Package: SeuratWrappers (via r-universe)

May 26, 2026

Title Community-Provided Methods and Extensions for the Seurat Object

Version 0.4.0

Date 2024-11-20

Description SeuratWrappers is a collection of community-provided methods and extensions for Seurat, curated by the Satija Lab at NYGC. These methods comprise functionality not presently found in Seurat, and are able to be updated much more frequently.

License GPL-3 | file LICENSE

Remotes welch-lab/liger, hms-dbmi/conos, immunogenomics/harmony, immunogenomics/presto, satijalab/seurat-data, velocyto-team/velocyto.R, SaskiaFreytag/schex@031320d, cole-trapnell-lab/monocle3, mojaveazure/seurat-disk, powellgenomicslab/Nebulosa, atakanekiz/CIPR-Package, prabhakarlab/Banksy

Depends R (>= 3.5.0)

Imports BiocManager, cowplot, ggplot2, igraph, Matrix, methods, remotes, rsvd, Seurat (>= 5.0.0), stats, utils, rlang

Collate 'internal.R' 'alevin.R' 'alra.R' 'banksy.R' 'cellbrowser.R' 'cogaps.R' 'conos.R' 'fast_mnn.R' 'fast_mnn_v5.R' 'glmpca.R' 'liger.R' 'miqc.R' 'monocle3.R' 'pacmap.R' 'presto.R' 'scVI.R' 'tricycle.R' 'velocity.R'

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

Suggests cipr, conos, rliger (>= 0.5.0), harmony, batchelor, SeuratData, SeuratDisk, velocyto.R, schex, tximport, fishpond, monocle3, CoGAPS, glmpca, Nebulosa, presto, flexmix, tricycle, Banksy

Config/pak/sysreqs cmake git libglpk-dev make libicu-dev libpng-dev libuv1-dev libxml2-dev libssl-dev python3 zlib1g-dev

Repository https://blaserlab.r-universe.dev

Date/Publication 2026-02-17 22:55:55 UTC

RemoteUrl https://github.com/satijalab/seurat-wrappers

RemoteRef HEAD

RemoteSha ffaf74e306279b1ec16e31c9cb2142ebb2bc4bc1

Contents

SeuratWrappers-package . . . 2
ALRAChooseKPlot . . . 3
as.cell_data_set . . . 4
as.Seurat . . . 5
CellBrowser . . . 7
FastMNNIntegration . . . 9
findMatrix . . . 11
LearnGraph . . . 11
PlotMiQC . . . 12
ReadAlevin . . . 13
ReadVelocity . . . 14
RunALRA . . . 14
RunBanksy . . . 16
RunCoGAPS . . . 18
RunFastMNN . . . 19
RunGLMPCA . . . 20
RunMiQC . . . 21
RunOptimizeALS . . . 23
RunPaCMAP . . . 24
RunPresto . . . 27
RunPrestoAll . . . 28
RunQuantileAlignSNF . . . 30
RunQuantileNorm . . . 31
RunSNF . . . 33
Runtricycle . . . 34
RunVelocity . . . 35
scVIIntegration . . . 36
StopCellbrowser . . . 37
writeSparseTsvChunks . . . 38

Index **39**

`SeuratWrappers-package`

SeuratWrappers: Community-Provided Methods and Extensions for the Seurat Object

---

Description

SeuratWrappers is a collection of community-provided methods and extensions for Seurat, curated by the Satija Lab at NYGC. These methods comprise functionality not presently found in Seurat, and are able to be updated much more frequently.

Author(s)

Maintainer: Paul Hoffman <nygcSatijalab@nygenome.org> (ORCID)

Authors:

- Andrew Butler <abutler@nygenome.org> (ORCID)
- Rahul Satija <rsatija@nygenome.org> (ORCID)
- Tim Stuart <tstuart@nygenome.org> (ORCID)

Other contributors:

- Saket Choudhary <schoudhary@nygenome.org> (ORCID) [contributor]
- David Collins <dcollins@nygenome.org> (ORCID) [contributor]
- Yuhan Hao <yhao@nygenome.org> (ORCID) [contributor]
- Austin Hartman <ahartman@nygenome.org> (ORCID) [contributor]
- Gesmira Molla <gmolla@nygenome.org> (ORCID) [contributor]

`ALRAChooseKPlot` *ALRA Approximate Rank Selection Plot*

Description

Plots the results of the approximate rank selection process for ALRA.

Usage

```
ALRAChooseKPlot(object, start = 0, combine = TRUE)
```

Arguments

`object`	Seurat object
`start`	Index to start plotting singular value spacings from. The transition from "signal" to "noise" in the is hard to see because the first singular value spacings are so large. Nicer visualizations result from skipping the first few. If set to 0 (default) starts from k/2.
`combine`	Combine plots into a single gg object; note that if TRUE, themeing will not work when plotting multiple features

Value

A list of 3 ggplot objects splotting the singular values, the spacings of the singular values, and the p-values of the singular values.

Author(s)

Jun Zhao, George Linderman

See Also

RunALRA

`as.cell_data_set` *Convert objects to Monocle3* `cell_data_set` *objects*

Description

Convert objects to Monocle3 `cell_data_set` objects

Usage

```
as.cell_data_set(x, ...)

## S3 method for class 'Seurat'
as.cell_data_set(
  x,
  assay = DefaultAssay(object = x),
  reductions = AssociatedDimReducs(object = x, assay = assay),
  default.reduction = DefaultDimReduc(object = x, assay = assay),
  graph = paste0(assay, "_snn"),
  group.by = NULL,
  ...
)
```

Arguments

`x`	An object
`...`	Arguments passed to other methods
`reductions`	A vector of dimensional reductions add to the `cell_data_set` object; defaults to all dimensional reductions calculated from `assay` and all global dimensional reductions
`default.reduction`	Name of dimensional reduction to use for clustering name
`graph`	Name of graph to be used for clustering results
`group.by`	Name of cell-level metadata column to use as identites; pass

Details

The Seurat method utilizes as.SingleCellExperiment to transfer over expression and cell-level metadata. The following additional information is also transferred over:

- Cell emebeddings are transferred over to the reducedDims slot. Dimensional reduction names are converted to upper-case (eg. "umap" to "UMAP") to match Monocle 3 style
- Feature loadings are transfered to cds@reduce_dim_aux$gene_loadings if present. **NOTE**: only the feature loadings of the last dimensional reduction are transferred over
- Standard deviations are added to cds@reduce_dim_aux$prop_var_expl if present. **NOTE**: only the standard deviations of the last dimensional reduction are transferred over
- Clustering information is transferred over in the following manner: if cell-level metadata entries "monocle3_clusters" and "monocle3_partitions" exist, then these will be set as the clusters and partitions, with no nearest neighbor graph being added to the object; otherwise, Seurat's nearest-neighbor graph will be converted to an igraph object and added to the cell_data_set object along with Seurat's clusters. No partition information is added when using Seurat's clsuters

Value

A cell_data_set object

See Also

as.SingleCellExperiment

---

as.Seurat *Extra conversions to Seurat objects*

Description

Extra conversions to Seurat objects

Usage

```
## S3 method for class 'Conos'
as.Seurat(
  x,
  method = "mnn",
  reduction = "largeVis",
  idents = names(x = x$clusters)[1],
  verbose = TRUE,
  ...
)

## S3 method for class 'cell_data_set'
as.Seurat(
```

```
  x,
  counts = "counts",
  data = NULL,
  assay = "RNA",
  project = "cell_data_set",
  loadings = NULL,
  clusters = NULL,
  ...
)

## S3 method for class 'list'
as.Seurat(
  x,
  default.assay = 1,
  slot = "counts",
  min.cells = 0,
  min.features = 0,
  verbose = TRUE,
  ...
)
```

Arguments

method	Name of matching method graph was built using
reduction	Name of graph embedding, if calculated
idents	Name of clutering method to set as identity class
loadings	Name of dimensional reduction to save loadings to, if present; defaults to first dimensional reduction present (eg. `SingleCellExperiment::reducedDimNames(x)[1]`); pass `NA` to suppress transfer of loadings
clusters	Name of clustering method to use for setting identity classes
default.assay	Name or index of matrix to use as default assay; defaults to name of first matrix in list
slot	Name of slot to store matrix in; choose from 'counts' or 'data'

Details

The `Conos` method for `as.Seurat` only works if all samples are `Seurat` objects. The object is initially constructed by merging all samples together using `merge`, any sample-level dimensional reductions and graphs will be lost during the merge. Extra information is added to the resulting Seurat object as follows:

- Pairwise alignments will be stored in miscellaneous data, as will any other miscellaneous information
- If a graph is present in the `graph` field, it will be stored as a `Graph` object, reordered to match cell order in the new `Seurat` object. It will be named "`DefaultAssay(SeuratObject)_method`"
- If an embedding is present in the `embedding` field as a `matrix`, it will be stored as a `DimReduc` object with the name `reduction` and a key value of "`toupper(reduction)_`"

- If the length of the clusters field is greater than zero, clustering information (groups field) will be added to object metadata. Extra information (result field) will be added to miscellaneous data with the name "conos.clustering.result"
- If present, the first clustering entry in the clusters field will be set as object identity classes

The cell_data_set method for as.Seurat utilizes the SingleCellExperiment method of as.Seurat to handle moving over expression data, cell embeddings, and cell-level metadata. The following additional information will also be transfered over:

- Feature loadings from cds@reduce_dim_aux$gene_loadings will be added to the dimensional reduction specified by loadings or the name of the first dimensional reduction that contains "pca" (case-insensitive) if loadings is not set
- Monocle 3 clustering will be set as the default identity class. In addition, the Monocle 3 clustering will be added to cell-level metadata as “monocle3_clusters”, if present
- Monocle 3 partitions will be added to cell-level metadata as “monocle3_partitions”, if present
- Monocle 3 pseudotime calculations will be added to “monocle3_pseudotime”, if present
- The nearest-neighbor graph, if present, will be converted to a Graph object, and stored as “assay_monocle3_graph”

The list method for as.Seurat takes a named list of matrices (dense or sparse) and creates a single Seurat object where each matrix is its own assay. The names of the list are taken to be the names of the assays. If not present, assays will be named as "Assay#" where "#" is the index number in the list of matrices. Objects will be constructed as follows:

- By default, all matrices are assumed to be raw counts and will be stored in the counts slot. This can be changed to store in the matrix in the data slot instead. The slot parameter is vectorized, so different matrices can be stored in either counts or data
- For any and all matrices designated as counts, the min.cells and min.features filtering will be applied. These parameters are vectorized, so different filterings can be applied to different matrices
- No extra information (eg. project) can be provided to CreateSeuratObject

See Also

as.Seurat

as.Seurat.SingleCellExperiment

CellBrowser *Export* Seurat *objects for UCSC cell browser and stop open cell browser instances from R*

Description

Export Seurat objects for UCSC cell browser and stop open cell browser instances from R

Usage

```
ExportToCellbrowser(
  object,
  dir,
  dataset.name = Project(object = object),
  reductions = NULL,
  markers.file = NULL,
  cluster.field = NULL,
  cb.dir = NULL,
  port = NULL,
  use.mtx = FALSE,
  meta.fields = NULL,
  meta.fields.names = NULL,
  matrix.slot = "counts",
  markers.n = 100,
  skip.markers = FALSE,
  skip.expr.matrix = FALSE,
  skip.metadata = FALSE,
  skip.reductions = FALSE
)
```

Arguments

`object`	Seurat object
`dir`	path to directory where to save exported files. These are: exprMatrix.tsv, tsne.coords.tsv, meta.tsv, markers.tsv and a default cellbrowser.conf
`dataset.name`	name of the dataset. Defaults to Seurat project name
`reductions`	vector of reduction names to export, defaults to all reductions.
`markers.file`	path to file with marker genes. By defaults, marker are searched in the object itself as misc$markers. If none are supplied in object or via this argument, they are recalculated with `FindAllMarkers`
`cluster.field`	name of the metadata field containing cell cluster
`cb.dir`	path to directory where to create UCSC cellbrowser static website content root, e.g. an index.html, .json files, etc. These files can be copied to any webserver. If this is specified, the cellbrowser package has to be accessible from R via reticulate.
`port`	on which port to run UCSC cellbrowser webserver after export
`use.mtx`	export the matrix in .mtx.gz format. Default is False, unless the matrix is bigger than R's maximum matrix size.
`meta.fields`	vector of meta fields to export, default is all.
`meta.fields.names`	vector meta field names to show in UI. Must have same length as meta.fields. Default is meta.fields.
`matrix.slot`	matrix to use, default is 'counts'

markers.n	if no markers were supplied, FindAllMarkers is run. This parameter indicates how many markers to calculate, default is 100
skip.markers	whether to skip exporting markers
skip.expr.matrix	whether to skip exporting expression matrix
skip.metadata	whether to skip exporting metadata
skip.reductions	whether to skip exporting reductions
...	specifies the metadata fields to export. To supply a field and its human readable name, pass name as field="name" parameter.

Value

This function exports Seurat object as a set of tsv files to dir directory, copying the markers.file if it is passed. It also creates the default cellbrowser.conf in the directory. This directory could be read by cbBuild to create a static website viewer for the dataset. If cb.dir parameter is passed, the function runs cbBuild (if it is installed) to create this static website in cb.dir directory. If port parameter is passed, it also runs the webserver for that directory and opens a browser.

Author(s)

Maximilian Haeussler, Nikolay Markov

Examples

```
## Not run:
ExportToCellbrowser(pbmc_small, dataset.name = "PBMC", dir = "out")

## End(Not run)
```

FastMNNIntegration *Run fastMNN in Seurat 5*

Description

Run fastMNN in Seurat 5

Usage

```
FastMNNIntegration(
  object,
  assay = NULL,
  orig = NULL,
  groups = NULL,
  layers = NULL,
  scale.layer = NULL,
```

```
  features = 2000,
  new.reduction = "integrated.mnn",
  reduction.key = "mnn_",
  reconstructed.assay = "mnn.reconstructed",
  verbose = TRUE,
  ...
)
```

Arguments

object	A merged seurat object
assay	Assay to use, defaults to the default assay of the first object
groups	A one-column data frame with grouping information
layers	Layers to use
features	Either a list of features to use when calculating batch correction, or a number (2000 by default) of variable features to select.
reduction.key	Key for resulting DimReduc
reconstructed.assay	Name for the assay containing the low-rank reconstruction of the expression matrix.
verbose	Print messages
...	Extra parameters passed to fastMNN
reduction.name	Name to store resulting DimReduc object as

Value

A Seurat object merged from the objects in object.list and a new DimReduc of name reduction.name (key set to reduction.key) with corrected embeddings matrix as well as the rotation matrix used for the PCA stored in the feature loadings slot. Also returns an expression matrix reconstructed from the low-rank approximation in the reconstructed.assay assay; all other metadata info fastMNN is stored in the tool slot, accessible with Tool

Note

This function requires the **batchelor** package to be installed

See Also

fastMNN Tool

Examples

```
## Not run:
# Preprocessing
obj <- SeuratData::LoadData("pbmcsca")
obj[["RNA"]] <- split(obj[["RNA"]], f = obj$Method)
obj <- NormalizeData(obj)
```

```
obj <- FindVariableFeatures(obj)
obj <- ScaleData(obj)
obj <- RunPCA(obj)

# After preprocessing, we integrate layers:
obj <- IntegrateLayers(object = obj, method = FastMNNIntegration,
  new.reduction = 'integrated.mnn', verbose = FALSE)

# We can also add parameters specific to FastMNN.
# Here we set `k` to specify the number of nearest neighbors to use when identifying MNNs:
obj <- IntegrateLayers(object = obj, method = FastMNNIntegration,
  new.reduction = 'integrated.mnn', k = 15, verbose = FALSE)

## End(Not run)
```

`findMatrix` *used by ExportToCellbrowser: Return a matrix object from a Seurat object or show an error message*

Description

used by ExportToCellbrowser: Return a matrix object from a Seurat object or show an error message

Usage

```
findMatrix(object, matrix.slot)
```

Arguments

`object`	Seurat object
`matrix.slot`	the name of the slot

`LearnGraph` *Run* `link[monocle3]{learn_graph}` *on a* `Seurat` *object*

Description

Run `link[monocle3]{learn_graph}` on a `Seurat` object

Usage

```
LearnGraph(object, reduction = DefaultDimReduc(object = object), ...)
```

Arguments

object	A Seurat object
reduction	Name of reduction to use for learning the pseudotime graph
...	Arguments passed to learn_graph

Value

A cell_data_set object with the pseudotime graph

See Also

learn_graph cell_data_set

PlotMiQC *Run miQC on a Seurat object*

Description

Run miQC on a Seurat object

Usage

```
PlotMiQC(
  seurat_object,
  percent.mt = "percent.mt",
  nFeature_RNA = "nFeature_RNA",
  model.slot = "flexmix_model",
  color.by = "miQC.probability"
)
```

Arguments

object	Seurat object

Details

Function to plot the miQC mixture model stored in a Seurat object. 'RunMiQC' must be run prior to plotting.

References

Hippen et al. (2021) miQC: An adaptive probabilistic framework for quality control of single-cell RNA-sequencing data. bioRxiv doi: 10.1101/2021.03.03.433798

ReadAlevin *Load alevin quantification data*

Description

A wrapper around tximport to create a SeuratObject from alevin quantification data.

Usage

```
ReadAlevin(file, getMeta = FALSE, meanAndVariance = FALSE, ...)
```

Arguments

file	path to quants_mat.gz file within alevin directory
getMeta	logical, option to use tximeta to programmatically obtain gene range information, default is FALSE. Ranges are stored in chr, start, and end in the meta.features slot.
meanAndVariance	logical, should mean and variance of counts be returned in counts and data slots, respectively
...	extra arguments passed to tximport, for example, alevinArgs=list(filterBarcodes=TRUE).

Value

returns a Seurat object with alevin counts

Author(s)

Avi Srivastava

References

Srivastava, Avi, et al. "Alevin efficiently estimates accurate gene abundances from dscRNA-seq data." Genome biology 20.1 (2019): 65.

See Also

alevin

ReadVelocity *Load RNA Velocity data from a loom file*

Description

This is a wrapper around `read.loom.matrices`, but sends messages to `stderr` instead of `stdout` (or silences messages with `verbose = FALSE`)

Usage

```
ReadVelocity(file, engine = "hdf5r", verbose = TRUE)
```

Arguments

`file`	Path to loom file
`engine`	Method to load data data, choose from 'hdf5r' or 'h5'
`verbose`	Display progress updates

See Also

`read.loom.matrices`

RunALRA *Run Adaptively-thresholded Low Rank Approximation (ALRA)*

Description

Runs ALRA, a method for imputation of dropped out values in scRNA-seq data. Computes the k-rank approximation to A_norm and adjusts it according to the error distribution learned from the negative values. Described in Linderman, G. C., Zhao, J., Kluger, Y. (2018). "Zero-preserving imputation of scRNA-seq data using low rank approximation." (bioRxiv:138677)

Usage

```
RunALRA(object, ...)

## Default S3 method:
RunALRA(
  object,
  k = NULL,
  q = 10,
  quantile.prob = 0.001,
  use.mkl = FALSE,
  mkl.seed = -1,
  ...
```

```
)

## S3 method for class 'Seurat'
RunALRA(
  object,
  k = NULL,
  q = 10,
  quantile.prob = 0.001,
  use.mkl = FALSE,
  mkl.seed = -1,
  assay = NULL,
  slot = "data",
  setDefaultAssay = TRUE,
  genes.use = NULL,
  K = NULL,
  thresh = 6,
  noise.start = NULL,
  q.k = 2,
  k.only = FALSE,
  ...
)
```

Arguments

object	An object
...	Arguments passed to other methods
k	The rank of the rank-k approximation. Set to NULL for automated choice of k.
q	The number of additional power iterations in randomized SVD when computing rank k approximation. By default, q=10.
quantile.prob	The quantile probability to use when calculating threshold. By default, quantile.prob = 0.001.
use.mkl	Use the Intel MKL based implementation of SVD. Needs to be installed from https://github.com/KlugerLab/rpca-mkl.
mkl.seed	Only relevant if use.mkl=T. Set the seed for the random generator for the Intel MKL implementation of SVD. Any number <0 will use the current timestamp. If use.mkl=F, set the seed using set.seed() function as usual.
assay	Assay to use
slot	slot to use
setDefaultAssay	If TRUE, will set imputed results as default Assay
genes.use	genes to impute
K	Number of singular values to compute when choosing k. Must be less than the smallest dimension of the matrix. Default 100 or smallest dimension.
noise.start	Index for which all smaller singular values are considered noise. Default K - 20.
q.k	Number of additional power iterations when choosing k. Default 2.

k.only	If TRUE, only computes optimal k WITHOUT performing ALRA
p.val.th	The threshold for "significance" when choosing k. Default 1e-10.

Author(s)

Jun Zhao, George Linderman

References

Linderman, G. C., Zhao, J., Kluger, Y. (2018). "Zero-preserving imputation of scRNA-seq data using low rank approximation." (bioRxiv:138677)

See Also

ALRAChooseKPlot

Examples

```
## Not run:
pbmc_small
# Example 1: Simple usage, with automatic choice of k.
pbmc_small_alra <- RunALRA(object = pbmc_small)
# Example 2: Visualize choice of k, then run ALRA
# First, choose K
pbmc_small_alra <- RunALRA(pbmc_small, k.only=TRUE)
# Plot the spectrum, spacings, and p-values which are used to choose k
ggouts <- ALRAChooseKPlot(pbmc_small_alra)
do.call(gridExtra::grid.arrange, c(ggouts, nrow=1))
# Run ALRA with the chosen k
pbmc_small_alra <- RunALRA(pbmc_small_alra)

## End(Not run)
```

RunBanksy *Run Banksy on a Seurat Object*

Description

Run Banksy on a Seurat Object

Usage

```
RunBanksy(
  object,
  lambda,
  assay = "RNA",
  slot = "data",
  use_agf = FALSE,
```

```
  dimx = NULL,
  dimy = NULL,
  dimz = NULL,
  ndim = 2,
  features = "variable",
  group = NULL,
  split.scale = TRUE,
  k_geom = 15,
  n = 2,
  sigma = 1.5,
  alpha = 0.05,
  k_spatial = 10,
  spatial_mode = "kNN_median",
  assay_name = "BANKSY",
  M = NULL,
  verbose = TRUE
)
```

Arguments

object	A Seurat object
lambda	(numeric) Spatial weight parameter
assay	(character) Assay in Seurat object to use
slot	(character) Slot in Seurat assay to use
use_agf	(boolean) Whether to use the AGF
dimx	(character) Column name of spatial x dimension (must be in metadata)
dimy	(character) Column name of spatial y dimension (must be in metadata)
dimz	(character) Column name of spatial z dimension (must be in metadata)
ndim	(integer) Number of spatial dimensions to extract
features	(character) Features to compute. Can be 'all', 'variable' or a vector of feature names
group	(character) Column name of a grouping variable (must be in metadata)
split.scale	(boolean) Whether to separate scaling by group
k_geom	(numeric) kNN parameter - number of neighbors to use
n	(numeric) kNN_rn parameter - exponent of radius
sigma	(numeric) rNN parameter - standard deviation of Gaussian kernel
alpha	(numeric) rNN parameter - determines radius used
k_spatial	(numeric) rNN parameter - number of neighbors to use
spatial_mode	(character) Kernel for neighborhood computation • kNN_median: k-nearest neighbors with median-scaled Gaussian kernel • kNN_r: k-nearest neighbors with $1/r$ kernel • kNN_rn: k-nearest neighbors with $1/r^n$ kernel • kNN_rank: k-nearest neighbors with rank Gaussian kernel

- kNN_unif: k-nearest neighbors wth uniform kernel
- rNN_gauss: radial nearest neighbors with Gaussian kernel

assay_name	(character) Name for Banksy assay in Seurat object
M	(numeric) Advanced usage. Highest azimuthal harmonic
verbose	(boolean) Print messages

Value

A Seurat object with new assay holding a Banksy matrix

Author(s)

Joseph Lee, Vipul Singhal

References

Vipul Singhal, Nigel Chou et. al. BANKSY: A Spatial Omics Algorithm that Unifies Cell Type Clustering and Tissue Domain Segmentation

See Also

ComputeBanksy

RunCoGAPS *Run CoGAPs on a Seurat object*

Description

Run CoGAPs on a Seurat object

Usage

```
RunCoGAPS(
  object,
  assay = NULL,
  slot = "counts",
  params = NULL,
  temp.file = NULL,
  reduction.name = "CoGAPS",
  reduction.key = "CoGAPS_",
  ...
)
```

Arguments

object	Seurat object
assay	Assay to pull data from
slot	Slot to pull data from.
params	CogapsParams object for specifying parameter settings
temp.file	Name of temporary data matrix file to create if running in a distributed mode. Setting to TRUE will generate the file name using tempfile.
reduction.name	Name of the CoGAPS reduction returned
reduction.key	Key for the CoGAPS reduction returned

Value

Returns a Seurat object with the CoGAPS results stored as a DimReduc object

References

E.J. Fertig, J. Ding, A.V. Favorov, G. Parmigiani, and M.F. Ochs (2010) CoGAPS: an integrated R/C++ package to identify overlapping patterns of activation of biological processes from expression data. Bioinformatics 26:2792-2793.

See Also

CoGAPS

RunFastMNN *Run fastMNN*

Description

Run fastMNN

Usage

```
RunFastMNN(
  object.list,
  assay = NULL,
  features = 2000,
  reduction.name = "mnn",
  reduction.key = "mnn_",
  reconstructed.assay = "mnn.reconstructed",
  verbose = TRUE,
  ...
)
```

Arguments

object.list	A list of Seurat objects
assay	Assay to use, defaults to the default assay of the first object
features	Either a list of features to use when calculating batch correction, or a number (2000 by default) of variable features to select.
reduction.name	Name to store resulting DimReduc object as
reduction.key	Key for resulting DimReduc
reconstructed.assay	Name for the assay containing the low-rank reconstruction of the expression matrix.
verbose	Print messages from SelectIntegrationFeatures
...	Extra parameters passed to fastMNN

Value

A Seurat object merged from the objects in object.list and a new DimReduc of name reduction.name (key set to reduction.key) with corrected embeddings matrix as well as the rotation matrix used for the PCA stored in the feature loadings slot. Also returns an expression matrix reconstructed from the low-rank approximation in the reconstructed.assay assay; all other metadata info fastMNN is stored in the tool slot, accessible with Tool

See Also

fastMNN Tool

RunGLMPCA *Run GLMPCA*

Description

Run GLMPCA

Usage

```
RunGLMPCA(
  object,
  L = 5,
  assay = NULL,
  features = NULL,
  reduction.name = "glmpca",
  reduction.key = "GLMPC_",
  verbose = TRUE,
  ...
)
```

Arguments

object	A Seurat object
L	The number of dimensions to return (defaults to 5)
assay	Assay to use, defaults to the default assay
features	A list of features to use when performing GLM-PCA. If null, defaults to variable features.
reduction.name	Name to store resulting DimReduc object as. Defaults to glmpca
reduction.key	Key for resulting DimReduc. Defaults to GLMPC_
...	Extra parameters passed to glmpca

Value

A Seurat object containing the output of GLMPCA stored as a DimReduc object.

Author(s)

Will Townes

References

Townes, W., Hicks, SC, Aryee, MJ, Irizarry, RA. (2019). "Feature selection and dimension reduction for single-cell RNA-Seq based on a multinomial model." Genome Biology.

Examples

```
## Not run:
pbmc_small
pbmc_small <- RunGLMPCA(pbmc_small)
DimPlot(pbmc_small, reduction = 'glmpca')

## End(Not run)
```

RunMiQC *Run miQC on a Seurat object*

Description

Run miQC on a Seurat object

Usage

```
RunMiQC(
  object,
  percent.mt = "percent.mt",
  nFeature_RNA = "nFeature_RNA",
  posterior.cutoff = 0.75,
  model.type = "linear",
  model.slot = "flexmix_model",
  verbose = TRUE,
  backup.option = "percentile",
  backup.percentile = 0.99,
  backup.percent = 5,
  ...
)
```

Arguments

object	Seurat object
percent.mt	(character) Name of the column in the Seurat metadata that contains the percent of reads attributed to mitochondrial genes. Defaults to "percent.mt".
nFeature_RNA	(character) Name of the column in the Seurat metadata that contains the number of reads per cell. Defaults to "nFeature_RNA".
posterior.cutoff	numeric) The posterior probability of a cell being part of the compromised distribution, a number between 0 and 1. Any cells below the appointed cutoff will be marked to keep. Defaults to 0.75.
model.type	(character) What type of model to generate. A linear mixture model ("linear") is recommended, but currently b-spline ("spline") and two-degree polynomial ("polynomial") are also supported Default = "linear".
verbose	Boolean. TRUE to show progress messages, FALSE to hide progress messages
backup.option	(character) In case flexmix fails to build a 2 cluster mixture model, what should RunMiQC do: "percent" (set miQC.keep values according to backup.percent), "percentile" (set miQC.keep values according to backup.percentile), "pass" (return original Seurat object), or "halt" (stop RunMiQC). "percent", "percentile", and "pass" are useful when processing multiple Seurat objects sequentially.
backup.percentile	(numeric) What percentile to use as cutoff in case flexmix fails to build a 2 cluster mixture model. Will only be used if backup.option is "percentile".
backup.percent	(numeric) What percent to use as cutoff in case flexmix fails to build a 2 cluster mixture model. Will only be used if backup.option is "percent".

Details

(Copied verbatim from miQC) _Function to fit a two-distribution mixture model on a Seurat object and find those cells probabistically determined to be compromised by the mixture model._

Value

Returns a Seurat object with probabilities and "keep" decisions stored as "miQC.probability" and "miQC.keep" in the object metadata, respectively.

References

Hippen et al. (2021) miQC: An adaptive probabilistic framework for quality control of single-cell RNA-sequencing data. bioRxiv doi: 10.1101/2021.03.03.433798

`RunOptimizeALS` *Run optimizeALS on a Seurat object*

Description

Run optimizeALS on a Seurat object

Usage

```
RunOptimizeALS(
  object,
  k,
  assay = NULL,
  split.by = "orig.ident",
  lambda = 5,
  thresh = 1e-06,
  max.iters = 30,
  reduction.name = "iNMF_raw",
  reduction.key = "riNMF_",
  nrep = 1,
  H.init = NULL,
  W.init = NULL,
  V.init = NULL,
  rand.seed = 1,
  print.obj = FALSE,
  ...
)
```

Arguments

`object`	A merged Seurat object
`k`	Inner dimension of factorization (number of factors). Run suggestK to determine appropriate value; a general rule of thumb is that a higher k will be needed for datasets with more sub-structure.
`assay`	Assay to use, defaults to the default assay of the first object
`split.by`	Attribute for splitting, defaults to "orig.ident"

lambda	Regularization parameter. Larger values penalize dataset-specific effects more strongly (ie. alignment should increase as lambda increases). Run suggest-Lambda to determine most appropriate value for balancing dataset alignment and agreement (default 5.0).
thresh	Convergence threshold. Convergence occurs when |obj0-objl/(mean(obj0,obj)) < thresh. (default 1e-6)
max.iters	Maximum number of block coordinate descent iterations to perform (default 30).
reduction.name	Name to store resulting DimReduc object as
reduction.key	Key for resulting DimReduc
nrep	Number of restarts to perform (iNMF objective function is non-convex, so taking the best objective from multiple successive initializations is recommended). For easier reproducibility, this increments the random seed by 1 for each consecutive restart, so future factorizations of the same dataset can be run with one rep if necessary. (default 1)
H.init	Initial values to use for H matrices. (default NULL)
W.init	Initial values to use for W matrix (default NULL)
V.init	Initial values to use for V matrices (default NULL)
rand.seed	Random seed to allow reproducible results (default 1).
print.obj	Print objective function values after convergence (default FALSE).
...	Arguments passed to other methods

Value

A Seurat object with embeddings and loadings from optimizeALS stored as a DimReduc object with name reduction.name (key set to reduction.key); per-dataset feature loadings matrices stored in the tool slot, accessible with Tool

See Also

optimizeALS Tool

RunPaCMAP *Run PaCMAP (Pairwise Controlled Manifold Approximation)*

Description

Runs PaCMAP, a method for dimensionality reduction for scRNA-seq data. data. Constructs three kinds of pairs of points: neighbor pairs (pair_neighbors), mid-near pair (pair_MN), and further pairs (pair_FP) based on positional relationship in the original space, and optimize a low-dimensional embedding accordingly. Described in Wang, Y., Huang, H., Rudin, C., & Shaposhnik, Y. (2021). "Understanding how dimension reduction tools work: an empirical approach to deciphering t-SNE, UMAP, TriMAP, and PaCMAP for data visualization." Journal of Machine Learning Research, 22(201), 1-73. This implementation is based on the work of Hao Zhang, as found in https://github.com/zhanghao-njmu/SCP/. We made modifications to ensure compatibility across multiple platforms, including Windows and macOS.

Usage

```
RunPaCMAP(object, ...)

## S3 method for class 'Seurat'
RunPaCMAP(
  object,
  reduction = "pca",
  dims = NULL,
  features = NULL,
  assay = NULL,
  layer = "data",
  n_components = 2,
  n.neighbors = NULL,
  MN_ratio = 0.5,
  FP_ratio = 2,
  distance_method = "euclidean",
  lr = 1,
  num_iters = 250L,
  apply_pca = TRUE,
  init = "random",
  reduction.name = "pacmap",
  reduction.key = "PaCMAP_",
  verbose = TRUE,
  seed.use = 11L,
  ...
)

## Default S3 method:
RunPaCMAP(
  object,
  assay = NULL,
  n_components = 2,
  n.neighbors = NULL,
  MN_ratio = 0.5,
  FP_ratio = 2,
  distance_method = "euclidean",
  lr = 1,
  num_iters = 250L,
  apply_pca = TRUE,
  init = "random",
  reduction.key = "PaCMAP_",
  verbose = TRUE,
  seed.use = 11L,
  ...
)
```

Arguments

object	An object. This can be a Seurat object or a matrix-like object.
...	Additional arguments to be passed to the pacmap.PaCMAP function.
reduction	A character string specifying the reduction to be used as input. Default is "pca".
dims	An integer vector specifying the dimensions to be used. Default is NULL.
features	A character vector specifying the features to be used. Default is NULL.
assay	A character string specifying the assay to be used. Default is NULL.
layer	A character string specifying the layer name to be used. Default is "data".
n_components	An integer specifying the number of PaCMAP components. Default is 2.
n.neighbors	An integer specifying the number of neighbors considered in the k-Nearest Neighbor graph. Default to 10 for dataset whose sample size is smaller than 10000. For large dataset whose sample size (n) is larger than 10000, the default value is: 10 + 15 * (log10(n) - 4).
MN_ratio	A numeric value specifying the ratio of the ratio of the number of mid-near pairs to the number of neighbors. Default is 0.5.
FP_ratio	A numeric value specifying the ratio of the ratio of the number of further pairs to the number of neighbors. Default is 2.
distance_method	A character string specifying the distance metric to be used. Default is "euclidean".
lr	A numeric value specifying the learning rate of the AdaGrad optimizer. Default is 1.
num_iters	An integer specifying the number of iterations for PaCMAP optimization. Default is 450.
apply_pca	A logical value indicating whether pacmap should apply PCA to the data before constructing the k-Nearest Neighbor graph. Using PCA to preprocess the data can largely accelerate the DR process without losing too much accuracy. Notice that this option does not affect the initialization of the optimization process. Default is TRUE.
init	A character string specifying the initialization of the lower dimensional embedding. One of "pca" or "random". Default is "random".
reduction.name	A character string specifying the name of the reduction to be stored in the Seurat object. Default is "pacmap".
reduction.key	A character string specifying the prefix for the column names of the PaCMAP embeddings. Default is "PaCMAP_".
verbose	A logical value indicating whether to print verbose output. Default is TRUE.
seed.use	An integer specifying the random seed to be used. Default is 11.
slot	A character string specifying the slot name to be used. Default is "data".

Author(s)

Yiyang Sun, Haiyang Huang, Gaurav Rajesh Parikh

References

Wang, Y., Huang, H., Rudin, C., & Shaposhnik, Y. (2021). "Understanding how dimension reduction tools work: an empirical approach to deciphering t-SNE, UMAP, TriMAP, and PaCMAP for data visualization." Journal of Machine Learning Research, 22(201), 1-73.

Examples

```
pancreas_sub <- Seurat::FindVariableFeatures(pancreas_sub)
pancreas_sub <- RunPaCMAP(object = pancreas_sub, features = Seurat::VariableFeatures(pancreas_sub))
DimPlot(pancreas_sub, reduction = "pacmap")
```

RunPresto *A Presto-based implementation of FindMarkers that runs Wilcoxon tests for the given identity classes*

Description

A Presto-based implementation of FindMarkers that runs Wilcoxon tests for the given identity classes

Usage

```
RunPresto(
  object,
  ident.1 = NULL,
  ident.2 = NULL,
  group.by = NULL,
  subset.ident = NULL,
  assay = NULL,
  slot = "data",
  reduction = NULL,
  features = NULL,
  logfc.threshold = 0.25,
  test.use = "wilcox",
  min.pct = 0.1,
  min.diff.pct = -Inf,
  verbose = TRUE,
  only.pos = FALSE,
  max.cells.per.ident = Inf,
  random.seed = 1,
  latent.vars = NULL,
  min.cells.feature = 3,
  min.cells.group = 3,
  mean.fxn = NULL,
  fc.name = NULL,
  base = 2,
```

```
  ...
)
```

Arguments

ident.1	Identity class to define markers for; pass an object of class phylo or 'clustertree' to find markers for a node in a cluster tree; passing 'clustertree' requires BuildClusterTree to have been run
ident.2	A second identity class for comparison; if NULL, use all other cells for comparison; if an object of class phylo or 'clustertree' is passed to ident.1, must pass a node to find markers for
group.by	Regroup cells into a different identity class prior to performing differential expression (see example)
subset.ident	Subset a particular identity class prior to regrouping. Only relevant if group.by is set (see example)
assay	Assay to use in differential expression testing
slot	Slot to pull data from; note that if test.use is "negbinom", "poisson", or "DESeq2", slot will be set to "counts"
reduction	Reduction to use in differential expression testing - will test for DE on cell embeddings
mean.fxn	Function to use for fold change or average difference calculation. If NULL, the appropriate function will be chose according to the slot used
fc.name	Name of the fold change, average difference, or custom function column in the output data.frame. If NULL, the fold change column will be named according to the logarithm base (eg, "avg_log2FC"), or if using the scale.data slot "avg_diff".
base	The base with respect to which logarithms are computed.

See Also

https://github.com/immunogenomics/presto

RunPrestoAll *A Presto-based implementation of FindAllMarkers that runs Wilcoxon tests for all identity classes*

Description

Finds markers (Wilcoxon-differentially expressed genes) for each of the identity classes in a dataset

Usage

```
RunPrestoAll(
  object,
  assay = NULL,
  features = NULL,
  logfc.threshold = 0.25,
  test.use = "wilcox",
  slot = "data",
  min.pct = 0.1,
  min.diff.pct = -Inf,
  node = NULL,
  verbose = TRUE,
  only.pos = FALSE,
  max.cells.per.ident = Inf,
  random.seed = 1,
  latent.vars = NULL,
  min.cells.feature = 3,
  min.cells.group = 3,
  mean.fxn = NULL,
  fc.name = NULL,
  base = 2,
  return.thresh = 0.01,
  ...
)
```

Arguments

assay	Assay to use in differential expression testing
slot	Slot to pull data from; note that if test.use is "negbinom", "poisson", or "DESeq2", slot will be set to "counts"
node	A node to find markers for and all its children; requires BuildClusterTree to have been run previously; replaces FindAllMarkersNode
mean.fxn	Function to use for fold change or average difference calculation. If NULL, the appropriate function will be chose according to the slot used
fc.name	Name of the fold change, average difference, or custom function column in the output data.frame. If NULL, the fold change column will be named according to the logarithm base (eg, "avg_log2FC"), or if using the scale.data slot "avg_diff".
base	The base with respect to which logarithms are computed.
return.thresh	Only return markers that have a p-value < return.thresh, or a power > return.thresh (if the test is ROC)

Value

Matrix containing a ranked list of putative markers, and associated statistics (p-values, logFC, etc.)

See Also

https://github.com/immunogenomics/presto

`RunQuantileAlignSNF` *Run quantileAlignSNF on a Seurat object*

Description

This is a deprecated function. Call 'RunQuantileNorm' instead.

Usage

```
RunQuantileAlignSNF(
  object,
  split.by = "orig.ident",
  reduction = "iNMF_raw",
  reduction.name = "iNMF",
  reduction.key = "iNMF_",
  recalc.snf = FALSE,
  ref_dataset = NULL,
  prune.thresh = 0.2,
  min_cells = 2,
  quantiles = 50,
  nstart = 10,
  resolution = 1,
  center = FALSE,
  id.number = NULL,
  print.mod = FALSE,
  print.align.summary = FALSE,
  ...
)
```

Arguments

`object`	A merged Seurat object
`split.by`	Attribute for splitting, defaults to "orig.ident"
`reduction`	Name of reduction to use
`reduction.name`	Name to store resulting DimReduc object as
`reduction.key`	Key for resulting DimReduc
`recalc.snf`	Recalculate SNF
`ref_dataset`	Name of dataset to use as a "reference" for normalization. By default, the dataset with the largest number of cells is used.
`prune.thresh`	Minimum allowed edge weight. Any edges below this are removed (given weight 0) (default 0.2)

min_cells	Minimum number of cells to consider a cluster shared across datasets (default 2)
quantiles	Number of quantiles to use for quantile normalization (default 50).
nstart	Number of times to perform Louvain community detection with different random starts (default 10).
resolution	Controls the number of communities detected. Higher resolution -> more communities. (default 1)
center	Centers the data when scaling factors (useful for less sparse modalities like methylation data). (default FALSE)
id.number	Number to use for identifying edge file (when running in parallel) (generates random value by default).
print.mod	Print modularity output from clustering algorithm (default FALSE).
print.align.summary	Print summary of clusters which did not align normally (default FALSE).
...	Arguments passed to other methods, and to SNF if recalc.snf = TRUE or SNF hasn't been run

Value

A Seurat object with embeddings from quantileAlignSNF stored as a DimReduc object with name reduction.name (key set to reduction.key)

See Also

RunQuantileNorm

RunQuantileNorm *Run quantile_norm on a Seurat object*

Description

Run quantile_norm on a Seurat object

Usage

```
RunQuantileNorm(
  object,
  split.by = "orig.ident",
  reduction = "iNMF_raw",
  reduction.name = "iNMF",
  reduction.key = "iNMF_",
  quantiles = 50,
  ref_dataset = NULL,
  min_cells = 20,
  knn_k = 20,
```

```
  dims.use = NULL,
  do.center = FALSE,
  max_sample = 1000,
  eps = 0.9,
  refine.knn = TRUE,
  ...
)
```

Arguments

object	A merged Seurat object
split.by	Attribute for splitting, defaults to "orig.ident"
reduction.name	Name to store resulting DimReduc object as
reduction.key	Key for resulting DimReduc
quantiles	Number of quantiles to use for quantile normalization (default 50).
ref_dataset	Name of dataset to use as a "reference" for normalization. By default, the dataset with the largest number of cells is used.
min_cells	Minimum number of cells to consider a cluster shared across datasets (default 20)
knn_k	Number of nearest neighbors for within-dataset knn graph (default 20).
dims.use	Indices of factors to use for shared nearest factor determination (default 1:ncol(H[[1]])).
do.center	Centers the data when scaling factors (useful for less sparse modalities like methylation data). (default FALSE)
max_sample	Maximum number of cells used for quantile normalization of each cluster and factor. (default 1000)
eps	The error bound of the nearest neighbor search. (default 0.9) Lower values give more accurate nearest neighbor graphs but take much longer to computer.
refine.knn	whether to increase robustness of cluster assignments using KNN graph.(default TRUE)
...	Arguments passed to other methods

Value

A Seurat object with embeddings from quantile_norm stored as a DimReduc object with name reduction.name (key set to reduction.key)

See Also

quantile_norm

RunSNF *Generate shared factor neighborhood graph*

Description

This is a deprecated function. Call 'RunQuantileNorm' instead.

Usage

```
RunSNF(
  object,
  split.by = "orig.ident",
  reduction = "iNMF_raw",
  dims.use = NULL,
  dist.use = "CR",
  center = FALSE,
  knn_k = 20,
  k2 = 500,
  small.clust.thresh = knn_k,
  ...
)
```

Arguments

`object`	A merged Seurat object
`split.by`	Attribute for splitting, defaults to "orig.ident"
`reduction`	Name of reduction to use
`...`	Arguments passed to other methods

Value

A Seurat object with the SNF list stored in the `tool` slot, accessible with `Tool`

See Also

`RunQuantileNorm` `Tool`

Runtricycle *Run estimate_cycle_position on a Seurat object*

Description

This function run estimate_cycle_position function on Seurat object. It uses the tricycle internal reference projection matrix.

Usage

```
Runtricycle(
  object,
  assay = NULL,
  slot = "data",
  reduction.name = "tricycleEmbedding",
  reduction.key = "tricycleEmbedding_",
  gname = NULL,
  gname.type = c("ENSEMBL", "SYMBOL"),
  species = c("mouse", "human"),
  AnnotationDb = NULL,
  center.pc1 = 0,
  center.pc2 = 0
)
```

Arguments

object	Seurat object
assay	Assay to use, defaults to the default assay
slot	Slot to use. It should be library size adjusted **log-expression** values. Note that it is convention that we rename "logcounts" to "data" when converting SingleCellExperiment to Seurat object. See also as.Seurat. Defaults to "data"
reduction.name	Name of the cell cycle projection returned
reduction.key	Key for the cell cycle projection returned
gname	Alternative rownames of object. If provided, this will be used to map genes within object with genes in reference. If not provided, the rownames of object will be used instead. Default: NULL
gname.type	The type of gene names as in gname or rownames of object. It can be either 'ENSEMBL' or 'SYMBOL'. Default: 'ENSEMBL'
species	The type of species in object. It can be either 'mouse' or 'human'. Default: 'mouse'
AnnotationDb	An AnnotationDb objects. If the user provides rownames in the format of Ensembl IDs and project human data, this object will be used to map Ensembl IDs to gene SYMBOLs. If no AnnotationDb object being given, the function will use org.Hs.eg.db.

center.pc1	The center of PC1 when defining the angle. Default: 0
center.pc2	The center of PC2 when defining the angle. Default: 0

RunVelocity *Run RNA Velocty*

Description

Run RNA Velocty

Usage

```
RunVelocity(
  object,
  spliced = "spliced",
  unspliced = "unspliced",
  ambiguous = NULL,
  spliced.average = 0.2,
  unspliced.average = 0.05,
  reduction = "pca",
  group.by = "ident",
  cells = NULL,
  graph = NULL,
  ncores = 1,
  verbose = TRUE,
  ...
)
```

Arguments

object	A Seurat object
spliced	Name of spliced assay
unspliced	Name of unspliced assay
ambiguous	Optional name of ambiguous assay
spliced.average, unspliced.average	Required minimum average expression count for the spliced and unspliced expression matrices
reduction	Name of reduction to use
group.by	Factor to group cells by
cells	Vector of cells to use; defaults to all cells (see gene.relative.velocity.estimates:steady.state.ce
graph	Optional name of nearest neighbor graph to use
ncores	Number of cores to use
verbose	Display progress updates
...	Extra parameters passed to gene.relative.velocity.estimates

Value

...

See Also

gene.relative.velocity.estimates Tool

---

scVIIntegration *scVI Integration*

Description

scVI Integration

Usage

```
scVIIntegration(
  object,
  features = NULL,
  layers = "counts",
  conda_env = NULL,
  new.reduction = "integrated.dr",
  ndims = 30,
  nlayers = 2,
  gene_likelihood = "nb",
  max_epochs = NULL,
  ...
)
```

Arguments

object	A StdAssay or STDAssay instance containing merged data
features	Features to integrate
layers	Layers to integrate
conda_env	conda environment to run scVI
new.reduction	Name under which to store resulting DimReduc object
ndims	Dimensionality of the latent space
nlayers	Number of hidden layers used for encoder and decoder NNs
gene_likelihood	Distribution to use for modelling expression data: "zinb", "nb", "poisson"
max_epochs	Number of passes through the dataset taken while training the model
...	Unused - currently just capturing parameters passed in from Seurat::IntegrateLayers intended for other integration methods

Value

A single-element named list DimReduc elements containing the integrated data

Note

This function requires the **scvi-tools** package to be installed

See Also

scVI

Examples

```
## Not run:
# Preprocessing
obj <- SeuratData::LoadData("pbmcsca")
obj[["RNA"]] <- split(obj[["RNA"]], f = obj$Method)
obj <- NormalizeData(obj)
obj <- FindVariableFeatures(obj)
obj <- ScaleData(obj)
obj <- RunPCA(obj)

# After preprocessing, we integrate layers, specifying a conda environment
obj <- IntegrateLayers(
  object = obj,
  method = scVIIntegration,
  new.reduction = "integrated.scvi",
  conda_env = "../miniconda3/envs/scvi-env",
  verbose = FALSE
)

# Alternatively, we can integrate SCTransformed data
obj <- SCTransform(object = obj)
obj <- IntegrateLayers(
  object = obj, method = scVIIntegration,
  orig.reduction = "pca", new.reduction = "integrated.scvi",
  assay = "SCT", conda_env = "../miniconda3/envs/scvi-env", verbose = FALSE
)

## End(Not run)
```

StopCellbrowser *Stop Cellbrowser web server*

Description

Stop Cellbrowser web server

Usage

```
StopCellbrowser()
```

Examples

```
## Not run:
StopCellbrowser()

## End(Not run)
```

`writeSparseTsvChunks` *Used by* `ExportToCellbrowser`*: Write a big sparse matrix to a .tsv.gz file by writing chunks, concating them with the Unix cat command, then gziping the result. This does not work on Windows, we'd have to use the copy /b command there.*

Description

Used by `ExportToCellbrowser`: Write a big sparse matrix to a .tsv.gz file by writing chunks, concating them with the Unix cat command, then gziping the result. This does not work on Windows, we'd have to use the copy /b command there.

Usage

```
writeSparseTsvChunks(inMat, outFname, sliceSize = 1000)
```

Arguments

`inMat`	input matrix
`outFname`	output file name, has to end with .gz
`sliceSize=1000`	size of each chunk in number of lines

Value

Invisibly returns `NULL`

Examples

```
## Not run:
writeSparseTsvChunks( pbmc_small@data, "exprMatrix.tsv.gz")

## End(Not run)
```

Index

alevin, *13*
ALRAChooseKPlot, 3, *16*
as.cell_data_set, 4
as.CellDataSet *(*as.cell_data_set*)*, 4
as.Seurat, 5, *6, 7*, *34*
as.Seurat.SingleCellExperiment, *7*
as.SingleCellExperiment, *5*

BuildClusterTree, *28, 29*

cell_data_set, *12*
CellBrowser, 7
CoGAPS, *19*
CogapsParams, *19*
ComputeBanksy, *18*
CreateSeuratObject, *7*

DimReduc, *19*

ExportToCellbrowser *(*CellBrowser*)*, 7

fastMNN, *10*, *20*
FastMNNIntegration, 9
findMatrix, 11

gene.relative.velocity.estimates, *35*, *36*
glmpca, *21*
global, *4*
Graph, *7*

igraph, *5*

learn_graph, *12*
LearnGraph, 11

matrix, *6*
merge, *6*

optimizeALS, *24*
optimizeALS *(*RunOptimizeALS*)*, 23

org.Hs.eg.db, *34*

PlotMiQC, 12

quantile_norm, *32*
quantile_norm *(*RunQuantileNorm*)*, 31
quantileAlignSNF, *31*
quantileAlignSNF *(*RunQuantileAlignSNF*)*, 30

read.loom.matrices, *14*
ReadAlevin, 13
ReadVelocity, 14
reducedDims, *5*
RunALRA, *4*, 14
RunBanksy, 16
RunCoGAPS, 18
RunFastMNN, 19
RunGLMPCA, 20
RunMiQC, 21
RunOptimizeALS, 23
RunPaCMAP, 24
RunPresto, 27
RunPrestoAll, 28
RunPrestoAllNode *(*RunPrestoAll*)*, 28
RunQuantileAlignSNF, 30
RunQuantileNorm, 31, *31*, *33*
RunSNF, 33
Runtricycle, 34
RunVelocity, 35

scVIIntegration, 36
SelectIntegrationFeatures, *20*
Seurat, *5*, *11, 12*
SeuratWrappers *(*SeuratWrappers-package*)*, 2
SeuratWrappers-package, 2
SingleCellExperiment, *7*
SNF, *30, 31*
SNF *(*RunSNF*)*, 33

StopCellbrowser, 37

Tool, *10*, *20*, *24*, *33*, *36*

writeSparseTsvChunks, 38