Package 'augere.solo'

June 23, 2026

Version 0.99.3

Date 2026-04-11

Title Automatic Generation of Single-Cell Analyses

Description Implements pipelines for generating single-cell analysis reports in the augere framework. This uses scrapper to execute routine steps such as quality control, normalization, feature selection, clustering and marker detection.
We also implement a pipeline for automatic cell type annotation against a labelled reference with SingleR.
Each pipeline function generates a self-contained Rmarkdown report with all of the steps required to reproduce its analysis.

License MIT + file LICENSE

Imports augere.core, scrapper, scater

Suggests testthat, knitr, rmarkdown, BiocStyle, BiocGenerics, S4Vectors, IRanges, GenomicRanges, SummarizedExperiment, SingleCellExperiment, scRNAseq, SingleR, celldex, jsonlite

VignetteBuilder knitr

RoxygenNote 7.3.3

Encoding UTF-8

URL https://github.com/augere-bioinfo/augere.solo

BugReports https://github.com/augere-bioinfo/augere.solo/issues

biocViews WorkflowManagement, ReportWriting, SingleCell

git_url https://git.bioconductor.org/packages/augere.solo

git_branch devel

git_last_commit 51e31e9

git_last_commit_date 2026-05-14

Repository Bioconductor 3.24

Date/Publication 2026-06-23

Author Aaron Lun [cre, aut] (ORCID: <https://orcid.org/0000-0002-3564-4813>)

Maintainer Aaron Lun <infinite.monkeys.with.keyboards@gmail.com>

Contents

`augere.solo-package` *augere.solo: Automatic Generation of Single-Cell Analyses*

Description

Implements pipelines for generating single-cell analysis reports in the augere framework. This uses scrapper to execute routine steps such as quality control, normalization, feature selection, clustering and marker detection. We also implement a pipeline for automatic cell type annotation against a labelled reference with SingleR. Each pipeline function generates a self-contained Rmarkdown report with all of the steps required to reproduce its analysis.

Author(s)

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

See Also

Useful links:

- https://github.com/augere-bioinfo/augere.solo
- Report bugs at https://github.com/augere-bioinfo/augere.solo/issues

---

`reexports` *Objects exported from other packages*

Description

These objects are imported from other packages. Follow the links below to see their documentation.

augere.core `readResult`, `wrapInput`

`runAnnotate` *Cell type annotation from scRNA-seq data*

Description

Annotate cells in a scRNA-seq dataset by computing correlations against reference data with known labels.

Usage

```
runAnnotate(
  test,
  references,
  test.assay = 1,
  test.id.field = NULL,
  test.block.field = NULL,
  test.is.lognorm = (test.assay == "logcounts"),
  test.is.ensembl = FALSE,
  test.symbol.field = NULL,
  cluster.field = NULL,
  reduced.dimensions = NULL,
  output.dir = "annotate",
  metadata = NULL,
  author = NULL,
  dry.run = FALSE,
  save.results = TRUE,
  suppress.plots = FALSE,
  num.threads = 1
)

configureReferenceAnnotation(
  ref,
  ref.label.field,
  ref.marker.method = NULL,
  ref.num.markers = NULL,
  ref.assay = "logcounts",
  ref.id.field = NULL,
  ref.block.field = NULL,
  ref.aggregate = NULL,
  ref.is.lognorm = (ref.assay == "logcounts")
)
```

Arguments

`test`	A SummarizedExperiment object containing cells in the test dataset to be assigned labels.
`references`	A list created by `configureReferenceAnnotation`, containing a configuration for a reference dataset. Alternatively, a list of these lists may be supplied to specify multiple references. This list may be named, in which case the names will be used to identify each reference in both the report and output of this function.

test.assay
: Integer or string specifying the assay of test containing the expression values to use for classification. This is expected to be raw counts or log-transformed normalized expression values.

test.id.field
: String specifying the name of the rowData(test) column containing the common gene identifiers. These identifiers should be consistent with those that are used in each entry of references. If NULL, the row names are assumed to contain the relevant identifiers.

test.block.field
: String specifying the name of the colData(test) column containing the block assignment for each cell. This is used to ensure that some marker-related diagnostics are not driven by uninteresting differences between blocks. If NULL, all cells in test are assumed to originate from the same block.

test.is.lognorm
: Boolean indicating whether the assay at test.assay contains log-normalized values. If FALSE, it is assumed to contain counts.

test.is.ensembl
: Boolean indicating whether rownames(test) contains Ensembl IDs. Otherwise, it is assumed to contain gene symbols. Only relevant for entries of references where ref is a **celldex** reference, where it is used to choose between Ensembl IDs or gene symbols for the row names.

test.symbol.field
: String specifying the name of the rowData(test) column that contains gene symbols. This used to improve the interpretability of some of the gene-based diagnostics in the report. If NULL, no symbols are added.

cluster.field
: String specifying the column of colData(test) with cluster assignments for each cell. If provided, this is used to generate some diagnostics to compare the empirical clusters to the predicted labels.

reduced.dimensions
: Character vector of reduced dimensions in reducedDims(test) to plot with the predicted labels in the report.

output.dir
: String containing the path to an output directory in which to write the Rmarkdown report and save results.

metadata
: Named list of additional fields to add to each result's metadata.

author
: Character vector of authors.

dry.run
: Boolean indicating whether to perform a dry run. This will write the Rmarkdown report without evaluating it.

save.results
: Boolean indicating whether the results should also be saved to file.

suppress.plots
: Boolean indicating whether to suppress the generation of plots. This can be set to TRUE for faster execution.

num.threads
: Integer specifying the number of threads to use in the various computations.

ref
: A SummarizedExperiment object containing reference samples with known labels in colData(ref)[[label.field]].

  Alternatively, a string can be provided containing the name of **celldex** reference dataset (e.g., "HumanPrimaryCellAtlasData"). In such cases, label.field is typically "label.main" or "label.fine".

ref.label.field
: String specifying the name of the colData(ref) column containing the label for each reference sample. For **celldex** references, this is usually either "label.main" or "label.fine".

`ref.marker.method`
: String specifying the method for choosing the top markers from each pairwise comparison between labels, see the `de.method=` argument in `trainSingleR` for more details. If `NULL`, this defaults to `"classic"` for the **celldex** references and `"t"` otherwise.

`ref.num.markers`
: Integer specifying the number of markers to use from each pairwise comparison between labels. See the `de.n=` argument in `trainSingleR` for more details.

`ref.assay`
: Integer or string specifying the assay of `ref` containing the expression values to use for creating references. This should be a matrix-like object that contains counts or their log-normalized values.

`ref.id.field`
: String specifying the name of the `rowData(ref)` column containing the common gene identifiers. These identifiers should be consistent with those that are used in `test`. If `NULL`, the row names are assumed to contain the relevant identifiers.

`ref.block.field`
: String specifying the name of the `colData(ref)` column containing the block assignment for each sample. This is used to ensure that marker calculations are not driven by uninteresting differences between blocks. If `NULL`, all cells in `test` are assumed to originate from the same block.

`ref.aggregate`
: Boolean indicating that references should be aggregated inside `trainSingleR`. This can be set to `TRUE` for faster classification of large single-cell references.

`ref.is.lognorm`
: Boolean indicating whether the assay at `assay` contains log-normalized values. If `FALSE`, it is assumed to contain counts.

Value

For `runAnnotate`, a Rmarkdown report named `report.Rmd` is written inside `output.dir` that contains the analysis commands.

If `dry.run=FALSE`, a list is returned containing:

- `predictions`, a list of length equal to `references`. Each entry is a DataFrame containing the classification results of `test` against the corresponding reference. See `classifySingleR` for more details on the expected columns.
- `combined`, a DataFrame containing the combined classification results across multiple references. See `combineRecomputedResults` for more details on the expected format. Only present if `references` contains more than one reference.

If `save.results=TRUE`, the results are saved in a `results` directory inside `output`.

If `dry.run=TRUE`, `NULL` is returned. Only the Rmarkdown report is saved to file.

For `configureReferenceAnnotation`, a list of class `"reference"` is returned containing the configuration details for each reference.

Examples

```
library(scRNAseq)
hESCs <- LaMannoBrainData('human-es')
hESCs <- hESCs[,1:100] # subsetting to speed it up.

tmp <- tempfile()
results <- runAnnotate(
    hESCs,
```

```
    configureReferenceAnnotation(
        "HumanPrimaryCellAtlasData",
        "label.main"
    ),
    output.dir = tmp,
    num.threads = 2 # speed it up a little.
)

list.files(tmp, recursive=TRUE)
results$predictions
```

---

runSolo *Simple analysis of single-cell data*

Description

Simple analysis of scRNA-seq or CITE-seq data, from quality control to clustering and marker gene detection.

Usage

```
runSolo(
  x,
  rna.experiment = TRUE,
  adt.experiment = NULL,
  subset.factor = NULL,
  subset.levels = NULL,
  block.field = NULL,
  qc.mito.seqnames = c("MT", "M", "chrM", "chrMT"),
  qc.mito.regex = NULL,
  qc.igg.regex = "IgG|igg|IGG",
  qc.num.mads = 3,
  qc.filter = TRUE,
  num.hvgs = 2000,
  num.pcs = 25,
  mnn.num.neighbors = 15,
  mnn.num.steps = 1,
  cluster.method = "graph",
  cluster.kmeans.k = 10,
  cluster.graph.method = c("multilevel", "leiden", "walktrap"),
  cluster.graph.num.neighbors = 10,
  cluster.graph.resolution = NULL,
  reduced.dimensions = c("tsne", "umap"),
  tsne.perplexity = 30,
  umap.num.neighbors = 15,
  umap.min.dist = 0.1,
  marker.effect.size = c("cohens.d", "auc", "delta.mean", "delta.detected"),
  marker.summary = c("min.rank", "mean", "median", "min"),
  marker.lfc.threshold = 0,
  assay = 1,
  symbol.field = NULL,
```

```
  metadata = NULL,
  output.dir = "solo",
  author = NULL,
  dry.run = FALSE,
  save.results = TRUE,
  suppress.plots = FALSE,
  num.threads = 1
)
```

Arguments

x	A SummarizedExperiment object where each column represents a single cell. Rows are usually expected to contain genes or antibody-derived tags, see rna.experiment= and adt.experiment= for details.
rna.experiment	Identity of the experiment containing the RNA data, when x is a SingleCellExperiment. If TRUE, the main experiment is assumed to contain the RNA data (see mainExpName). If a string is supplied, it is treated as the name of the alteranative experiment (see altExps) containing the RNA data. If FALSE or NULL, it is assumed that no RNA data is available.
adt.experiment	Identity of the experiment containing the ADT data, when x is a SingleCellExperiment. If TRUE, the main experiment is assumed to contain the ADT data (see mainExpName). If a string is supplied, it is treated as the name of the alteranative experiment (see altExps) containing the ADT data. If FALSE or NULL, it is assumed that no ADT data is available.
subset.factor	String specifying the name of the colData(se) column containing a factor with which to subset the cells.
subset.levels	Vector containing the subset of levels to retain in the factor specified by subset.factor.
block.field	String specifying the name of the colData(se) column containing the block assignment for each cell.
qc.mito.seqnames	Character vector containing the sequence names of the mitochondrial chromosome. Only used if rna.experiment= indicates that RNA data is available and qc.mito.regex= is not specified.
qc.mito.regex	String containing a regular expression to identify mitochondrial genes from the row names. If symbol.field= is specified, this is applied to the gene symbols instead. If NULL, the mitochondrial genes are identified from the sequence names of rowRanges matching qc.mito.seqnames. Only used if rna.experiment= indicates that RNA data is available.
qc.igg.regex	String containing a regular expression to identify IgG controls from the row names. If symbol.field= is specified, this is applied to the gene symbols instead. Only used if adt.experiment= indicates that ADT data is available.
qc.num.mads	Integer specifying the number of median absolute deviations (MADs) with which to define a quality control (QC) filtering threshold. Smaller values increase the stringency of the filter.
qc.filter	Boolean indicating whether putative low-quality cells should be removed. If FALSE, low-quality cells are identified but not removed prior to further cells.
num.hvgs	Integer specifying the number of highly variable genes (HVGs) to retain for downstream analyses. More HVGs capture more biological signal at the cost of capturing more technical noise and increasing computational work. Only used if rna.experiment= indicates that RNA data is available.

num.pcs Integer specifying the number of top principal components (PCs) to retain for downstream analyses. More PCs capture more biological signal at the cost of capturing more technical noise and increasing computational work.

mnn.num.neighbors
Integer specifying the number of neighbors for batch correction with mutual nearest neighbors. Larger values improve stability but reduce resolution for rare subpopulations.

mnn.num.steps Integer specifying the number of steps for the center of mass calculation during batch correction with mutual nearest neighbors. Larger values improve intermingling of cells from different batches but increase the risk of merging the wrong subpopulations.

cluster.method Character vector specifying the clustering methods to run on the top PCs, namely graph-based clustering ("graph") or k-means clustering ("kmeans"). Multiple methods may be specified here to generate multiple clusterings, but only the clusters from the first method are used for marker detection.

cluster.kmeans.k
Integer specifying the number of clusters to generate from k-means clustering. Only used if "kmeans" is in cluster.method.

cluster.graph.method
String naming the community detection method to use in graph-based clustering. These are roughly equivalent to the functions of the same name from the **igraph** R package. (Note that "multilevel" is a synonym for the Louvain method.) Only used if "graph" is in cluster.method.

cluster.graph.num.neighbors
Integer specifying the number of nearest neighbors to use during construction of the shared-nearest neighbor graph. Larger values increase graph connectivity and decrease cluster resolution. Only used if "graph" is in cluster.method.

cluster.graph.resolution
Number specifying the resolution to use in the multi-level or Leiden community detection algorithms. Larger values usually result in a greater number of smaller clusters. Only used if "graph" is in cluster.method.

reduced.dimensions
Character vector specifying the dimensionality reduction algorithms to use for visualization. This can be zero, one or both of "tsne" (for t-SNEs) and "umap" (for UMAPs).

tsne.perplexity
Number specifying the perplexity to use in t-SNE. Larger values increase the size of each cell's neighborhood and focus more on global structure. Only used if "tsne" is in reduced.dimensions.

umap.num.neighbors
Integer specifying the number of neighbors to use in the UMAP. Larger values increase connectivity and focus more on global structure. Only used if "umap" is in reduced.dimensions.

umap.min.dist Number specifying the minimum distance between points in the UMAP. Larger values favor a more even distribution of cells throughout the low-dimensional space. Only used if "umap" is in reduced.dimensions.

marker.effect.size
String naming the effect size to use when ranking marker genes. This should be one of:

- "cohens.d": Cohen's d, i.e., the standardized difference in log-expression. This is analogous to the t-statistic in a two-group t-test and accounts for the variance within each group.
- "auc": AUC, i.e., the area under the curve. This is closely related to the U statistic in a Wilcoxon rank sum test, and is a more robust/less sensitive alternative Cohen's d.
- "delta.mean": difference in the mean log-expression. This is an estimate of the log-fold change.
- "delta.detected": difference in the proportion of cells with detected expression. Large differences correspond to genes that are silent in one group and activated in another.

This is combined with marker.summary to determine the actual statistic for ranking, see ?scoreMarkers.

marker.summary — String specifying the summary statistic to use when ranking marker genes. This should be one of:

- "min.rank": the minimum rank of each gene across all pairwise comparisons for a particular cluster. This creates a ranking where the top genes are guaranteed to separate the cluster of interest from every other cluster.
- "mean": the mean effect size of each gene across all pairwise comparisons for a particular cluster. This creates a ranking where the top genes are upregulated in the cluster of interest against the average of all other clusters.
- "median": the median effect size of each gene across all pairwise comparisons for a particular cluster. This creates a ranking where the top genes are upregulated in the cluster of interest against most other clusters.
- "min": the minimum effect size of each gene across all pairwise comparisons for a particular cluster. This creates a ranking where the top genes must be upregulated in the cluster of interest against all other clusters.

This is combined with marker.effect to determine the actual statistic for ranking, see ?scoreMarkers.

marker.lfc.threshold — Non-negative number specifying the log-fold change threshold to test against. Larger values focus on marker genes with larger log-fold changes at the expense of those with smaller variances.

assay — String or integer specifying the assay in x containing the count matrix. For SingleCellExperiment objects, the same assay is used for the main and alternative experiments.

symbol.field — String specifying the name of the rowData column that contains gene symbols. This is added to the various gene-based results, e.g., marker detection tables. If NULL, no symbols are added.

metadata — Named list of additional fields to add to each result's metadata.

output.dir — String containing the path to an output directory in which to write the Rmarkdown report and save results.

author — Character vector of authors.

dry.run — Boolean indicating whether to perform a dry run. This will write the Rmarkdown report without evaluating it.

save.results — Boolean indicating whether the results should also be saved to file.

suppress.plots — Boolean indicating whether to suppress the generation of plots. This can be set to TRUE for faster execution.

num.threads — Integer specifying the number of threads to use in the various computations.

Value

A Rmarkdown report named `report.Rmd` is written inside `output.dir` that contains the analysis commands.

If `dry.run=FALSE`, a list is returned containing:

- `sce`, a SingleCellExperiment object with dimensionality reduction and clustering results. This may have fewer cells than the input `x` if `qc.filter=TRUE`.
- `qc.rna`, a DataFrame object with RNA-based QC metrics for each cell in the input `x`. Only returned if `rna.experiment=` indicates that RNA data is available.
- `qc.adt`, a DataFrame object with ADT-based QC metrics for each cell in the input `x`. Only returned if `adt.experiment=` indicates that ADT data is available.
- `markers.rna`, a named list of DataFrame results containing marker gene statistics for each cluster. Only returned if `rna.experiment=` indicates that RNA data is available.
- `markers.adt`, a named list of DataFrame results containing marker tag statistics for each cluster. Only returned if `adt.experiment=` indicates that ADT data is available.

If `save.results=TRUE`, the results are saved in a `results` directory inside `output`.

If `dry.run=TRUE`, `NULL` is returned. Only the Rmarkdown report is saved to file.

Author(s)

Aaron Lun

Examples

```
library(scRNAseq)
se <- ZeiselBrainData()

tmp <- tempfile()
output <- runSolo(
    se,
    qc.mito.regex="^mt-",
    output.dir=tmp,
    num.threads = 2 # speed it up a little.
)

list.files(tmp, recursive=TRUE)
output$sce
output$markers
```

Index