Package 'velociraptor'

July 21, 2025

Title Toolkit for Single-Cell Velocity

Version 1.18.0

Date 2024-10-11

Description This package provides Bioconductor-
friendly wrappers for RNA velocity calculations in single-cell RNA-seq data.
We use the basilisk package to manage Conda environments,
and the zellkonverter package to convert data structures between SingleCellExperiment (R) and AnnData (Python).
The information produced by the velocity methods is stored in the various components of the SingleCellExperiment class.

Depends SummarizedExperiment

Imports methods, stats, Matrix, BiocGenerics, reticulate, S4Vectors,
DelayedArray, basilisk, zellkonverter, scuttle,
SingleCellExperiment, BiocParallel, BiocSingular

Suggests BiocStyle, testthat, knitr, rmarkdown, pkgdown, scran,
scater, scRNAseq, Rtsne, graphics, grDevices, ggplot2, cowplot,
GGally, patchwork, metR

StagedInstall no

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

URL https://github.com/kevinrue/velociraptor

BugReports https://github.com/kevinrue/velociraptor/issues

biocViews SingleCell, GeneExpression, Sequencing, Coverage

VignetteBuilder knitr

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

git_branch RELEASE_3_21

git_last_commit b2c6328

git_last_commit_date 2025-04-15

Repository Bioconductor 3.21

Date/Publication 2025-07-20

Author Kevin Rue-Albrecht [aut, cre] (ORCID:
<https://orcid.org/0000-0003-3899-3872>),
Aaron Lun [aut] (ORCID: <https://orcid.org/0000-0002-3564-4813>),
Charlotte Soneson [aut] (ORCID:
<https://orcid.org/0000-0003-3833-2169>),
Michael Stadler [aut] (ORCID: <https://orcid.org/0000-0002-2269-4934>)

Maintainer Kevin Rue-Albrecht <kevinrue67@gmail.com>

Contents

`velociraptor-package` *velociraptor: Toolkit for Single-Cell Velocity*

Description

This package provides Bioconductor-friendly wrappers for RNA velocity calculations in single-cell RNA-seq data. We use the basilisk package to manage Conda environments, and the zellkonverter package to convert data structures between SingleCellExperiment (R) and AnnData (Python). The information produced by the velocity methods is stored in the various components of the SingleCellExperiment class.

Author(s)

Maintainer: Kevin Rue-Albrecht <kevinrue67@gmail.com> (ORCID)

Authors:

- Aaron Lun <infinite.monkeys.with.keyboards@gmail.com> (ORCID)
- Charlotte Soneson <charlottesoneson@gmail.com> (ORCID)
- Michael Stadler <michael.stadler@fmi.ch> (ORCID)

See Also

Useful links:

- https://github.com/kevinrue/velociraptor
- Report bugs at https://github.com/kevinrue/velociraptor/issues

`embedVelocity` *Project velocities onto an embedding*

Description

Project the velocity vector for each cell onto an existing low-dimensional embedding.

Usage

```
embedVelocity(x, vobj, ...)

## S4 method for signature 'ANY'
embedVelocity(x, vobj, ...)

## S4 method for signature 'SingleCellExperiment'
embedVelocity(x, vobj, ..., use.dimred = 1)
```

Arguments

`x`	A numeric matrix of low-dimensional coordinates, e.g., after t-SNE. Alternatively, a SingleCellExperiment containing such coordinates in its `reducedDims`.
`vobj`	A SingleCellExperiment containing the output of the velocity calculations, typically after running `scvelo`.
`...`	For the generic, further arguments to pass to specific methods. For the ANY method, further arguments to pass to the `velocity_embedding` Python function from **scVelo**. For the SingleCellExperiment method, further arguments to pass to the ANY method.
`use.dimred`	String or integer scalar specifying the reduced dimensions to retrieve from `x`.

Details

This is a simple wrapper around the `scvelo.tools.velocity_embedding` function. Briefly, we construct a cell-cell transition matrix where a cell is more likely to transition to one of its neighbors if its velocity vector is pointing in the same direction as that neighbor. The resulting matrix is then used to compute a weighted average of the positions in `x`, allowing us to compute a velocity in the low-dimensional embedding.

Value

A numeric matrix of the same dimensions as `x`, containing the projected velocity vectors in that embedding.

Author(s)

Aaron Lun

Examples

```
example(scvelo, echo=FALSE) # recycling that example.

# Making up a new embedding.
tsne.results <- matrix(rnorm(2*ncol(out)), ncol=2)

# Projecting the future state of each cell:
projected <- embedVelocity(tsne.results, out)
```

`gridVectors` *Summarize vectors into a grid*

Description

Summarize the velocity vectors into a grid, usually for easy plotting.

Usage

```
gridVectors(x, embedded, ...)

## S4 method for signature 'ANY'
gridVectors(
  x,
  embedded,
  resolution = 40,
  scale = TRUE,
  as.data.frame = TRUE,
  return.intermediates = FALSE
)

## S4 method for signature 'SingleCellExperiment'
gridVectors(x, embedded, ..., use.dimred = 1)
```

Arguments

`x`	A numeric matrix of low-dimensional coordinates, e.g., after t-SNE. Alternatively, a SingleCellExperiment containing such coordinates in its `reducedDims`.
`embedded`	A low-dimensional projection of the velocity vectors into the embedding of `x`. This should be of the same dimensions as `x` and is typically produced by `embedVelocity`.
`...`	For the generic, further arguments to pass to specific methods. For the SingleCellExperiment method, further arguments to pass to the ANY method.
`resolution`	Integer scalar specifying the resolution of the grid, in terms of the number of grid intervals along each axis.

`scale`	Logical scalar indicating whether the averaged vectors should be scaled by the grid resolution.
`as.data.frame`	Logical scalar indicating whether the output should be a data.frame. If `FALSE`, a list of two matrices is returned.
`return.intermediates`	Logical scalar indicating whether intermediate objects should also be returned. This enforces `as.data.frame=FALSE` and throws a warning if it is `TRUE`.
`use.dimred`	String or integer scalar specifying the reduced dimensions to retrieve from `x`.

Details

This partitions the bounding box of `x` into a grid with `resolution` units in each dimension. The locations and vectors of all cells in each block are averaged to obtain a representative of that block. This is most obviously useful for visualization to avoid overplotting of velocity vectors.

If `scale=TRUE`, per-block vectors are scaled so that the median vector length is comparable to the spacing between blocks. This improves visualization when the scales of `x` and `embedded` are not immediately comparable.

Value

If `as.data.frame=FALSE`, a list is returned containing `start` and `end`, two numeric matrices with one row per non-empty block in the grid and one column per column in `x`. `start` contains the mean location of all cells inside that block, and `end` contains the endpoint after adding the (scaled) average of the block's cell's velocity vectors.

If `as.data.frame=TRUE`, a data.frame is returned with numeric columns of the same contents as the list above. Column names are prefixed by `start.*` and `end.*`.

If `return.intermediates=TRUE`, a list is returned (irrespective of the value of `as.data.frame`) that in addition to `start` and `end` also contains intermediate objects `limits` (the ranges in `x` and `y`), `delta` (the grid intervals in `x` and `y`), `categories` (a DataFrame with integer row and column indices for each cell that specify the grid field that it is contained in), `grp` (numerical index of grid fields for each cell) and `vec` (velocity vectors for non-empty grid fields).

Author(s)

Aaron Lun

See Also

`embedVelocity`, to generate `embedded`.

Examples

```
tsne.results <- matrix(rnorm(10000), ncol=2)
tsne.vectors <- matrix(rnorm(10000), ncol=2)

out <- gridVectors(tsne.results, tsne.vectors)

# Demonstration for plotting.
plot(tsne.results[,1], tsne.results[,2], col='grey')
```

```
arrows(out$start.1, out$start.2, out$end.1, out$end.2, length=0.05)
```

---

plotVelocity *Phase and velocity graphs for a set of genes*

Description

For a each gene in a set of genes, show the phase graph (spliced versus unspliced counts and fitted model) and reduced dimension graphs with cell colored by velocity and (spliced) expression.

Usage

```
plotVelocity(
  x,
  genes,
  use.dimred = 1,
  assay.splicedM = "Ms",
  assay.unsplicedM = "Mu",
  which.plots = c("phase", "velocity", "expression"),
  genes.per.row = 1,
  color_by = "#222222",
  color.alpha = 0.4,
 colors.velocity = c("#A50026", "#D73027", "#F46D43", "#FDAE61", "#FEE08B", "#FFFFBF",
    "#D9EF8B", "#A6D96A", "#66BD63", "#1A9850", "#006837"),
 colors.expression = c("#440154", "#482576", "#414487", "#35608D", "#2A788E", "#21908C",
    "#22A884", "#43BF71", "#7AD151", "#BBDF27", "#FDE725"),
  max.abs.velo = 0.001
)
```

Arguments

x	A SingleCellExperiment object with RNA velocity results as returned by scvelo, and low-dimensional coordinates, e.g., after t-SNE, in its reducedDims.
genes	A character vector with one or several genes for which to plot phase and velocity graphs. genes have to be in rownames(x).
use.dimred	String or integer scalar specifying the reduced dimensions to retrieve from x.
assay.splicedM	An integer scalar or string specifying the assay of x containing the moments of spliced abundances.
assay.unsplicedM	An integer scalar or string specifying the assay of x containing the moments unspliced abundances.
which.plots	A character vector specifying which plots to create for each gene. Possible values are "phase", "velocity", "expression" and correspond to the phase graph or reduced dimension graphs with cells colored by velocity or (spliced) expression.

genes.per.row — An integer scalar with the numbers of genes to visualize per row of plots. For example, if which.plots = c("phase","expression") and genes.per.row = 2, the resulting figure will have four plot panels per row.

color_by — A character scalar specifying a column in colData(x) to color cells in the phase graph. Alternatively, color_by can be set to vector of valid R colors, either of length one (recycled for all cells) or of length ncol(x), which will then be used to color cells in the phase graph.

color.alpha — An integer scalar giving the transparency of colored cells. Possible values are between 0 (fully transparent) and 1.0 (opaque).

colors.velocity, colors.expression — Character vectors specifying the color ranges used for mapping velocities and expression values. The defaults are RColorBrewer::brewer.pal(11, "RdYlGn") for the velocities and viridisLite::viridis(11) for the expression values.

max.abs.velo — A numeric scalar greater than zero giving the maximum absolute velocity to limit the color scale for the "velocity" graph.

Details

Please note that plotVelocity will modify parameters of the current graphics device using layout and par, in order to create the layout for the generated graph panels.

Value

A patchwork object with the plots selected by which.plot for the genes in genes, arranged in a grid according to genes.per.row.

Author(s)

Michael Stadler

See Also

scvelo, to generate x, brewer.pal and viridis for creation of color palettes, packages **ggplot2** and **patchwork** used to generate and arrange the plots.

Examples

```
library(scuttle)
set.seed(42)
sce1 <- mockSCE(ncells = 100, ngenes = 500)
sce2 <- mockSCE(ncells = 100, ngenes = 500)

datlist <- list(X=counts(sce1), spliced=counts(sce1), unspliced=counts(sce2))

out1 <- scvelo(datlist, mode = "steady_state")
out2 <- scvelo(datlist, mode = "dynamical")

plotVelocity(out1, c("Gene_0031","Gene_0268"))
plotVelocity(out2, c("Gene_0031","Gene_0268"))
```

`plotVelocityStream` *Velocity stream plot in low-dimensional space*

Description

Plot velocities embedded into low-dimensional space as a stream plot. Stream lines are lines that follow the gradient in the velocity field and illustrate paths that cells could follow based on observed RNA velocities.

Usage

```
plotVelocityStream(
  sce,
  embedded,
  use.dimred = 1,
  color_by = "#444444",
  color.alpha = 0.2,
  grid.resolution = 60,
  scale = TRUE,
  stream.L = 10,
  stream.min.L = 0,
  stream.res = 4,
  stream.width = 8,
  color.streamlines = FALSE,
 color.streamlines.map = c("#440154", "#482576", "#414487", "#35608D", "#2A788E",
    "#21908C", "#22A884", "#43BF71", "#7AD151", "#BBDF27", "#FDE725"),
  arrow.angle = 8,
  arrow.length = 0.8
)
```

Arguments

`sce`	A SingleCellExperiment object containing low-dimensional coordinates, e.g., after t-SNE, in its `reducedDims`.
`embedded`	A low-dimensional projection of the velocity vectors into the embedding of `sce`. This should be of the same dimensions as `sce` and is typically produced by `embedVelocity`.
`use.dimred`	String or integer scalar specifying the reduced dimensions to retrieve from `sce`.
`color_by`	A character scalar specifying a column in `colData(sce)` to color cells in the phase graph. Alternatively, `color_by` can be set to a valid R color to be used to color cells.
`color.alpha`	An integer scalar giving the transparency of colored cells. Possible values are between 0 (fully transparent) and 1.0 (opaque).
`grid.resolution`	Integer scalar specifying the resolution of the grid, in terms of the number of grid intervals along each axis.

`scale` Logical scalar indicating whether the averaged vectors should be scaled by the grid resolution.

`stream.L` Integer scalar giving the typical length of a streamline low-dimensional space units.

`stream.min.L` A numeric scalar with the minimum length of segments to be shown.

`stream.res` Numeric scalar specifying the resolution of estimated streamlines (higher numbers increase smoothness of lines but also the time for computation).

`stream.width` A numeric scalar controlling the width of streamlines.

`color.streamlines`
Logical scalar. If `TRUE` streamlines will be colored by local velocity. Arrows cannot be shown in that case.

`color.streamlines.map`
A character vector specifying the color range used for mapping local velocities to streamline colors. The default is `viridisLite::viridis(11)`.

`arrow.angle`, `arrow.length`
Numeric scalars giving the `angle` and `length` of arrowheads.

Details

`grid.resolution` and `scale` are passed to `gridVectors`, which is used to summarized the velocity vectors into an initial grid. A full regular grid is computed from that and used in `geom_streamline` to calculate streamlines. The following arguments are passed to the arguments given in parenthesis of `geom_streamline`: `stream.L` (`L`), `stream.res` (`res`), `stream.min.L` (`min.L`), `arrow.angle` (`arrow.angle`) and `arrow.length` (`arrow.length`). Streamlines are computed by simple integration with a forward Euler method, and `stream.L` and `stream.res` are used to compute the number of steps and the time interval between steps for the integration. `stream.width` is multiplied with `..step..` estimated by `geom_streamline` to control the width of streamlines.

Value

A `ggplot2` object with the streamline plot.

Author(s)

Michael Stadler

See Also

`gridVectors` used to summarize velocity vectors into a grid (velocity field), the **ggplot2** package used for plotting, `geom_streamline` in package **metR** used to calculate and add streamlines from the RNA velocity field to the plot, `viridis` for creation of color palettes.

Examples

```
library(scuttle)
set.seed(42)
sce1 <- mockSCE(ncells = 100, ngenes = 500)
sce2 <- mockSCE(ncells = 100, ngenes = 500)
```

```
datlist <- list(X=counts(sce1), spliced=counts(sce1), unspliced=counts(sce2))

out <- scvelo(datlist, mode = "dynamical")

em <- embedVelocity(reducedDim(out, 1), out)[,1:2]

# https://github.com/kevinrue/velociraptor/issues/90

if (!basilisk.utils::isMacOSX()){
plotVelocityStream(out, em)
plotVelocityStream(out, em, color.streamlines = TRUE)
}
```

---

scvelo *RNA velocity with* **scVelo**

Description

Perform RNA velocity calculations with the **scVelo** package.

Usage

```
scvelo(x, ...)

## S4 method for signature 'ANY'
scvelo(
  x,
  subset.row = NULL,
  sf.X = NULL,
  sf.spliced = NULL,
  sf.unspliced = NULL,
  use.theirs = FALSE,
  mode = c("steady_state", "deterministic", "stochastic", "dynamical"),
  scvelo.params = list(),
  dimred = NULL,
  ncomponents = 30,
  BPPARAM = SerialParam(),
  BSPARAM = bsparam()
)

## S4 method for signature 'SummarizedExperiment'
scvelo(
  x,
  ...,
  assay.X = "counts",
  assay.spliced = "spliced",
```

```
  assay.unspliced = "unspliced"
)

## S4 method for signature 'SingleCellExperiment'
scvelo(x, ..., sf.X = sizeFactors(x), dimred = NULL, use.dimred = NULL)
```

Arguments

x	A named list of three matrices of the same dimensions where genes are in rows and cells are in columns. The list should contain "spliced" and "unspliced" entries containing spliced and unspliced counts, respectively. It should also contain an "X" entry containing the "usual" count matrix, see details below. Alternatively, a SummarizedExperiment object containing three such matrices among its assays.
...	For the generic, further arguments to pass to specific methods. For the SummarizedExperiment and SingleCellExperiment methods, further arguments to pass to the ANY method.
subset.row	A character, integer or logical vector specifying the genes to use for the velocity calculations. Defaults to all genes.
sf.X	A numeric vector containing size factors for usual count matrix. Defaults to librarySizeFactors on the "X" matrix in x.
sf.spliced	A numeric vector containing size factors for the spliced counts for each cell. Defaults to librarySizeFactors on the "spliced" matrix in x.
sf.unspliced	A numeric vector containing size factors for the unspliced counts for each cell. Defaults to librarySizeFactors on the "unspliced" matrix in x.
use.theirs	Logical scalar indicating whether **scVelo**'s gene filtering and normalization should be used.
mode	String specifying the method to use to estimate the transcriptional dynamics.
scvelo.params	List of lists containing arguments for individual **scVelo** functions, see details below.
dimred	A low-dimensional representation of the cells with number of rows equal to the number of cells in x, used to find the nearest neighbors.
ncomponents	Numeric scalar indicating the number of principal components to obtain. Only used if use.theirs=FALSE and dimred=NULL.
BPPARAM	A BiocParallelParam object specifying whether the PCA calculations should be parallelized. Only used if use.theirs=FALSE and dimred=NULL.
BSPARAM	A BiocSingularParam object specifying which algorithm should be used to perform the PCA. Only used if use.theirs=FALSE and dimred=NULL.
assay.X	An integer scalar or string specifying the assay of x containing the usual count matrix.
assay.spliced	An integer scalar or string specifying the assay of x containing the spliced counts.
assay.unspliced	An integer scalar or string specifying the assay of x containing the unspliced counts.

use.dimred String naming the entry of reducedDims(x) to use for nearest neighbor calculations. Ignored if dimred is supplied.

Details

This function uses the **scVelo** Python package (https://pypi.org/project/scvelo/) for RNA velocity calculations. The main difference from the original **velocyto** approach is that the dynamical model of **scVelo** does not rely on the presence of observed steady-state populations, which should improve the reliability of the velocity calculations in general applications.

For consistency with other Bioconductor workflows, we perform as many standard steps in R as we can before starting the velocity calculations with **scVelo**. This involves:

1. Size factor-based normalization with sf.* values and normalizeCounts. For "X", log-transformation is performed as well, while for the others, only scaling normalization is performed.
2. Subsetting all matrices to subset.row, most typically to a subset of interest, e.g., highly variable genes. Note that, if set, any subsetting is done *after* normalization so that library sizes are correctly computed.
3. If dimred=NULL, the PCA step on the log-expression values derived from the "X" matrix, using the specified BSPARAM to obtain the first ncomponents PCs.

This allows us to guarantee that, for example, the log-expression matrix of HVGs or the PCA coordinates are the same as that used in other applications like clustering or trajectory reconstruction.

Nonetheless, one can set use.theirs=TRUE to directly use the entire **scVelo** normalization and filtering pipeline. This ignores all of the size factors arguments (sf.*), all of the PCA-related arguments (ncomponents, BSPARAM) and subset.row. However, if a low-dimensionality result is supplied via dimred or use.dimred, the **scVelo** PCA will always be omitted.

Upon first use, this function will instantiate a conda environment containing the **scVelo** package. This is done via the **basilisk** package - see the documentation for that package for trouble-shooting.

Value

A SingleCellExperiment is returned containing the output of the velocity calculations. Of particular interest are:

- the velocity_pseudotime field in the colData, containing the velocity pseudotime for each cell.
- the velocity entry of the assays, containing the velocity vectors for each cell.

The output will always have number of columns equal to the number of cells supplied in x, though the number of rows will depend on whether any subsetting (if subset.row is supplied) or feature selection (if use.theirs=TRUE) was performed.

Comments on the three matrices

Strictly speaking, only the spliced and unspliced matrices are necessary for the velocity calculations. However, it is often the case that the spliced matrix is not actually the same as a "usual" count matrix (e.g., generated by summing counts across all exons for all mapped genes). This is due to differences in the handling of ambiguous reads that map across exon-intron boundaries, or to genomic regions

that can be either exonic or intronic depending on the isoform; the spliced count matrix is more likely to exclude such reads.

We request the usual count matrix as the `"X"` entry of `x` to ensure that the PCA and nearest neighbor detection in **scVelo** are done on the same data as that used in other steps of the large analysis (e.g., clustering, visualization, trajectory reconstruction). In practice, if the usual count matrix is not available, one can often achieve satisfactory results by simply re-using the spliced count matrix as both the `"X"` and `"spliced"` entries of `x`.

Note that if reduced dimensions are supplied in `dimred`, any `"X"` entry is only used to create the AnnData object and is not used in any actual calculations.

Additional arguments to Python

Additional arguments to **scVelo** functions are provided via `scvelo.params`. This is a named list where each entry is named after a function and is itself a named list of arguments for that function. The following function names are currently recognized:

- `"filter_and_normalize"`, for gene selection and normalization. This is not used unless `use.theirs=TRUE`.
- `"moments"`, for PCA and nearest neighbor detection. The PCA is not performed if `dimred` or `use.dimred` is already supplied.
- `"recover_dynamics"`
- `"velocity"`
- `"velocity_graph"`
- `"velocity_pseudotime"`
- `"latent_time"`
- `"velocity_confidence"`

See the **scVelo** documentation for more details about the available arguments and the examples below for a syntax example.

Supported operating systems and architectures

scVelo dependencies are pinned in a Conda environment to ensure reproducibility.

Differences in packages and versions available from Conda require different environments for different operating systems and architectures. **basilisk.utils** is used to determine the operating system and architecture of the computer used to run `scvelo()`, using to the appropriate Conda environment.

As of the latest **velociraptor** update (24 May 2024):

All environments **tqdm** and **ipywidgets** are installed to suppress the message "Unable to create progress bar".

Linux **scVelo** v0.3.2 from conda-forge is used. This is the latest version available to date. **libtiff** is pinned to v4.5.1 and **pillow** is pinned to v10.0.0 (https://github.com/conda-forge/libtiff-feedstock/issues/104#issuecomment-2375893029), **scipy** is pinned to v1.13.1 (https://github.com/theislab/scvelo/issues/1260).

Linux AArch64 **scVelo** v0.3.2 from conda-forge is used. This is the latest version available to date. **libtiff** is pinned to v4.5.1 and **pillow** is pinned to v10.0.0 (https://github.com/conda-forge/libtiff-feedstock/issues/104#issuecomment-2375893029), **scipy** is pinned to v1.13.1 (https://github.com/theislab/scvelo/issues/1260). Please note that this environment has not been validated yet; it is derived from the environment from Linux (above) and requires additional testing to identify a working environment before pinning all the packages and versions in the environment.

macOS **scVelo** v0.3.2 from conda-forge is used. This is the latest version available to date. **scipy** is pinned to v1.13.1 (https://github.com/theislab/scvelo/issues/1260).

macOS ARM **scVelo** v0.3.2 from conda-forge is used. This is the latest version available to date. **scipy** is pinned to v1.13.1 (https://github.com/theislab/scvelo/issues/1260).

Windows **scVelo** v0.2.5 from bioconda is used. Later versions of **scVelo** depend on **jaxlib** which is not supported on Windows (https://github.com/google/jax/issues/438). **matplotlib** is pinned to v3.6.3 (https://github.com/scverse/scanpy/issues/2411), **pandas** is pinned to v1.5.2 (https://stackoverflow.com/questions/76234312/importerror-cannot-import-name-is-categor and **numpy** is pinned to v1.21.1 (https://github.com/theislab/scvelo/issues/1109).

Author(s)

Aaron Lun, Charlotte Soneson, Kevin Rue-Albrecht

References

Bergen, V., Lange, M., Peidli, S. et al. Generalizing RNA velocity to transient cell states through dynamical modeling. Nat Biotechnol 38, 1408–1414 (2020). https://doi.org/10.1038/s41587-020-0591-3

Examples

```
# Using mock data to demonstrate the process:
library(scuttle)
sce1 <- mockSCE()
sce2 <- mockSCE()

spliced <- counts(sce1)
unspliced <- counts(sce2)

out <- scvelo(list(X=spliced, spliced=spliced, unspliced=unspliced))

# make scvelo use 10 rather than the default 30 neighbors to compute moments for velocity estimation:
out <- scvelo(list(X=spliced, spliced=spliced, unspliced=unspliced),
              scvelo.params=list(neighbors=list(n_neighbors=10L)))
```

Index