# Package 'scMerge'

July 24, 2025

**Type** Package

**Title** scMerge: Merging multiple batches of scRNA-seq data

**Version** 1.24.0

**Description** Like all gene expression data, single-cell data suffers from batch effects and other unwanted variations that makes accurate biological interpretations difficult. The scMerge method leverages factor analysis, stably expressed genes (SEGs) and (pseudo-) replicates to remove unwanted variations and merge multiple single-cell data. This package contains all the necessary functions in the scMerge pipeline, including the identification of SEGs, replication-identification methods, and merging of single-cell data.

**License** GPL-3

**Encoding** UTF-8

**LazyData** false

**Depends** R (>= 3.6.0)

**Imports** BiocParallel, BiocSingular, BiocNeighbors, cluster, DelayedArray, DelayedMatrixStats, distr, igraph, M3Drop (>= 1.9.4), proxyC, ruv, cvTools, scater, batchelor, scran, methods, S4Vectors (>= 0.23.19), SingleCellExperiment (>= 1.7.3), SummarizedExperiment

**RoxygenNote** 7.2.3

**Suggests** BiocStyle, covr, HDF5Array, knitr, Matrix, rmarkdown, scales, proxy, testthat, badger

**VignetteBuilder** knitr

**biocViews** BatchEffect, GeneExpression, Normalization, RNASeq, Sequencing, SingleCell, Software, Transcriptomics

**URL** https://github.com/SydneyBioX/scMerge

**BugReports** https://github.com/SydneyBioX/scMerge/issues

**git_url** https://git.bioconductor.org/packages/scMerge

**git_branch** RELEASE_3_21

**git_last_commit** 6ed059a

**git_last_commit_date** 2025-04-15

**Repository** Bioconductor 3.21

**Date/Publication** 2025-07-23

**Author** Yingxin Lin [aut, cre],
Kevin Wang [aut],
Sydney Bioinformatics and Biometrics Group [fnd]

**Maintainer** Yingxin Lin <yingxin.lin@sydney.edu.au>

# Contents



---

`example_sce` *Subsetted mouse ESC 'SingleCellExperiment' object*

---

## Description

A dataset containing 300 cells and 2026 genes from two batches of mouse ESC data

## Usage

```
data(example_sce, package = 'scMerge')
```

## Format

A 'SingleCellExperiment' object

## Source

https://www.ebi.ac.uk/arrayexpress/experiments/E-MTAB-2600/

## References

Kolodziejczyk et al.

---

`fastRUVIII` *A fast version of the ruv::RUVIII algorithm*

---

## Description

Perform a fast version of the ruv::RUVIII algorithm for scRNA-Seq data noise estimation

## Usage

```
fastRUVIII(
  Y,
  M,
  ctl,
  k = NULL,
  eta = NULL,
  svd_k = 50,
  include.intercept = TRUE,
  average = FALSE,
  BPPARAM = SerialParam(),
  BSPARAM = ExactParam(),
  fullalpha = NULL,
  return.info = FALSE,
  inputcheck = TRUE
)
```

## Arguments

| | |
|---|---|
| `Y` | The unnormalised scRNA-Seq data matrix. A m by n matrix, where m is the number of observations and n is the number of features. |
| `M` | The replicate mapping matrix. The mapping matrix has m rows (one for each observation), and each column represents a set of replicates. The (i, j)-th entry of the mapping matrix is 1 if the i-th observation is in replicate set j, and 0 otherwise. See ruv::RUVIII for more details. |
| `ctl` | An index vector to specify the negative controls. Either a logical vector of length n or a vector of integers. |
| `k` | The number of unwanted factors to remove. This is inherited from the ruvK argument from the scMerge::scMerge function. |
| `eta` | Gene-wise (as opposed to sample-wise) covariates. See ruv::RUVIII for details. |
| `svd_k` | If BSPARAM is set to `RandomParam` or `IrlbaParam` class from `BiocSingular` package, then `svd_k` will be used to used to reduce the computational cost of singular value decomposition. Default to 50. |

include.intercept
: When eta is specified (not NULL) but does not already include an intercept term, this will automatically include one. See ruv::RUVIII for details.

average
: Average replicates after adjustment. See ruv::RUVIII for details.

BPPARAM
: A BiocParallelParam class object from the BiocParallel package is used. Default is SerialParam().

BSPARAM
: A BiocSingularParam class object from the BiocSingular package is used. Default is ExactParam().

fullalpha
: Not used. Please ignore. See ruv::RUVIII for details.

return.info
: Additional information relating to the computation of normalised matrix. We recommend setting this to true.

inputcheck
: We recommend setting this to true.

## Value

A normalised matrix of the same dimensions as the input matrix Y.

## Author(s)

Yingxin Lin, John Ormerod, Kevin Wang

## Examples

```
L = ruvSimulate(m = 200, n = 500, nc = 400, nCelltypes = 3, nBatch = 2, lambda = 0.1, sce = FALSE)
Y = L$Y; M = L$M; ctl = L$ctl
improved1 = scMerge::fastRUVIII(Y = Y, M = M, ctl = ctl,
k = 20, BSPARAM = BiocSingular::ExactParam())
improved2 = scMerge::fastRUVIII(Y = Y, M = M, ctl = ctl,
k = 20, BSPARAM = BiocSingular::RandomParam(), svd_k = 50)
old = ruv::RUVIII(Y = Y, M = M, ctl = ctl, k = 20)
all.equal(improved1, old)
all.equal(improved2, old)
```

---

# getAdjustedMat *getAdjustedMat*

---

## Description

Get Adjusted Matrix with scMerge2 parameter estimated

## Usage

```
getAdjustedMat(
  exprsMat,
  fullalpha,
  ctl = rownames(exprsMat),
  adjusted_means = NULL,
```

```
  ruvK = 20,
  return_subset_genes = NULL
)
```

## Arguments

| | |
|---|---|
| exprsMat | A gene (row) by cell (column) matrix to be adjusted. |
| fullalpha | A matrix indicates the estimated alpha returned by scMerge2(). |
| ctl | A character vector of negative control. It should have a non-empty intersection with the rows of exprsMat. |
| adjusted_means | A rowwise mean of the gene by cell matrix |
| ruvK | An integer indicates the number of unwanted variation factors that are removed, default is 20. |
| return_subset_genes | An optional character vector of indicates the subset of genes will be adjusted. |

## Value

Returns the adjusted matrix will be return.

## Author(s)

Yingxin Lin

## Examples

```
## Loading example data
data('example_sce', package = 'scMerge')
## Previously computed stably expressed genes
data('segList_ensemblGeneID', package = 'scMerge')
## Running an example data with minimal inputs
library(SingleCellExperiment)
scMerge2_res <- scMerge2(exprsMat = logcounts(example_sce),
batch = example_sce$batch,
ctl = segList_ensemblGeneID$mouse$mouse_scSEG,
return_matrix = FALSE)
cosineNorm_mat <- batchelor::cosineNorm(logcounts(example_sce))
adjusted_means <- DelayedMatrixStats::rowMeans2(cosineNorm_mat)
newY <- getAdjustedMat(cosineNorm_mat, scMerge2_res$fullalpha,
              ctl = segList_ensemblGeneID$mouse$mouse_scSEG,
              ruvK = 20,
              adjusted_means = adjusted_means)
assay(example_sce, "scMerge2") <- newY

example_sce = scater::runPCA(example_sce, exprs_values = 'scMerge2')
scater::plotPCA(example_sce, colour_by = 'cellTypes', shape = 'batch')
```

`ruvSimulate` *Simulate a simple matrix or SingleCellExperiment to test internals of scMerge*

## Description

This function is designed to generate Poisson-random-variable data matrix to test on the internal algorithms of scMerge. It does not represent real biological situations and it is not intended to be used by end-users.

## Usage

```
ruvSimulate(
  m = 100,
  n = 5000,
  nc = floor(n/2),
  nCelltypes = 3,
  nBatch = 2,
  k = 20,
  lambda = 0.1,
  sce = FALSE
)
```

## Arguments

| | |
|---|---|
| `m` | Number of observations |
| `n` | Number of features |
| `nc` | Number of negative controls |
| `nCelltypes` | Number of cell-types |
| `nBatch` | Number of batches |
| `k` | Number of unwanted factors in simulation |
| `lambda` | Rate parameter for random Poisson generation |
| `sce` | If `TRUE`, returns a SingleCellExperiment object |

## Value

If `sce` is FALSE, then the output is a list consists of

- Y, expression matrix generated through Poisson random variables,
- ctl, a logical vector indicating the control genes,
- M, replicate mapping matrix,
- cellTypes, a vector indicating simulated cell types
- batch, a vector indicating simulated batches

if `sce` is TRUE, a SingleCellExperiment wrapper will be applied on all above simulated objects.

## Examples

```
set.seed(1)
L = ruvSimulate(m = 200, n = 1000, nc = 200,
nCelltypes = 3, nBatch = 2, lambda = 0.1, k = 10, sce = TRUE)
print(L)
example <- scMerge(sce_combine = L,
                     ctl = paste0('gene', 1:500),
                     cell_type = L$cellTypes,
                     ruvK = 10,
                     assay_name = 'scMerge')
L = scater::runPCA(L, exprs_values = "logcounts")
scater::plotPCA(L, colour_by = 'cellTypes', shape = 'batch')

example = scater::runPCA(example, exprs_values = 'scMerge')
scater::plotPCA(example, colour_by = 'cellTypes', shape = 'batch')
```

---

# sce_cbind *Combind several* SingleCellExperiment *objects from different batches/experiments*

---

## Description

Combind several SingleCellExperiment objects from different batches/experiments.

## Usage

```
sce_cbind(
  sce_list,
  method = "intersect",
  cut_off_batch = 0.01,
  cut_off_overall = 0.01,
  exprs = c("counts", "logcounts"),
  colData_names = NULL,
  batch_names = NULL
)
```

## Arguments

| | |
|---|---|
| sce_list | A list contains the SingleCellExperiment Object from each batch |
| method | A string indicates the method of combining the gene expression matrix, either union or intersect. Default to intersect. union only supports matrix class. |
| cut_off_batch | A numeric vector indicating the cut-off for the proportion of a gene is expressed within each batch |
| cut_off_overall | A numeric vector indicating the cut-off for the proportion of a gene is expressed overall data |

| | |
|---|---|
| exprs | A string vector indicating the expression matrices to be combined. The first assay named will be used to determine the proportion of zeores. |
| colData_names | A string vector indicating the colData that are combined |
| batch_names | A string vector indicating the batch names for the output sce object |

## Value

A SingleCellExperiment object with the list of SCE objects combined.

## Author(s)

Yingxin Lin

## Examples

```
data('example_sce', package = 'scMerge')
batch_names<-unique(example_sce$batch)
sce_list<-list(example_sce[,example_sce$batch=='batch2'],
               example_sce[,example_sce$batch=='batch3'])
sce_combine<-sce_cbind(sce_list,batch_names=batch_names)
```

---

# scMerge *Perform the scMerge algorithm*

---

## Description

Merge single-cell RNA-seq data from different batches and experiments leveraging (pseudo)-replicates and control genes.

## Usage

```
scMerge(
  sce_combine,
  ctl = NULL,
  kmeansK = NULL,
  exprs = "logcounts",
  hvg_exprs = "counts",
  batch_name = "batch",
  marker = NULL,
  marker_list = NULL,
  ruvK = 20,
  replicate_prop = 1,
  cell_type = NULL,
  cell_type_match = FALSE,
  cell_type_inc = NULL,
  BSPARAM = ExactParam(),
  svd_k = 50,
```

```
  dist = "cor",
  WV = NULL,
  WV_marker = NULL,
  BPPARAM = SerialParam(),
  return_all_RUV = FALSE,
  assay_name = NULL,
  plot_igraph = TRUE,
  verbose = FALSE
)
```

## Arguments

| | |
|---|---|
| sce_combine | A SingleCellExperiment object contains the batch-combined matrix with batch info in colData. |
| ctl | A character vector of negative control. It should have a non-empty intersection with the rows of sce_combine. |
| kmeansK | A vector indicates the kmeans's K for each batch. The length of kmeansK needs to be the same as the number of batch. |
| exprs | A string indicating the name of the assay requiring batch correction in sce_combine, default is logcounts. |
| hvg_exprs | A string indicating the assay that to be used for highly variable genes identification in sce_combine, default is counts. |
| batch_name | A character indicating the name of the batch column, default to "batch" |
| marker | An optional vector of markers, to be used in calculation of mutual nearest cluster. If no markers input, highly variable genes will be used instead. |
| marker_list | An optional list of markers for each batch, which will be used in calculation of mutual nearest cluster. |
| ruvK | An optional integer/vector indicating the number of unwanted variation factors that are removed, default is 20. |
| replicate_prop | A number indicating the ratio of cells that are included in pseudo-replicates, ranges from 0 to 1. Default to 1. |
| cell_type | An optional vector indicating the cell type information for each cell in the batch-combined matrix. If it is NULL, pseudo-replicate procedure will be run to identify cell type. |
| cell_type_match | An optional logical input for whether to find mutual nearest cluster using cell type information. |
| cell_type_inc | An optional vector indicating the indices of the cells that will be used to supervise the pseudo-replicate procedure. |
| BSPARAM | A BiocSingularParam class object from the BiocSingular package is used. Default is ExactParam(). |
| svd_k | If BSPARAM is set to RandomParam or IrlbaParam class from BiocSingular package, then svd_k will be used to used to reduce the computational cost of singular value decomposition. Default to 50. |

| | |
|---|---|
| dist | The distance metrics that are used in the calculation of the mutual nearest cluster, default is Pearson correlation. |
| WV | A optional vector indicating the wanted variation factor other than cell type info, such as cell stages. |
| WV_marker | An optional vector indicating the markers of the wanted variation. |
| BPPARAM | A BiocParallelParam class object from the BiocParallel package is used. Default is SerialParam(). |
| return_all_RUV | If FALSE, then only returns a SingleCellExperiment object with original data and one normalised matrix. Otherwise, the SingleCellExperiment object will contain the original data and one normalised matrix for each ruvK value. In this latter case, assay_name must have the same length as ruvK. |
| assay_name | The assay name(s) for the adjusted expression matrix(matrices). If return_all_RUV = TRUE assay_name must have the same length as ruvK. |
| plot_igraph | If TRUE, then during the un/semi-supervised scMerge, igraph plot will be displayed |
| verbose | If TRUE, then all intermediate steps will be shown. Default to FALSE. |

## Value

Returns a SingleCellExperiment object with following components:

- assays: the original assays and also the normalised matrix
- metadata: containing the ruvK vector, ruvK_optimal based on F-score, and the replicate matrix

## Author(s)

Yingxin Lin, Kevin Wang

## Examples

```
## Loading example data
data('example_sce', package = 'scMerge')
## Previously computed stably expressed genes
data('segList_ensemblGeneID', package = 'scMerge')
## Running an example data with minimal inputs
sce_mESC <- scMerge(sce_combine = example_sce,
ctl = segList_ensemblGeneID$mouse$mouse_scSEG,
kmeansK = c(3, 3),
assay_name = 'scMerge')

sce_mESC = scater::runPCA(sce_mESC, exprs_values = "logcounts")
scater::plotPCA(sce_mESC, colour_by = 'cellTypes', shape = 'batch')

sce_mESC = scater::runPCA(sce_mESC, exprs_values = 'scMerge')
scater::plotPCA(sce_mESC, colour_by = 'cellTypes', shape = 'batch')
```

scMerge2 *Perform the scMerge2 algorithm*

## Description

Merge single-cell data from different batches and experiments leveraging (pseudo)-replicates, control genes and pseudo-bulk.

## Usage

```
scMerge2(
  exprsMat,
  batch,
  cellTypes = NULL,
  condition = NULL,
  ctl = rownames(exprsMat),
  chosen.hvg = NULL,
  ruvK = 20,
  use_bpparam = BiocParallel::SerialParam(),
  use_bsparam = BiocSingular::RandomParam(),
  use_bnparam = BiocNeighbors::AnnoyParam(),
  pseudoBulk_fn = "create_pseudoBulk",
  k_pseudoBulk = 30,
  k_celltype = 10,
  exprsMat_counts = NULL,
  cosineNorm = TRUE,
  return_subset = FALSE,
  return_subset_genes = NULL,
  return_matrix = TRUE,
  byChunk = TRUE,
  chunkSize = 50000,
  verbose = TRUE,
  seed = 1
)
```

## Arguments

| | |
|---|---|
| exprsMat | A gene (row) by cell (column) log-transformed matrix to be adjusted. |
| batch | A vector indicating the batch information for each cell in the batch-combined matrix. |
| cellTypes | An optional vector indicating the cell type information for each cell in the batch-combined matrix. If it is NULL, pseudo-replicate procedure will be run to identify cell type. |
| condition | An optional vector indicating the condition information for each cell in the batch-combined matrix. |

| | |
|---|---|
| ctl | A character vector of negative control. It should have a non-empty intersection with the rows of exprsMat |
| chosen.hvg | An optional character vector of highly variable genes chosen. |
| ruvK | An integer indicates the number of unwanted variation factors that are removed, default is 20. |
| use_bpparam | A BiocParallelParam class object from the BiocParallel package is used. Default is SerialParam(). |
| use_bsparam | A BiocSingularParam class object from the BiocSingular package is used. Default is RandomParam(). |
| use_bnparam | A BiocNeighborsParam class object from the BiocNeighbors package is used. Default is AnnoyParam(). |
| pseudoBulk_fn | A character indicates the way of pseudobulk constructed. |
| k_pseudoBulk | An integer indicates the number of pseudobulk constructed within each cell grouping. Default is 30. |
| k_celltype | An integer indicates the number of nearest neighbours used in buildSNNGraph when grouping cells within each batch. Default is 10. |
| exprsMat_counts | A gene (row) by cell (column) counts matrix to be adjusted. |
| cosineNorm | A logical vector indicates whether cosine normalisation is performed on input data. |
| return_subset | If TRUE, adjusted matrix of only a subset of genes (hvg or indicates in return_subset_genes) will be return. |
| return_subset_genes | An optional character vector of indicates the subset of genes will be adjusted. |
| return_matrix | A logical value indicates whether the adjusted matrix is calculated and returned. |
| byChunk | A logical value indicates whether it calculates the adjusted matrix by chunk |
| chunkSize | A numeric indicates the size of the chunk. If FALSE, then only the estimated parameters will be returned. |
| verbose | If TRUE, then all intermediate steps will be shown. Default to FALSE. |
| seed | A numeric input indicates the seed used. |

## Value

Returns a list object with following components:

- newY: if return_matrix is TRUE, the adjusted matrix will be return.
- fullalpha: Alpha estimated from the fastRUVIII model.
- M: Replicate matrix.

## Author(s)

Yingxin Lin

## Examples

```
## Loading example data
data('example_sce', package = 'scMerge')
## Previously computed stably expressed genes
data('segList_ensemblGeneID', package = 'scMerge')
## Running an example data with minimal inputs
library(SingleCellExperiment)
exprsMat <- scMerge2(exprsMat = logcounts(example_sce),
batch = example_sce$batch,
ctl = segList_ensemblGeneID$mouse$mouse_scSEG)
assay(example_sce, "scMerge2") <- exprsMat$newY


example_sce = scater::runPCA(example_sce, exprs_values = 'scMerge2')
scater::plotPCA(example_sce, colour_by = 'cellTypes', shape = 'batch')
```

---

scMerge2h *Perform the scMerge2 (hierarchical) algorithm*

---

## Description

Merge single-cell data hierarchically from different batches and experiments leveraging (pseudo)-replicates, control genes and pseudo-bulk.

## Usage

```
scMerge2h(
  exprsMat,
  batch_list = list(),
  h_idx_list = list(),
  cellTypes = NULL,
  condition = NULL,
  ctl = rownames(exprsMat),
  chosen.hvg = NULL,
  ruvK_list = 20,
  use_bpparam = BiocParallel::SerialParam(),
  use_bsparam = BiocSingular::RandomParam(),
  use_bnparam = BiocNeighbors::AnnoyParam(),
  pseudoBulk_fn = "create_pseudoBulk",
  k_pseudoBulk = 30,
  k_celltype = 10,
  exprsMat_counts = NULL,
  cosineNorm = TRUE,
  return_subset = FALSE,
  return_subset_genes = NULL,
  return_matrix = TRUE,
  verbose = TRUE,
  seed = 1
)
```

## Arguments

| | |
|---|---|
| exprsMat | A gene (row) by cell (column) log-transformed matrix to be adjusted. |
| batch_list | A list indicating the batch information for each cell in the batch-combined matrix. |
| h_idx_list | A list indicating the indeces information in the hierarchical merging. |
| cellTypes | An optional vector indicating the cell type information for each cell in the batch-combined matrix. If it is NULL, pseudo-replicate procedure will be run to identify cell type. |
| condition | An optional vector indicating the condition information for each cell in the batch-combined matrix. |
| ctl | A character vector of negative control. It should have a non-empty intersection with the rows of exprsMat |
| chosen.hvg | An optional character vector of highly variable genes chosen. |
| ruvK_list | An integer indicates the number of unwanted variation factors that are removed, default is 20. |
| use_bpparam | A BiocParallelParam class object from the BiocParallel package is used. Default is SerialParam(). |
| use_bsparam | A BiocSingularParam class object from the BiocSingular package is used. Default is RandomParam(). |
| use_bnparam | A BiocNeighborsParam class object from the BiocNeighbors package is used. Default is AnnoyParam(). |
| pseudoBulk_fn | A character indicates the way of pseudobulk constructed. |
| k_pseudoBulk | An integer indicates the number of pseudobulk constructed within each cell grouping. Default is 30. |
| k_celltype | An integer indicates the number of nearest neighbours used in buildSNNGraph when grouping cells within each batch. Default is 10. |
| exprsMat_counts | A gene (row) by cell (column) counts matrix to be adjusted. |
| cosineNorm | A logical vector indicates whether cosine normalisation is performed on input data. |
| return_subset | If TRUE, adjusted matrix of only a subset of genes (hvg or indicates in return_subset_genes) will be return. |
| return_subset_genes | An optional character vector of indicates the subset of genes will be adjusted. |
| return_matrix | A logical vector indicates whether the adjusted matrix is calculated and returned. If FALSE, then only the estimated parameters will be returned. |
| verbose | If TRUE, then all intermediate steps will be shown. Default to FALSE. |
| seed | A numeric input indicates the seed used. |

## Author(s)

Yingxin Lin

## Examples

```
## Loading example data
data('example_sce', package = 'scMerge')
## Previously computed stably expressed genes
data('segList_ensemblGeneID', package = 'scMerge')


# Create a fake sample information
example_sce$sample <- rep(c(1:4), each = 50)

# Construct a hierarchical index list
h_idx_list <- list(level1 = split(1:200, example_sce$batch),
                   level2 = list(1:200))

# Construct a batch information list
batch_list <- list(level1 = split(example_sce$sample, example_sce$batch),
                   level2 = list(example_sce$batch))
library(SingleCellExperiment)
exprsMat <- scMerge2h(exprsMat = logcounts(example_sce),
batch_list = batch_list,
h_idx_list = h_idx_list,
ctl = segList_ensemblGeneID$mouse$mouse_scSEG,
ruvK_list = c(2, 5))
assay(example_sce, "scMerge2") <- exprsMat[[length(h_idx_list)]]
example_sce = scater::runPCA(example_sce, exprs_values = 'scMerge2')
scater::plotPCA(example_sce, colour_by = 'cellTypes', shape = 'batch')
```

---

# scReplicate *Create replicate matrix for scMerge algorithm*

---

## Description

Create replicate matrix for scMerge algorithm using un-/semi-/supervised approaches.

## Usage

```
scReplicate(
  sce_combine,
  batch = NULL,
  kmeansK = NULL,
  exprs = "logcounts",
  hvg_exprs = "counts",
  marker = NULL,
  marker_list = NULL,
  replicate_prop = 1,
  cell_type = NULL,
  cell_type_match = FALSE,
  cell_type_inc = NULL,
```

```
  dist = "cor",
  WV = NULL,
  WV_marker = NULL,
  BPPARAM = SerialParam(),
  return_all = FALSE,
  BSPARAM = ExactParam(),
  plot_igraph = TRUE,
  verbose = FALSE
)
```

## Arguments

| | |
|---|---|
| sce_combine | A SingleCellExperiment object contains the batch-combined matrix with batch info in colData |
| batch | A vector indicates the batch information for each cell in the batch-combined matrix. |
| kmeansK | A vector indicates the kmeans's K for each batch, length of kmeansK needs to be the same as the number of batch. |
| exprs | A string indicates the assay that are used for batch correction, default is log-counts |
| hvg_exprs | A string indicates the assay that are used for highly variable genes identification, default is counts |
| marker | A vector of markers, which will be used in calculation of mutual nearest cluster. If no markers input, highly variable genes will be used instead |
| marker_list | A list of markers for each batch, which will be used in calculation of mutual nearest cluster. |
| replicate_prop | A number indicating the ratio of cells that are included in pseudo-replicates, ranges from 0 to 1. Default to 1. |
| cell_type | A vector indicates the cell type information for each cell in the batch-combined matrix. If it is NULL, pseudo-replicate procedure will be run to identify cell type. |
| cell_type_match | Whether find mutual nearest cluster using cell type information |
| cell_type_inc | A vector indicates the indices of the cells that will be used to supervise the pseudo-replicate procedure |
| dist | The distance metrics that are used in the calculation of the mutual nearest cluster, default is Pearson correlation. |
| WV | A vector indicates the wanted variation factor other than cell type info, such as cell stages. |
| WV_marker | A vector indicates the markers of the wanted variation. |
| BPPARAM | A BiocParallelParam class object from the BiocParallel package is used. Default is SerialParam(). |
| return_all | If FALSE, only return the replicate matrix. |
| BSPARAM | A BiocSingularParam class object from the BiocSingular package is used. Default is ExactParam(). |

| | |
|---|---|
| plot_igraph | If TRUE, then during the un/semi-supervised scMErge, igraph plot will be displayed |
| verbose | If TRUE, then all intermediate steps will be shown. Default to FALSE. |

## Value

If return_all is FALSE, return a replicate matrix. If return_sce is TRUE, return the followings

| | |
|---|---|
| repMat | replicate matrix |
| mnc | mutual nearest cluster |
| replicate vector | replicate vector |
| HVG | highly variable genes used in scReplicate |

A cell-replicates mapping matrix. Each row correspond to a cell from the input expression matrix, and each column correspond to a cell-cluster/cell-type. An element of the mapping matrix is 1 if the scReplicate algorithm determines that this cell should belong to that cell cluster and 0 otherwise.

## Author(s)

Yingxin Lin, Kevin Wang

## Examples

```
## Loading example data
set.seed(1)
data('example_sce', package = 'scMerge')
scRep_result = scReplicate(
  sce_combine = example_sce,
  batch = example_sce$batch,
  kmeansK = c(3,3))
```

---

# scRUVg *RUVg function for single cell (under development)*

---

## Description

Modified based on RUV2 from package ruv and RUVg from package RUVSeq function (see these function's documentations for full documentations and usage)

## Usage

```
scRUVg(
  Y,
  ctl,
  k,
  Z = 1,
  eta = NULL,
  include.intercept = TRUE,
  fullW = NULL,
  svdyc = NULL
)
```

## Arguments

| | |
|---|---|
| Y | The data. A m by n matrix, where m is the number of observations and n is the number of features. |
| ctl | index vector to specify the negative controls. |
| k | The number of unwanted factors to use. |
| Z | Any additional covariates to include in the model. |
| eta | Gene-wise (as opposed to sample-wise) covariates. |
| include.intercept | Applies to both Z and eta. When Z or eta (or both) is specified (not NULL) but does not already include an intercept term, this will automatically include one. If only one of Z or eta should include an intercept, this variable should be set to FALSE, and the intercept term should be included manually where desired. |
| fullW | Can be included to speed up execution. Is returned by previous calls of scRUVg |
| svdyc | Can be included to speed up execution. For internal use; please use fullW instead. |

## Value

A list consists of:

- A matrix newY, the normalised matrix,
- A matrix W, the unwanted variation matrix, and ;
- A matrix alpha, this corresponding coefficient matrix for W.

## Author(s)

Yingxin Lin, Kevin Wang

## Examples

```
L = scMerge::ruvSimulate(m = 80, n = 1000, nc = 50, nCelltypes = 10)
Y = L$Y; ctl = L$ctl
ruvgRes = scMerge::scRUVg(Y = Y, ctl = ctl, k = 20)
```

scRUVIII *scRUVIII: RUVIII algorithm optimised for single cell data*

## Description

A function to perform location/scale adjustment to data as the input of RUVIII which also provides the option to select optimal RUVk according to the silhouette coefficient

## Usage

```
scRUVIII(
  Y = Y,
  M = M,
  ctl = ctl,
  fullalpha = NULL,
  k = k,
  cell_type = NULL,
  batch = NULL,
  return_all_RUV = TRUE,
  BPPARAM = SerialParam(),
  BSPARAM = ExactParam(),
  svd_k = 50
)
```

## Arguments

| | |
|---|---|
| Y | The unnormalised SC data. A m by n matrix, where m is the number of observations and n is the number of features. |
| M | The replicate mapping matrix. The mapping matrix has m rows (one for each observation), and each column represents a set of replicates. The (i, j)-th entry of the mapping matrix is 1 if the i-th observation is in replicate set j, and 0 otherwise. See ruv::RUVIII for more details. |
| ctl | An index vector to specify the negative controls. Either a logical vector of length n or a vector of integers. |
| fullalpha | Not used. Please ignore. |
| k | The number of unwanted factors to remove. This is inherited from the ruvK argument from the scMerge::scMerge function. |
| cell_type | An optional vector indicating the cell type information for each cell in the batch-combined matrix. If it is NULL, pseudo-replicate procedure will be run to identify cell type. |
| batch | Batch information inherited from the scMerge::scMerge function. |
| return_all_RUV | Whether to return extra information on the RUV function, inherited from the scMerge::scMerge function |
| BPPARAM | A BiocParallelParam class object from the BiocParallel package is used. Default is SerialParam(). |

| | |
|---|---|
| BSPARAM | A BiocSingularParam class object from the BiocSingular package is used. Default is ExactParam(). |
| svd_k | If BSPARAM is set to RandomParam or IrlbaParam class from BiocSingular package, then svd_k will be used to used to reduce the computational cost of singular value decomposition. Default to 50. |

## Value

A list consists of:

- RUV-normalised matrices: If k has multiple values, then the RUV-normalised matrices using all the supplied k values will be returned.
- optimal_ruvK: The optimal RUV k value as determined by silhouette coefficient.

## Author(s)

Yingxin Lin, Kevin Wang

## Examples

```
L = ruvSimulate(m = 200, n = 1000, nc = 100, nCelltypes = 3, nBatch = 2, lambda = 0.1, sce = FALSE)
Y = t(log2(L$Y + 1L)); M = L$M; ctl = L$ctl; batch = L$batch;
res = scRUVIII(Y = Y, M = M, ctl = ctl, k = c(5, 10, 15, 20), batch = batch)
```

---

# scSEGIndex *Single Cell Stably Express Gene Index*

---

## Description

This function computes the single-cell Stably Expressed Gene (scSEG) index from Lin. et al. (2019) for a given single-cell count data matrix. Each gene in the data is fitted with a gamma-normal mixture model and the final SEG index is computed as an average of key parameters that measure the expression stability of a gene.

We recommend using either the pre-computed genes (see "See Also" below) or the top SEG genes from an user's own data as the control genes in the scMerge function (see the ctl argument in the scMerge function).

## Usage

```
scSEGIndex(
  exprs_mat,
  cell_type = NULL,
  BPPARAM = SerialParam(progressbar = TRUE),
  return_all = FALSE
)
```

## Arguments

| | |
|---|---|
| exprs_mat | A log-transformed single-cell data, assumed to have no batch effect and covered a wide range of cell types. A n by m matrix, where n is the number of genes and m is the number of cells. |
| cell_type | A vector indicating the cell type information for each cell in the gene expression matrix. If it is NULL, the function calculates the scSEG index without using F-statistics. |
| BPPARAM | A BiocParallelParam class object from the BiocParallel package is used. Default is SerialParam(progressbar = TRUE). |
| return_all | Default to FALSE. If set to TRUE, then all genes will be returned. Otherwise, only genes with less than 80 percent zeroes will be returned. |

## Value

Returns a data frame. Each row is a gene and each column is a statistic relating to the stability of expression of each gene. The main statistic is the segIdx column, which is the SEG index.

## Author(s)

Shila Ghazanfar, Yingxin Lin, Pengyi Yang

## References

Evaluating stably expressed genes in single cells (2019). doi:10.1093/gigascience/giz106.

## See Also

Download human SEG directly from this link; Download mouse SEG directly from this link.

## Examples

```
## Loading example data
data('example_sce', package = 'scMerge')
## subsetting genes to illustrate usage.
exprs_mat = SummarizedExperiment::assay(example_sce, 'logcounts')[1:110, 1:20]
set.seed(1)
result1 = scSEGIndex(exprs_mat = exprs_mat)
## If parallelisation is needed:
param = BiocParallel::MulticoreParam(workers = 2, progressbar = TRUE)
result2 = scSEGIndex(exprs_mat = exprs_mat, BPPARAM = param)
## Closing the parallelisation
BiocParallel::register(BPPARAM = BiocParallel::SerialParam())
```

## segList *Stably expressed gene list in official gene symbols for both human and mouse*

### Description

A list includes the stably expressed genes for both human and mouse

### Usage

```
data(segList, package = 'scMerge')
```

### Format

An object of class `list` of length 2.

## segList_ensemblGeneID *Stably expressed gene list in EnsemblGeneID for both human and mouse*

### Description

A list includes the stably expressed genes for both human and mouse

### Usage

```
data(segList_ensemblGeneID, package = 'scMerge')
```

### Format

An object of class `list` of length 2.

# Index