of app initiation.

Note

Navigate file input control to location of tiffs

Examples

```
if (interactive()) demoapp(simple = TRUE)
```

e2sym	<i>helper function to map ENS ids to symbols</i>
-------	--

Description

helper function to map ENS ids to symbols

Usage

```
e2sym(x)
```

Arguments

x character() mix of Ensembl Ids and other strings; the latter are left unchanged

Value

a vector like x with gene symbols from v79 mapping substituted where possible

Examples

```
e2sym(c("ABC", "ENSG00000213088", "ENSG00000107796", "ENSG00000163017"))
```

`e79sym`*mapping from ENSG to symbols based on EnsDb.Hsapiens.v79*

Description

mapping from ENSG to symbols based on EnsDb.Hsapiens.v79

Usage`e79sym`**Format**`character()`**Value**

character vector

Note

named character vector with values gene symbols, name ENSG ids

Examples

```
data(e79sym)
head(e79sym)
```

`getCellBoundaries`*method for cell boundary extraction*

Description

method for cell boundary extraction

Usage`getCellBoundaries(x)`**Arguments**

`x` instance of XensPEP

Value

reference to ingested parquet

Examples

```
showMethods("getCellBoundaries")
```

getCellBoundaries, XenSPEP-method
method for cell boundary extraction

Description

method for cell boundary extraction

Usage

```
## S4 method for signature 'XenSPEP'  
getCellBoundaries(x)
```

Arguments

x instance of XenSPEP

Value

reference to ingested parquet

getNucleusBoundaries *method for nucleus boundary extraction*

Description

method for nucleus boundary extraction

Usage

```
getNucleusBoundaries(x)
```

Arguments

x instance of XenSPEP

Value

reference to ingested parquet

Examples

```
showMethods("getNucleusBoundaries")
```

getNucleusBoundaries, XenSPEP-method
method for nucleus boundary extraction

Description

method for nucleus boundary extraction

Usage

```
## S4 method for signature 'XenSPEP'  
getNucleusBoundaries(x)
```

Arguments

x instance of XenSPEP

Value

reference to ingested parquet

getTranscripts *method for transcript extraction*

Description

method for transcript extraction

Usage

```
getTranscripts(x)
```

Arguments

x instance of XenSPEP

Value

reference to ingested parquet

Examples

```
showMethods("getTranscripts")
```

getTranscripts,XenSPEP-method
method for transcript extraction

Description

method for transcript extraction

Usage

```
## S4 method for signature 'XenSPEP'
getTranscripts(x)
```

Arguments

x instance of XenSPEP

Value

reference to ingested parquet

Examples

```
showMethods("getTranscripts")
```

ggprepSeg *prepare a XenSPEP for ggplot2 visualization*

Description

prepare a XenSPEP for ggplot2 visualization

Usage

```
ggprepSeg(xsce, xlim = c(5800, 6200), ylim = c(6300, 6700))
```

Arguments

xsce XenSPEP instance
xlim numeric(2)
ylim numeric(2)

Value

a list with components 'bounds' (data.frame including relevant colData rows (all colData variables) and cell boundary coordinates) and 'txdata', a filtered arrow Table.

Note

This is idiosyncratic. Quintiles of `cell_area` (values in `'sizq'`) are produced, and transcript locations are filtered. A more general approach that allows selection of coloring of cells by feature characteristics is needed.

Examples

```
pa <- cacheXenLuad()
luad <- restoreZipXenSPEP(pa)
hh <- ggprepSeg(luad, c(4000, 4500), c(2000, 2500))
ggplot2::ggplot(hh$bounds, ggplot2::aes(
  x = vertex_x, y = vertex_y, group = cell_id,
  colour = sizq, fill = sizq
)) +
  ggplot2::geom_polygon(alpha = .5)
```

ingest_xen	<i>produce a pre-loaded XenSPEP (SpatialExperiment with parquet references)</i>
------------	---

Description

produce a pre-loaded XenSPEP (SpatialExperiment with parquet references)

Usage

```
ingest_xen(folder)
```

Arguments

folder character(1) 'standard' Xenium output folder

Value

instance of XenSPEP

Examples

```
chkns <- function(pkstring) {
  if (!requireNamespace(pkstring)) {
    message(sprintf("install %s to use this feature; returning NULL", pkstring))
    return(NULL)
  }
}
chkns("SFEData")
chkns("HDF5Array")
chkns("SingleCellExperiment")
if (requireNamespace("SFEData")) {
  td <- tempdir()
```

```
z <- SFEData::XeniumOutput("v2", td)
ii <- ingest_xen(file.path(td, "xenium2"))
print(validObject(ii))
plot(SpatialExperiment::spatialCoords(ii), pch = ".")
}
```

loadGeometry	<i>read and bind parquet data to XenSPEP</i>
--------------	--

Description

read and bind parquet data to XenSPEP

Usage

```
loadGeometry(x)
```

Arguments

x instance of XenSPEP

Value

instance of XenSPEP

loadGeometry, XenSPEP-method
<i>read and bind parquet data to XenSPEP</i>

Description

read and bind parquet data to XenSPEP

Usage

```
## S4 method for signature 'XenSPEP'
loadGeometry(x)
```

Arguments

x instance of XenSPEP

Value

instance of XenSPEP

plotXenGgprep	<i>plot method for ggplot2-prepared XenSPEP</i>
---------------	---

Description

plot method for ggplot2-prepared XenSPEP

Usage

```
plotXenGgprep(x, y, ...)
```

Arguments

x	instance of S3 class 'xen_ggprep'
y	not used
...	not used

Value

ggplot

Note

roxygen had problems with this

Examples

```
pa <- cacheXenLuad()
luad <- restoreZipXenSPEP(pa)
hh <- ggprepSeg(luad, c(4000, 4500), c(2000, 2500))
plotXenGgprep(hh)
```

printXenGgprep	<i>print method for ggplot2-prepared XenSPEP</i>
----------------	--

Description

print method for ggplot2-prepared XenSPEP

Usage

```
printXenGgprep(x, ...)
```

Arguments

x	instance of S3 class 'xen_ggprep'
...	not used

Value

operates with cat()

resetParqPaths	<i>utility for dealing with cached Xen_SPEP in temp folder</i>
----------------	--

Description

utility for dealing with cached Xen_SPEP in temp folder

Usage

```
resetParqPaths(xsp, base)
```

Arguments

xsp	instance of XenSPEP
base	folder path where parquet files are found

Value

XenSPEP instance

Note

Will prepend current folder path to parquet-oriented slot values.

restoreZipXenSPEP	<i>use unzip, readRDS, and loadGeometry to restore a XenSPEP</i>
-------------------	--

Description

use unzip, readRDS, and loadGeometry to restore a XenSPEP

Usage

```
restoreZipXenSPEP(zipf, exdir = tempdir())
```

Arguments

zipf	character(1) path to zip file created with 'zipXenSPEP'
exdir	character(1) defaults to tempdir(), where contents are unpacked

Value

instance of XenSPEP

Note

Session folder position will change with `setwd()`, `on.exit` ensures return to position when started.

Examples

```
# used implicitly
if (interactive()) {
  example(cacheXenLuad)
}
```

`show, XenSPEP-method` *display aspects of XenSPEP*

Description

display aspects of XenSPEP

Usage

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

Arguments

`object` instance of XenSPEP

Value

operates with `cat()`

`viewSeg` *naive polygon viewer*

Description

naive polygon viewer

Usage

```
viewSeg(x, xlim, ylim, show_tx = FALSE, ...)
```


Arguments

x	instance of XenSPEP
xlim	numeric(2) ordered vector of max and min on x
ylim	numeric(2) ordered vector of max and min on y
show_tx	logical(1) display transcript locations if TRUE, defaults to FALSE.
...	passed to polygon()

Value

run for side effect of plotting

Note

This is more RAM-sparing than clipRect followed by view.

Examples

```

luad <- cacheXenLuad()
pa <- cacheXenLuad()
luad <- restoreZipXenSPEP(pa)
rownames(luad) <- make.names(SummarizedExperiment::rowData(luad)$Symbol, unique = TRUE)
out <- viewSeg(luad, c(5800, 6300), c(1300, 1800), lwd = .5)
out$ncells

```

viewSegG2	<i>naive polygon viewer, will indicate presence of transcripts for two genes in cells</i>
-----------	---

Description

naive polygon viewer, will indicate presence of transcripts for two genes in cells

Usage

```
viewSegG2(x, xlim, ylim, gene1, gene2, show_tx = FALSE, ...)
```

Arguments

x	instance of XenSPEP
xlim	numeric(2) ordered vector of max and min on x
ylim	numeric(2) ordered vector of max and min on y
gene1	character(1) gene to be checked, cell polygon will be filled if gene has non-zero count
gene2	character(1) gene to be checked, cell polygon will be filled if gene has non-zero count
show_tx	logical(1) display transcript locations if TRUE, defaults to FALSE.
...	passed to polygon()

Value

Primarily for plotting. A list is invisibly returned with elements polys, ncells and call.

Note

This is more RAM-sparing than clipRect followed by view. Colors are pre-assigned for individual and joint occupancies in this draft of this visualizer.

Examples

```
luad <- cacheXenLuad()
pa <- cacheXenLuad()
luad <- restoreZipXenSPEP(pa)
rownames(luad) <- make.names(SummarizedExperiment::rowData(luad)$Symbol, unique = TRUE)
out <- viewSegG2(luad, c(5800, 6300), c(1300, 1800), lwd = .5, gene1 = "CD4", gene2 = "EPCAM")
legend(5800, 1370, fill = c("purple", "cyan", "pink"), legend = c("CD4", "EPCAM", "both"))
out$ncells
```

XenSPEP

XenSPEP (SpatialExperiment with parquet references) constructor

Description

XenSPEP (SpatialExperiment with parquet references) constructor

Usage

```
XenSPEP(folder)
```

Arguments

folder character(1) 'standard' Xenium output folder

Value

instance of XenSPEP

Examples

```
# is not used yet
args(XenSPEP)
```

XenSPEP-class

manage SpatialExperiment with parquet references

Description

manage SpatialExperiment with parquet references

zipXenSPEP	<i>serialize the collection of XenSPEP and parquet with zip</i>
------------	---

Description

serialize the collection of XenSPEP and parquet with zip

Usage

```
zipXenSPEP(xsp, targetfile)
```

Arguments

xsp	instance of XenSPEP with geometry loaded
targetfile	character(1) destination of zip process

Value

output of zip()

Note

a .rds and three parquet files are zipped together for restoration by 'restoreZipXenSPEP'. The outcome is 'paste0(targetfile, ".zip")'.

Examples

```
zipXenSPEP
```

[,XenSPEP,ANY,ANY,ANY-method	<i>formal bracket definition, that leaves parquet geometry information alone.</i>
------------------------------	---

Description

formal bracket definition, that leaves parquet geometry information alone.

Usage

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

Arguments

x	instance of XenSPEP
i	feature selection
j	cell selection
...	passed to SpatialExperiment methods
drop	logical(1)

Value

XenSPEP instance

Note

Gives a message and calls callNextMethod.

Index

- * **datasets**
 - e79sym, [8](#)
 - [,XenSPEP,ANY,ANY,ANY-method, [19](#)
- cacheMtif, [2](#)
- cacheSfeLungNtx, [3](#)
- cacheXenLuad, [4](#)
- cacheXenPdmellite, [4](#)
- cacheXenProstLite, [5](#)
- clipRect, [6](#)
- demoapp, [6](#)
- e2sym, [7](#)
- e79sym, [8](#)
- getCellBoundaries, [8](#)
- getCellBoundaries,XenSPEP-method, [9](#)
- getNucleusBoundaries, [9](#)
- getNucleusBoundaries,XenSPEP-method,
[10](#)
- getTranscripts, [10](#)
- getTranscripts,XenSPEP-method, [11](#)
- ggprepSeg, [11](#)
- ingest_xen, [12](#)
- loadGeometry, [13](#)
- loadGeometry,XenSPEP-method, [13](#)
- plotXenGgprep, [14](#)
- printXenGgprep, [14](#)
- resetParqPaths, [15](#)
- restoreZipXenSPEP, [15](#)
- show,XenSPEP-method, [16](#)
- viewSeg, [16](#)
- viewSegG2, [17](#)
- XenSPEP, [18](#)
- XenSPEP-class, [18](#)
- zipXenSPEP, [19](#)