Package ‘scMerge’

October 16, 2019

Type Package

Title scMerge: Merging multiple batches of scRNA-seq data

Version 1.0.0

Description Like all gene expression data, single-cell RNA-seq (scRNA-Seq) 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 scRNA-Seq data.
This package contains all the necessary functions in the
scMerge pipeline, including the identification of SEGs, replication-identification methods, and
merging of scRNA-Seq data.

License GPL-3

Encoding UTF-8

LazyData false

Depends R (>= 3.6.0)

Imports BiocParallel, cluster, distr, doSNOW, foreach, igraph, irlba,
iterators, matrixStats, M3Drop (>= 1.9.4), parallel, pdist,
proxy, Rcpp (>= 0.12.18), RcppEigen (>= 0.3.3.4.0), ruv, rsvd,
S4Vectors, SingleCellExperiment, SummarizedExperiment

LinkingTo Rcpp (>= 0.12.18), RcppEigen, testthat

RoxygenNote 6.1.1

Suggests BiocStyle, covr, knitr, Matrix, rmarkdown, scales, scater,
testthat

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_9

git_last_commit 74ec5e4

git_last_commit_date 2019-05-02

Date/Publication 2019-10-15

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

Maintainer Kevin Wang <kevin.wang@sydney.edu.au>

R topics documented:

eigenMatMult *Fast matrix multiplication using RcppEigen*

Description

Fast matrix multiplication using RcppEigen

Usage

```
eigenMatMult(A, B)
```

Arguments

A	a matrix
B	a matrix

Value

The matrix product of A times B

Examples

```
A = matrix(0, ncol = 500, nrow = 500)
system.time(A %*% A)
system.time(eigenMatMult(A, A))
```

`eigenResidop` *fast matrix residual operator using RcppEigen*

Description

fast matrix residual operator using RcppEigen

Usage

```
eigenResidop(A, B)
```

Arguments

`A`	a matrix
`B`	a matrix

Value

The matrix product of

$$A - B(B^tB)^{-1}B^tA$$

Examples

```
Y = M = diag(1, 500)
system.time(scMerge::eigenResidop(Y, M))
system.time(ruv::residop(Y, M))
```

`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’)

Usage

```
example_sce
```

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, fast_svd = FALSE,
  rsvd_prop = 0.1, include.intercept = TRUE, average = FALSE,
  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.

`fast_svd` If `TRUE`, fast algorithms will be used for singular value decomposition calculation via the `irlba` and `rsvd` packages. We recommend using this option when the number of cells is large (e.g. more than 1000 cells).

`rsvd_prop` If `fast_svd` = `TRUE`, then `rsvd_prop` will be used to used to reduce the computational cost of randomised singular value decomposition. We recommend setting this number to less than 0.25 to achieve a balance between numerical accuracy and computational costs.

`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.

`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, fast_svd = FALSE)
improved2 = scMerge::fastRUVIII(Y = Y, M = M, ctl = ctl, k = 20, fast_svd = TRUE, rsvd_prop = 0.1)
old = ruv::RUVIII(Y = Y, M = M, ctl = ctl, k = 20)
all.equal(improved1, old)
all.equal(improved2, old)
```

---

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')

scater::plotPCA(L, colour_by = 'cellTypes', shape = 'batch',
                run_args = list(exprs_values = 'logcounts'))

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

---

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 = NULL, 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`
`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

```
library(SingleCellExperiment)
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", marker = NULL, marker_list = NULL,
  ruvK = 20, replicate_prop = 0.5, cell_type = NULL,
  cell_type_match = FALSE, cell_type_inc = NULL, fast_svd = FALSE,
  rsvd_prop = 0.1, dist = "cor", WV = NULL, WV_marker = NULL,
  parallel = FALSE, parallelParam = NULL, return_all_RUV = FALSE,
  assay_name = NULL, 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.
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.

`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.

`fast_svd` If `TRUE`, fast algorithms will be used for singular value decomposition calculation via the `irlba` and `rsvd` packages. We recommend using this option when the number of cells is large (e.g. more than 1000 cells).

`rsvd_prop` If `fast_svd = TRUE`, then `rsvd_prop` will be used to used to reduce the computational cost of randomised singular value decomposition. We recommend setting this number to less than 0.25 to achieve a balance between numerical accuracy and computational costs.

`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.

`parallel` If `TRUE`, then `BiocParallel` package will be used to perform parallelised computations.

`parallelParam` The `BiocParallelParam` class from the `BiocParallel` package is used. Default is bpparam().

`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.

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

`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, parallelParam = SerialParam(),
  return_all = FALSE, fast_svd, 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 logcounts

`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 indicates the ratio of cells that are included in pseudo-replicates, ranges from 0 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.

parallelParam — The BiocParallelParam class from the BiocParallel package is used. Default is SerialParam().

return_all — If FALSE, only return the replicate matrix.

fast_svd — If TRUE, fast algorithms will be used for singular value decomposition calculation via the irlba and rsvd packages. We recommend using this option when the number of cells is large (e.g. more than 1000 cells).

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),
  fast_svd = FALSE)
```

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,
  fast_svd = FALSE, rsvd_prop = 0.1)
```

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
`fast_svd`	If `TRUE`, fast algorithms will be used for singular value decomposition calculation via the `irlba` and `rsvd` packages. We recommend using this option when the number of cells is large (e.g. more than 1000 cells).
`rsvd_prop`	If `fast_svd = TRUE`, then `rsvd_prop` will be used to used to reduce the computational cost of randomised singular value decomposition. We recommend setting this number to less than 0.25 to achieve a balance between numerical accuracy and computational costs. If a lower value is used on a lower dimensional data (say < 1000 cell) will potentially yield a less accurate computed result but with a gain in speed. The default of 0.1 tends to achieve a balance between speed and accuracy.

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 = 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 *scSEGIndex*

Description

Calculate single-cell Stably Expressed Gene (scSEG) index from Lin. et. al. (2018).

Usage

```
scSEGIndex(exprsMat, cell_type = NULL, ncore = 1)
```

Arguments

exprsMat	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.
ncore	Number of cores that are used in parallel

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

https://www.biorxiv.org/content/10.1101/229815v2

Examples

```
## Loading example data
data('example_sce', package = 'scMerge')
## subsetting genes to illustrate usage.
exprsMat = SummarizedExperiment::assay(example_sce, 'counts')[1:110, 1:20]
set.seed(1)
result = scSEGIndex(exprsMat = exprsMat)
head(result)
```

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