Package 'zellkonverter'

November 9, 2024

Title Conversion Between scRNA-seq Objects

Version 1.16.0

Date 2024-10-18

Description Provides methods to convert between Python AnnData objects and SingleCellExperiment objects. These are primarily intended for use by downstream Bioconductor packages that wrap Python methods for single-cell data analysis. It also includes functions to read and write H5AD files used for saving AnnData objects to disk.

License MIT + file LICENSE

URL https://github.com/theislab/zellkonverter

BugReports https://github.com/theislab/zellkonverter/issues

Imports basilisk, cli, DelayedArray, Matrix, methods, reticulate, S4Vectors, SingleCellExperiment (>= 1.11.6), SummarizedExperiment, utils

Suggests anndata, BiocFileCache, BiocStyle, covr, HDF5Array, knitr, pkgload, rhdf5 (>= 2.45.1), rmarkdown, scRNAseq, spelling, testthat, withr

VignetteBuilder knitr

biocViews SingleCell, DataImport, DataRepresentation

Encoding UTF-8

Language en-GB

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

StagedInstall no

Collate 'AnnData2SCE.R' 'SCE2AnnData.R' 'ui.R' 'basilisk.R' 'read.R' 'reticulate.R' 'utils.R' 'validation.R' 'write.R' 'zellkonverter-package.R'

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

git_branch RELEASE_3_20

git_last_commit d492566

git_last_commit_date 2024-10-29

Repository Bioconductor 3.20

Date/Publication 2024-11-08

Author Luke Zappia [aut, cre] (<https://orcid.org/0000-0001-7744-8565>),
Aaron Lun [aut] (<https://orcid.org/0000-0002-3564-4813>),
Jack Kamm [ctb] (<https://orcid.org/0000-0003-2412-756X>),
Robrecht Cannoodt [ctb] (<https://orcid.org/0000-0003-3641-729X>, rcannood),
Gabriel Hoffman [ctb] (<https://orcid.org/0000-0002-0957-0224>, GabrielHoffman)

Maintainer Luke Zappia <luke@lazappi.id.au>

Contents

`zellkonverter-package` *zellkonverter: Conversion Between scRNA-seq Objects*

Description

Provides methods to convert between Python AnnData objects and SingleCellExperiment objects. These are primarily intended for use by downstream Bioconductor packages that wrap Python methods for single-cell data analysis. It also includes functions to read and write H5AD files used for saving AnnData objects to disk.

Author(s)

Maintainer: Luke Zappia <luke@lazappi.id.au> (ORCID)

Authors:

- Aaron Lun <infinite.monkeys.with.keyboards@gmail.com> (ORCID)

Other contributors:

- Jack Kamm <jackkamm@gmail.com> (ORCID) [contributor]
- Robrecht Cannoodt <rcannood@gmail.com> (ORCID) (rcannood) [contributor]
- Gabriel Hoffman <gabriel.hoffman@mssm.edu> (ORCID) (GabrielHoffman) [contributor]

See Also

Useful links:

- https://github.com/theislab/zellkonverter
- Report bugs at https://github.com/theislab/zellkonverter/issues

AnnData-Conversion *Convert between AnnData and SingleCellExperiment*

Description

Conversion between Python AnnData objects and SingleCellExperiment objects.

Usage

```
AnnData2SCE(
  adata,
  X_name = NULL,
  layers = TRUE,
  uns = TRUE,
  var = TRUE,
  obs = TRUE,
  varm = TRUE,
  obsm = TRUE,
  varp = TRUE,
  obsp = TRUE,
  raw = FALSE,
  skip_assays = FALSE,
  hdf5_backed = TRUE,
  verbose = NULL
)

SCE2AnnData(
  sce,
  X_name = NULL,
  assays = TRUE,
  colData = TRUE,
  rowData = TRUE,
  varm = TRUE,
  reducedDims = TRUE,
  metadata = TRUE,
  colPairs = TRUE,
  rowPairs = TRUE,
  skip_assays = FALSE,
  verbose = NULL
)
```

Arguments

adata	A **reticulate** reference to a Python AnnData object.
X_name	For SCE2AnnData() name of the assay to use as the primary matrix (X) of the AnnData object. If NULL, the first assay of sce will be used by default. For AnnData2SCE() name used when saving X as an assay. If NULL looks for an X_name value in uns, otherwise uses "X".

`layers`, `uns`, `var`, `obs`, `varm`, `obsm`, `varp`, `obsp`, `raw`
: Arguments specifying how these slots are converted. If `TRUE` everything in that slot is converted, if `FALSE` nothing is converted and if a character vector only those items or columns are converted.

`skip_assays`
: Logical scalar indicating whether to skip conversion of any assays in `sce` or `adata`, replacing them with empty sparse matrices instead.

`hdf5_backed`
: Logical scalar indicating whether HDF5-backed matrices in `adata` should be represented as HDF5Array objects. This assumes that `adata` is created with `backed="r"`.

`verbose`
: Logical scalar indicating whether to print progress messages. If `NULL` uses `getOption("zellkonverter.verbose")`.

`sce`
: A SingleCellExperiment object.

`assays`, `colData`, `rowData`, `reducedDims`, `metadata`, `colPairs`, `rowPairs`
: Arguments specifying how these slots are converted. If `TRUE` everything in that slot is converted, if `FALSE` nothing is converted and if a character vector only those items or columns are converted.

Details

These functions assume that an appropriate Python environment has already been loaded. As such, they are largely intended for developer use, most typically inside a **basilisk** context.

The conversion is not entirely lossless. The current mapping is shown below (also at https://tinyurl.com/AnnData2SCE):

	SingleCellExperiment				AnnData		
	R object used by Bioconductor packages				Python object used by Scanpy and related packages		
	Rows contain features and columns contain cells				*Rows contain observations (cells) and columns contain variables (features)*		
IMPLEMENTED					X	Numpy array	*Primary matrix of expression data*
IMPLEMENTED	*Each element is a matrix of expression values with the same dimensions*	SimpleList of matrices	**assays**	⇄	**layers**	Dictionary of arrays	*Dictionary-like object where each element is an array of expression values with the same dimensions as X*
IMPLEMENTED	*Any unstructured data*	list	**metadata**	⇔	**uns**	OrderedDict	*Unstructured annotation*
IMPLEMENTED	*Names of rows (cells)*	vector	**colnames**	⇄	**obs_names**	Pandas index	*Names of observations (cells)*
IMPLEMENTED	*Columns describe annotations of the columns (cells)*	DataFrame	**colData**	⇄	**obs**	Pandas DataFrame	*One-dimensional annotations of the observations (cells)*
IMPLEMENTED	*Each element is a matrix where the number of rows is equal to the number of cells and each column is a dimension*	List of matrices	**reducedDims**	⇄	**obsm**	Dictionary of arrays	*Dictionary-like object where each element is an array where the number of rows is equal to the number of observations (cells)*
IMPLEMENTED	*Relationships between columns (cells)*	List of SelfHits	**colPairs**	⇄	**obsp**	Dictionary of arrays	*Dictionary-like object where each element is a square array containing annotations between observations (cells)*
IMPLEMENTED	*Names of rows (features)*	vector	**rownames**	⇄	**var_names**	Pandas index	*Names of variables (features)*
IMPLEMENTED	*Columns describe annotations of the rows (features)*	DataFrame	**rowData**	⇄	**vars**	Pandas DataFrame	*One-dimensional annotations of the variables (features)*
IMPLEMENTED				⇄	**varm**	Dictionary of arrays	*Dictionary-like object where each element is an array where the number of rows is equal to the number of variables (features)*
IMPLEMENTED	*Relationships between rows (features)*	List of SelfHits	**rowPairs**	⇄	**varp**	Dictionary of arrays	*Dictionary-like object where each element is a square array containing annotations between variables (features)*
NOT IMPLEMENTED	*Nested SingleCellExperiments with information about alternative feature sets*	List of SingleCellExperiments	**altExp**		**raw**	AnnData	*Raw version of X and var prior to any filtering. Is not indexed as part of the object.*
NOT IMPLEMENTED	*Internal unstructured data that is not meant to be modified by users*	list	**int_metadata**				
NOT IMPLEMENTED	*Internal annotation for columns (cells) that is not meant to be modified by users*	DataFrame	**int_colData**				
NOT IMPLEMENTED	*Internal annotation for rows (features) that is not meant to be modified by users*	DataFrame	**int_elementMetadata**				

In `SCE2AnnData()`, matrices are converted to a **numpy**-friendly format. Sparse matrices are converted to dgCMatrix objects while all other matrices are converted into ordinary matrices. If `skip_assays = TRUE`, empty sparse matrices are created instead and the user is expected to fill in the assays on the Python side.

For `AnnData2SCE()`, a warning is raised if there is no corresponding R format for a matrix in the AnnData object, and an empty sparse matrix is created instead as a placeholder. If `skip_assays = NA`, no warning is emitted but variables are created in the `int_metadata()` of the output to specify which assays were skipped.

If `skip_assays = TRUE`, empty sparse matrices are created for all assays, regardless of whether they might be convertible to an R format or not. In both cases, the user is expected to fill in the assays on the R side, see `readH5AD()` for an example.

We attempt to convert between items in the SingleCellExperiment `metadata()` slot and the `AnnData` `uns` slot. If an item cannot be converted a warning will be raised.

Values stored in the `varm` slot of an `AnnData` object are stored in a column of `rowData()` in a SingleCellExperiment as a DataFrame of matrices. If this column is present an attempt is made to transfer this information when converting from SingleCellExperiment to `AnnData`.

Value

`AnnData2SCE()` will return a SingleCellExperiment containing the equivalent data from `adata`.

`SCE2AnnData()` will return a **reticulate** reference to an AnnData object containing the content of `sce`.

Author(s)

Luke Zappia

Aaron Lun

See Also

`writeH5AD()` and `readH5AD()` for dealing directly with H5AD files.

Examples

```
if (requireNamespace("scRNAseq", quietly = TRUE)) {
    library(basilisk)
    library(scRNAseq)
    seger <- SegerstolpePancreasData()

    # These functions are designed to be run inside
    # a specified Python environment
    roundtrip <- basiliskRun(fun = function(sce) {
        # Convert SCE to AnnData:
        adata <- zellkonverter::SCE2AnnData(sce)

        # Maybe do some work in Python on 'adata':
        # BLAH BLAH BLAH

        # Convert back to an SCE:
        zellkonverter::AnnData2SCE(adata)
    }, env = zellkonverterAnnDataEnv(), sce = seger)
}
```

AnnData-Environment *AnnData environment*

Description

The Python environment used by **zellkonverter** for interfacing with the **anndata** Python library (and H5AD files) is described by the dependencies in returned by `AnnDataDependencies()`. The `zellkonverterAnnDataEnv()` functions returns the `basilisk::BasiliskEnvironment()` containing these dependencies used by **zellkonverter**. Allowed versions of **anndata** are available in `.AnnDataVersions`.

Usage

```
.AnnDataVersions

AnnDataDependencies(version = .AnnDataVersions)

zellkonverterAnnDataEnv(version = .AnnDataVersions)
```

Arguments

`version`	A string giving the version of the **anndata** Python library to use. Allowed values are available in `.AnnDataVersions`. By default the latest version is used.

Format

For `.AnnDataVersions` a character vector containing allowed **anndata** version strings.

Details

Using Python environments:

When a **zellkonverter** is first run a conda environment containing all of the necessary dependencies for that version with be instantiated. This will not be performed on any subsequent run or if any other **zellkonverter** function has been run prior with the same environment version.

By default the **zellkonverter** conda environment will become the shared R Python environment if one does not already exist. When one does exist (for example when a **zellkonverter** function has already been run using a a different environment version) then a separate environment will be used. See `basilisk::setBasiliskShared()` for more information on this behaviour. Note the when the environment is not shared progress messages are lost.

Development:

The `AnnDataDependencies()` function is exposed for use by other package developers who want an easy way to define the dependencies required for creating a Python environment to work with AnnData objects, most typically within a **basilisk** context. For example, we can simply combine this vector with additional dependencies to create a **basilisk** environment with Python package versions that are consistent with those in **zellkonverter**.

If you want to run code in the exact environment used by **zellkonverter** this can be done using `zellkonverterAnnDataEnv()` in combination with `basilisk::basiliskStart()` and/or `basilisk::basiliskRun()`. Please refer to the **basilisk** documentation for more information on using these environments.

Value

For `AnnDataDependencies` a character vector containing the pinned versions of all Python packages to be used by `zellkonverterAnnDataEnv()`.

For `zellkonverterAnnDataEnv` a `basilisk::BasiliskEnvironment()` containing **zellkonverter**'s AnnData Python environment.

Author(s)

Luke Zappia

Aaron Lun

Examples

```
.AnnDataVersions

AnnDataDependencies()
AnnDataDependencies(version = "0.7.6")

cl <- basilisk::basiliskStart(zellkonverterAnnDataEnv())
anndata <- reticulate::import("anndata")
basilisk::basiliskStop(cl)
```

---

expectSCE *Expect SCE*

Description

Test that a SingleCellExperiment matches an expected object. Designed to be used inside `testthat::test_that()` during package testing.

Usage

```
expectSCE(sce, expected)
```

Arguments

sce	A SingleCellExperiment object.
expected	A template SingleCellExperiment object to compare to.

Value

`TRUE` invisibly if checks pass

Author(s)

Luke Zappia

---

r-py-conversion *Convert between Python and R objects*

Description

Convert between Python and R objects

Usage

```
## S3 method for class 'numpy.ndarray'
py_to_r(x)
```

Arguments

x	A Python object.

Details

These functions are extensions of the default conversion functions in the `reticulate` package for the following reasons:

- `numpy.ndarray` - Handle conversion of **numpy** recarrays
- `pandas.core.arrays.masked.BaseMaskedArray` - Handle conversion of **pandas** arrays (used when by `AnnData` objects when there are missing values)
- `pandas.core.arrays.categorical.Categorical` - Handle conversion of **pandas** categorical arrays

Value

An R object, as converted from the Python object.

Author(s)

Luke Zappia

See Also

`reticulate::py_to_r()` for the base `reticulate` functions

`readH5AD` *Read H5AD*

Description

Reads a H5AD file and returns a SingleCellExperiment object.

Usage

```
readH5AD(
  file,
  X_name = NULL,
  use_hdf5 = FALSE,
  reader = c("python", "R"),
  version = NULL,
  verbose = NULL,
  ...
)
```

Arguments

`file`	String containing a path to a `.h5ad` file.
`X_name`	Name used when saving `X` as an assay. If `NULL` looks for an `X_name` value in `uns`, otherwise uses `"X"`.
`use_hdf5`	Logical scalar indicating whether assays should be loaded as HDF5-based matrices from the **HDF5Array** package.
`reader`	Which HDF5 reader to use. Either `"python"` for reading with the **anndata** Python package via **reticulate** or `"R"` for **zellkonverter**'s native R reader.

`version` A string giving the version of the **anndata** Python library to use. Allowed values are available in `.AnnDataVersions`. By default the latest version is used.

`verbose` Logical scalar indicating whether to print progress messages. If `NULL` uses `getOption("zellkonverter.verbose")`.

`...` Arguments passed on to `AnnData2SCE`

`layers,uns,var,obs,varm,obsm,varp,obsp,raw` Arguments specifying how these slots are converted. If `TRUE` everything in that slot is converted, if `FALSE` nothing is converted and if a character vector only those items or columns are converted.

`skip_assays` Logical scalar indicating whether to skip conversion of any assays in `sce` or `adata`, replacing them with empty sparse matrices instead.

Details

Setting `use_hdf5 = TRUE` allows for very large datasets to be efficiently represented on machines with little memory. However, this comes at the cost of access speed as data needs to be fetched from the HDF5 file upon request.

Setting `reader = "R"` will use an experimental native R reader instead of reading the file into Python and converting the result. This avoids the need for a Python environment and some of the issues with conversion but is still under development and is likely to return slightly different output.

See AnnData-Environment for more details on **zellkonverter** Python environments.

Value

A SingleCellExperiment object is returned.

Author(s)

Luke Zappia

Aaron Lun

See Also

`writeH5AD()`, to write a SingleCellExperiment object to a H5AD file.

`AnnData2SCE()`, for developers to convert existing AnnData instances to a SingleCellExperiment.

Examples

```
library(SummarizedExperiment)

file <- system.file("extdata", "krumsiek11.h5ad", package = "zellkonverter")
sce <- readH5AD(file)
class(assay(sce))

sce2 <- readH5AD(file, use_hdf5 = TRUE)
class(assay(sce2))

sce3 <- readH5AD(file, reader = "R")
```

setZellkonverterVerbose *Set zellkonverter verbose*

Description

Set the zellkonverter verbosity option

Usage

```
setZellkonverterVerbose(verbose = TRUE)
```

Arguments

verbose	Logical value for the verbosity option.

Details

Running `setZellkonverterVerbose(TRUE)` will turn on **zellkonverter** progress messages by default without having to set `verbose = TRUE` in each function call. This is done by setting the `"zellkonverter.verbose"` option. Running `setZellkonverterVerbose(FALSE)` will turn default verbosity off.

Value

The value of getOption("zellkonverter.verbose") invisibly

Examples

```
current <- getOption("zellkonverter.verbose")
setZellkonverterVerbose(TRUE)
getOption("zellkonverter.verbose")
setZellkonverterVerbose(FALSE)
getOption("zellkonverter.verbose")
setZellkonverterVerbose(current)
getOption("zellkonverter.verbose")
```

validateH5ADSCE *Validate H5AD SCE*

Description

Validate a SingleCellExperiment created by `readH5AD()`. Designed to be used inside `testthat::test_that()` during package testing.

Usage

```
validateH5ADSCE(sce, names, missing)
```

Arguments

sce	A SingleCellExperiment object.
names	Named list of expected names. Names are slots and values are vectors of names that are expected to exist in that slot.
missing	Named list of known missing names. Names are slots and values are vectors of names that are expected to not exist in that slot.

Details

This function checks that a SingleCellExperiment contains the expected items in each slot. The main reason for this function is avoid repeating code when testing multiple `.h5ad` files. The following items in `names` and `missing` are recognised:

- `assays` - Assay names
- `colData` - colData column names
- `rowData` - rowData column names
- `metadata` - metadata names
- `redDim` - Reduced dimension names
- `varm` - Column names of the `varm` rowData column (from the AnnData varm slot)
- `colPairs` - Column pair names
- `rowPairs` - rowData pair names
- `raw_rowData` - rowData columns names in the `raw` altExp
- `raw_varm` - Column names of the raw `varm` rowData column (from the AnnData varm slot)

If an item in `names` or `missing` is `NULL` then it won't be checked. The items in `missing` are checked that they explicitly do not exist. This is mostly for record keeping when something is known to not be converted but can also be useful when the corresponding `names` item is `NULL`.

Value

If checks are successful `TRUE` invisibly, if not other output depending on the context

Author(s)

Luke Zappia

`writeH5AD` *Write H5AD*

Description

Write a H5AD file from a SingleCellExperiment object.

Usage

```
writeH5AD(
  sce,
  file,
  X_name = NULL,
  skip_assays = FALSE,
  compression = c("none", "gzip", "lzf"),
  version = NULL,
  verbose = NULL,
  ...
)
```

Arguments

sce	A SingleCellExperiment object.
file	String containing a path to write the new .h5ad file.
X_name	Name of the assay to use as the primary matrix (X) of the AnnData object. If NULL, the first assay of sce will be used by default.
skip_assays	Logical scalar indicating whether assay matrices should be ignored when writing to file.
compression	Type of compression when writing the new .h5ad file.
version	A string giving the version of the **anndata** Python library to use. Allowed values are available in .AnnDataVersions. By default the latest version is used.
verbose	Logical scalar indicating whether to print progress messages. If NULL uses getOption("zellkonverter.verbose").
...	Arguments passed on to SCE2AnnData
	assays,colData,rowData,reducedDims,metadata,colPairs,rowPairs Arguments specifying how these slots are converted. If TRUE everything in that slot is converted, if FALSE nothing is converted and if a character vector only those items or columns are converted.

Details

Skipping assays:

Setting skip_assays = TRUE can occasionally be useful if the matrices in sce are stored in a format that is not amenable for efficient conversion to a **numpy**-compatible format. In such cases, it can be better to create an empty placeholder dataset in file and fill it in R afterwards.

DelayedArray assays:

If sce contains any **DelayedArray** matrices as assays writeH5AD() will write them to disk using the **rhdf5** package directly rather than via Python to avoid instantiating them in memory. However there is currently an issue which prevents this being done for sparse **DelayedArray** matrices.

Known conversion issues:

Coercion to factors:

The **anndata** package automatically converts some character vectors to factors when saving .h5ad files. This can effect columns of rowData(sce) and colData(sce) which may change type when the .h5ad file is read back into R.

Environment:

See AnnData-Environment for more details on **zellkonverter** Python environments.

Value

A NULL is invisibly returned.

Author(s)

Luke Zappia

Aaron Lun

See Also

readH5AD(), to read a SingleCellExperiment file from a H5AD file.

SCE2AnnData(), for developers to create an AnnData object from a SingleCellExperiment.

Examples

```
# Using the Zeisel brain dataset
if (requireNamespace("scRNAseq", quietly = TRUE)) {
    library(scRNAseq)
    sce <- ZeiselBrainData()

    # Writing to a H5AD file
    temp <- tempfile(fileext = ".h5ad")
    writeH5AD(sce, temp)
}
```

Index