Package ‘XeniumIO’

July 19, 2025

Type Package

Title Import and represent Xenium data from the 10X Xenium Analyzer

Version 1.1.3

Description The package allows users to readily import spatial data obtained from the 10X Xenium Analyzer pipeline. Supported formats include 'parquet', 'h5', and 'mtx' files. The package mainly represents data as SpatialExperiment objects.

License Artistic-2.0

Depends TENxIO, R (>= 4.5.0)

Imports BiocBaseUtils, BiocGenerics, BiocIO, jsonlite, methods, S4Vectors, SingleCellExperiment, SpatialExperiment, SummarizedExperiment, VisiumIO

Suggests arrow, BiocFileCache, BiocStyle, knitr, rmarkdown, tinytest

biocViews Software, Infrastructure, DataImport, SingleCell, Spatial

VignetteBuilder knitr

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

BugReports https://github.com/waldronlab/XeniumIO/issues

URL https://github.com/waldronlab/XeniumIO

Collate 'XeniumFile.R' 'TENxXenium-class.R' 'XeniumIO-package.R' 'utilities.R'

Date 2025-04-04

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

git_branch devel

git_last_commit 94e988c

git_last_commit_date 2025-07-03

Repository Bioconductor 3.22

Date/Publication 2025-07-18

Author Marcel Ramos [aut, cre] (ORCID: <https://orcid.org/0000-0002-3242-0582>), Dario Righelli [ctb], Estella Dong [ctb]

Maintainer Marcel Ramos <marcel.ramos@sph.cuny.edu>

Contents

`XeniumIO-package` *XeniumIO: Import and represent Xenium data from the 10X Xenium Analyzer*

Description

The package allows users to readily import spatial data obtained from the 10X Xenium Analyzer pipeline. Supported formats include 'parquet', 'h5', and 'mtx' files. The package mainly represents data as SpatialExperiment objects.

Author(s)

Maintainer: Marcel Ramos `<marcel.ramos@sph.cuny.edu>` (ORCID)

Other contributors:

- Dario Righelli [contributor]
- Estella Dong [contributor]

See Also

Useful links:

- `https://github.com/waldronlab/XeniumIO`
- Report bugs at `https://github.com/waldronlab/XeniumIO/issues`

`TENxXenium-class` *A class to represent Xenium output data*

Description

This class is a composed class of TENxFileList which can contain a list of TENxFile objects for the cell-feature matrix. It is meant to handle a single Xenium sample from 10X Genomics.

Usage

```
TENxXenium(
  resources,
  xeniumOut,
  sample_id = "sample01",
  format = c("mtx", "h5"),
  boundaries_format = c("parquet", "csv.gz"),
  spatialCoordsNames = c("x_centroid", "y_centroid"),
  ...
)

## S4 method for signature 'TENxXenium,ANY,ANY'
import(con, format, text, ...)
```

Arguments

`resources`	A TENxFileList object or a file path to the tarball containing the matrix / assay data resources.
`xeniumOut`	`character(1)` The path to the Xenium output directory.
`sample_id`	`character(1)` A single string specifying the sample ID.
`format`	The format of the output. If missing and con is a file name, the format is derived from the file extension. This argument is unnecessary when con is a derivative of BiocFile.
`boundaries_format`	`character(1)` Either "parquet" or "csv.gz" to specify the file extension of the boundaries file. Default is "parquet".
`spatialCoordsNames`	`character()` A vector of strings specifying the names of the columns in the spatial data containing the spatial coordinates.
`...`	In the constructor, additional arguments passed to TENxFileList; otherwise, not used.
`con`	The connection from which data is loaded or to which data is saved. If this is a `character` vector, it is assumed to be a file name and a corresponding file connection is created and then closed after exporting the object. If it is a BiocFile derivative, the data is loaded from or saved to the underlying resource. If missing, the function will return the output as a character vector, rather than writing to a connection.
`text`	If con is missing, this can be a character vector directly providing the string data to import.

Details

Note that one can provide a `ref` argument to `import` method which will get passed to the internal `splitAltExps` operation. This allows one to set a `mainExpName` in the output object.

Value

A SpatialExperiment object

Functions

- `import(con = TENxXenium, format = ANY, text = ANY)`: Import Xenium Analyzer data

Slots

resources A TENxFileList or TENxH5 object containing the cell feature matrix.

boundaries Either a TENxSpatialParquet or TENxSpatialCSV object containing the spatial boundaries data.

coordNames character() A vector specifying the names of the columns in the spatial data containing the spatial coordinates.

sampleId character(1) A scalar specifying the sample identifier.

colData TENxSpatialParquet A TENxSpatialParquet object containing the spatial coordinates data.

metadata XeniumFile A XeniumFile object containing the metadata information.

See Also

https://www.10xgenomics.com/support/software/xenium-onboard-analysis/latest/analysis/xoa-output-understanding-outputs

Examples

```
showClass("TENxXenium")

zipfile <- paste0(
    "https://mghp.osn.xsede.org/bir190004-bucket01/BiocXenDemo/",
    "Xenium_Prime_MultiCellSeg_Mouse_Ileum_tiny_outs.zip"
)
destfile <- XeniumIO:::.cache_url_file(zipfile)
outfold <- file.path(
    tempdir(), tools::file_path_sans_ext(basename(zipfile))
)
if (!dir.exists(outfold))
    dir.create(outfold, recursive = TRUE)
unzip(
    zipfile = destfile, exdir = outfold, overwrite = FALSE
)
TENxXenium(xeniumOut = outfold) |>
    import(ref = "Gene Expression")
```

XeniumFile-class *A minimal class to represent Xenium metadata*

Description

This class is a minimal class to represent Xenium metadata. It is dedicated to importing experiment.xenium metadata files. It uses the jsonlite package to import the metadata.

Usage

```
XeniumFile(resource)

## S4 method for signature 'XeniumFile,ANY,ANY'
import(con, format, text, ...)
```

Arguments

`resource`	`character(1)` The path to the Xenium metadata file.
`con`	The connection from which data is loaded or to which data is saved. If this is a `character` vector, it is assumed to be a file name and a corresponding file connection is created and then closed after exporting the object. If it is a BiocFile derivative, the data is loaded from or saved to the underlying resource. If missing, the function will return the output as a character vector, rather than writing to a connection.
`format`	The format of the output. If missing and `con` is a file name, the format is derived from the file extension. This argument is unnecessary when `con` is a derivative of BiocFile.
`text`	If `con` is missing, this can be a character vector directly providing the string data to import.
`...`	Parameters to pass to the format-specific method.

Value

A XeniumFile object

Functions

- `import(con = XeniumFile, format = ANY, text = ANY)`: Import Xenium metadata

Examples

```
showClass("XeniumFile")
```

Index