Package ‘BisqueRNA’

March 19, 2020

Title  Decomposition of Bulk Expression with Single-Cell Sequencing

Version  1.0.2


Depends  R (>= 3.5.0)

License  GPL-3

Encoding  UTF-8

LazyData  true

RoxygenNote  6.1.1

Imports  Biobase, lsei, methods, stats

Suggests  Seurat, plyr, knitr, rmarkdown, testthat

URL  https://www.biorxiv.org/content/10.1101/669911v1

BugReports  https://github.com/cozygene/bisque/issues

VignetteBuilder  knitr

NeedsCompilation  no

Author  Brandon Jew [aut, cre],
        Marcus Alvarez [aut]

Maintainer  Brandon Jew <brandon.jew@ucla.edu>

Repository  CRAN

Date/Publication  2020-03-19 05:20:11 UTC
CalculateSCCellProportions

*R topics documented:*

CalculateSCCellProportions .................................................. 2
CorTri .................................................................................. 3
CountsToCPM ........................................................................... 3
EstimatePCACellTypeProportions ............................................. 4
FilterUnexpressedGenes ............................................................. 4
FilterZeroVarianceGenes ............................................................. 5
GenerateSCReference ............................................................... 5
GetCTP .................................................................................. 6
GetNumGenes ............................................................................ 7
GetNumGenesWeighted ............................................................. 7
GetOverlappingGenes ............................................................... 8
GetOverlappingSamples ............................................................ 8
GetUniqueMarkers ................................................................. 9
MarkerBasedDecomposition ....................................................... 9
ReferenceBasedDecomposition .................................................. 11
SemisupervisedTransformBulk ............................................... 12
SeuratToExpressionSet ............................................................ 13
SimulateBarcode ..................................................................... 14
SimulateData .......................................................................... 14
SupervisedTransformBulk ........................................................ 15

**Index**

CalculateSCCellProportions

*Calculate cell proportions based on single-cell data*

**Description**

Returns proportion of each cell type out of total cells for each individual in the single-cell Expression Set

**Usage**

CalculateSCCellProportions(sc.eset, subject.names, cell.types)

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sc.eset</td>
<td>Expression Set with single-cell data</td>
</tr>
<tr>
<td>subject.names</td>
<td>A character string. Name of phenoData attribute in sc.eset that indicates individual ID.</td>
</tr>
<tr>
<td>cell.types</td>
<td>A character string. Name of phenoData attribute in sc.eset that indicates cell type</td>
</tr>
</tbody>
</table>
CorTri

Value

sc.props Matrix. Cell proportions with number of cell types rows by number of individuals columns

Description

This function runs correlation between markers of a data frame or matrix, returning the values of the lower/upper triangular of the correlation matrix in a vector.

Usage

CorTri(x, method = "pearson")

Arguments

x Data frame or matrix. Column vectors are correlated
method Character string. Name of method passed to cor. Pearson by default.

Value

cors Numeric vector. Correlation coefficients of pairs

CountsToCPM

Convert counts data in Expression Set to counts per million (CPM)

Description

Convert counts data in Expression Set to counts per million (CPM)

Usage

CountsToCPM(eset)

Arguments

eset Expression Set containing counts assay data.

Value

eset Expression Set containing CPM assay data
EstimatePCACellTypeProportions

*Estimate cell type proportions using first PC of expression matrix*

**Description**

Estimate cell type proportions using first PC of expression matrix

**Usage**

EstimatePCACellTypeProportions(x, weighted = FALSE, w = NULL)

**Arguments**

- **x**: A sample by gene bulk expression matrix. Genes should be marker genes
- **weighted**: Boolean. If weighted=TRUE, multiply scaled gene expression by gene weights
- **w**: Numeric vector. Weights of genes

**Value**

ret List. Attribute `pcs` contains matrix of PCs, where PC1 should be used as estimates for cell type abundances. Attribute `sdev` contains eigenvalues of eigendecomposition of var-covar matrix. The 1st eigenvalue should explain most of the variance. Attribute `genes` contains names of genes.

FilterUnexpressedGenes

*Remove genes in Expression Set with zero expression in all samples*

**Description**

Remove genes in Expression Set with zero expression in all samples

**Usage**

FilterUnexpressedGenes(eset, verbose = TRUE)

**Arguments**

- **eset**: Expression Set
- **verbose**: Boolean. Print logging info

**Value**

eset Expression Set with zero expression genes removed
FilterZeroVarianceGenes

Remove genes in Expression Set with zero variance across samples

Description

Remove genes in Expression Set with zero variance across samples

Usage

FilterZeroVarianceGenes(eset, verbose = TRUE)

Arguments

eset       Expression Set
verbose    Boolean. Print logging info

Value

eset Expression Set with zero variance genes removed

GenerateSCReference

Generate reference profile for cell types identified in single-cell data

Description

Averages expression within each cell type across all samples to use as reference profile.

Usage

GenerateSCReference(sc.eset, cell.types)

Arguments

sc.eset       Expression Set with single-cell data
cell.types    A character string. Name of phenoData attribute in sc.eset that indicates cell type

Value

sc.ref Matrix. Reference profile with number of gene rows by number of cell types columns.
GetCTP

Return cell type proportions from bulk

Description

Calculate cell type proportions from a data frame containing bulk expression values. Uses PCA (weighted or regular) to estimate relative proportions within each cell type.

Usage

GetCTP(bulk, cell_types, markers, ct_col, gene_col, min_gene, max_gene, weighted, w_col, verbose)

Arguments

bulk 
Expression Set containing bulk data

cell_types 
Character vector. Names of cell types.

markers 
Data frame with columns specifying cluster and gene, and optionally a column for weights, typically the fold-change of the gene. Important that the genes for each cell type are row-sorted by significance.

ct_col 
Character string. Column name specifying cluster/cell type corresponding to each marker gene in markers.

gene_col 
Character string. Column name specifying gene names in markers.

min_gene 
Numeric. Min number of genes to use for each cell type.

max_gene 
Numeric. Max number of genes to use for each cell type.

weighted 
Boolean. Whether to use weights for gene prioritization

w_col 
Character string. Column name for weights, such as "avg_logFC", in markers

verbose 
Boolean. Whether to print log info during decomposition. Errors will be printed regardless.

Value

A List. Slot cors contains list of vectors with correlation coefficients. Slot ctps contains list of CTP objects returned by GetCTP
GetNumGenes

Get number of genes to use with no weighted information

**Description**

Get number of genes to use with no weighted information

**Usage**

GetNumGenes(x, min.gene = 25, max.gene = 200)

**Arguments**

- **x**: Numeric Matrix. A sample by gene expression matrix containing the marker genes.
- **min.gene**: Numeric. Minimum number of genes to consider as markers.
- **max.gene**: Numeric. Maximum number of genes to consider as markers.

**Value**

best.n Numeric. Number of genes to use

---

GetNumGenesWeighted

Get number of genes to use with weighted PCA

**Description**

Get number of genes to use with weighted PCA

**Usage**

GetNumGenesWeighted(x, w, min.gene = 25, max.gene = 200)

**Arguments**

- **x**: Numeric Matrix. A sample by gene expression matrix containing the marker genes.
- **w**: Numeric Vector. The weights of the genes that correspond to the columns of x.
- **min.gene**: Numeric. Minimum number of genes to consider as markers.
- **max.gene**: Numeric. Maximum number of genes to consider as markers.

**Value**

best.n Numeric. Number of genes to use
GetOverlappingGenes  
Find overlapping genes in single-cell data, bulk data, and marker genes

Description
Find overlapping genes in single-cell data, bulk data, and marker genes

Usage
GetOverlappingGenes(sc.eset, bulk.eset, markers, verbose)

Arguments
- **sc.eset**  
  Expression Set with single-cell data
- **bulk.eset**  
  Expression Set with bulk data
- **markers**  
  Character vector. List of relevant marker genes
- **verbose**  
  Boolean. Print logging info

Value
overlapping.genes Character vector. List of genes found in markers and both datasets.

GetOverlappingSamples  
Find overlapping samples in single-cell and bulk data

Description
Find overlapping samples in single-cell and bulk data

Usage
GetOverlappingSamples(sc.eset, bulk.eset, subject.names, verbose)

Arguments
- **sc.eset**  
  Expression Set with single-cell data
- **bulk.eset**  
  Expression Set with bulk data
- **subject.names**  
  A character string. Name of phenoData attribute in sc.eset that indicates individual ID (that would be found in bulk.eset if overlapping)
- **verbose**  
  Boolean. Print logging info

Value
samples A list with attributes **overlapping** and **remaining**. Each attribute refers to a character vector that lists the samples found in both datasets and samples found only in bulk, respectively.
GetUniqueMarkers

Get unique markers present in only 1 cell type

Description

Given a data frame of marker genes for cell types, returns a new data frame with non-unique markers removed.

Usage

GetUniqueMarkers(x, gene_col = "gene")

Arguments

x
Data frame. Contains column with marker gene names

gene_col
Character string. Name of the column that contains the marker genes

Value

x Data frame. Markers with non-unique markers removed

MarkerBasedDecomposition

Performs marker-based decomposition of bulk expression using marker genes

Description

Estimates relative abundances of cell types from PCA-based decomposition. Uses a list of marker genes to subset the expression data, and returns the first PC of each sub-matrix as the cell type fraction estimates. Optionally, weights for each marker gene can be used to prioritize genes that are highly expressed in the given cell type.

Usage

MarkerBasedDecomposition(bulk.eset, markers, ct_col = "cluster", gene_col = "gene", min_gene = 5, max_gene = 200, weighted = FALSE, w_col = "avg_logFC", unique_markers = TRUE, verbose = TRUE)
Argument

bulk.eset  Expression Set. Normalized bulk expression data.
markers  Data frame with columns specifying cluster and gene, and optionally a column for weights, typically the fold-change of the gene. Important that the genes for each cell type are row-sorted by significance.
ct_col  Character string. Column name specifying cluster/cell type corresponding to each marker gene in markers.
gene_col  Character string. Column name specifying gene names in markers.
min_gene  Numeric. Min number of genes to use for each cell type.
max_gene  Numeric. Max number of genes to use for each cell type.
weighted  Boolean. Whether to use weights for gene prioritization
w_col  Character string. Column name for weights, such as "avg_logFC", in markers
unique_markers  Boolean. If TRUE, subset markers to include only genes that are markers for only one cell type
verbose  Boolean. Whether to print log info during decomposition. Errors will be printed regardless.

Details

Note that this method expects the input bulk data to be normalized, unlike the reference-based method.

Value

A List. Slot bulk.props contains estimated relative cell type abundances. Slot var.explained contains variance explained by first 20 PCs for cell type marker genes. Slot genes.used contains vector of genes used for decomposition.

Examples

library(Biobase)
sim.data <- SimulateData(n.ind=10, n.genes=100, n.cells=100,
cell.types=c("Neurons", "Astrocytes", "Microglia"),
avg.props=c(.5, .3, .2))
res <- MarkerBasedDecomposition(sim.data$bulk.eset, sim.data$markers, weighted=FALSE)
estimated.cell.proportions <- res$bulk.props
ReferenceBasedDecomposition

Performs reference-based decomposition of bulk expression using single-cell data

Description

Generates a reference profile based on single-cell data. Learns a transformation of bulk expression based on observed single-cell proportions and performs NNLS regression on these transformed values to estimate cell proportions.

Usage

ReferenceBasedDecomposition(bulk.eset, sc.eset, markers = NULL, cell.types = "cellType", subject.names = "SubjectName", use.overlap = TRUE, verbose = TRUE)

Arguments

bulk.eset  Expression Set containin bulk data. No PhenoData required but if overlapping option used, IDs returned by sampleNames(bulk.eset) should match those found in sc.eset phenoData individual labels.

sc.eset   Expression Set containing single-cell data. PhenoData of this Expression Set should contain cell type and individual labels for each cell. Names of these fields specified by arguments below.

markers   Structure, such as character vector, containing marker genes to be used in decomposition. ‘base::unique(base::unlist(markers))’ should return a simple vector containing each gene name. If no argument or NULL provided, the method will use all available genes for decomposition.

cell.types Character string. Name of phenoData attribute in sc.eset indicating cell type label for each cell

subject.names Character string. Name of phenoData attribute in sc.eset indicating individual label for each cell

use.overlap Boolean. Whether to use and expect overlapping samples in decomposition.

verbose Boolean. Whether to print log info during decomposition. Errors will be printed regardless.

Details

Expects read counts for both datasets, as they will be converted to counts per million (CPM). Two options available: Use overlapping individuals found in both single-cell and bulk datasets to learn transformation or learn transformation from single-cell alone. The overlapping option is expected to have better performance.
SemisupervisedTransformBulk

Description

For a specific gene, this function learns a transformation of the bulk expression to match the distribution produced by the single-cell based reference and observed single-cell based cell proportions.

Usage

SemisupervisedTransformBulk(gene, Y.train, X.pred)

Arguments

gene

Character string. Gene name that corresponds to row in Y.train

Y.train

Numeric Matrix. Number of gene rows by number of overlapping individuals columns. Contains weighted sum of reference profile by single-cell based cell proportion estimates for each individual

X.pred

Numeric Matrix. Number of gene rows by number of remaining individuals columns. Contains observed bulk expression for each individual to be transformed.

Value

Y.pred Numeric Matrix. One row for given gene by number of remaining individuals columns. Contains transformed bulk expression for each individual.
Description

`SeuratToExpressionSet()` returns an Expression Set with phenotype data indicating cell type (cell-Type) and individual (SubjectName) for each cell in a Seurat object. Raw counts data is used for assay data.

Usage

```r
SeuratToExpressionSet(seurat.object, delimiter, position, version = c("v2", "v3"))
```

Arguments

- `seurat.object`: Seurat object with attributes `raw.data`, `ident`, and `cell.names`
- `delimiter`: Character to split cell names with to find individual ID.
- `position`: Integer indicating 1-indexed position of individual ID after splitting cell name with `delimiter`.
- `version`: Character string. Either "v2" or "v3". Seurat version used to create Seurat object.

Details

Note that the Seurat and Biobase libraries should be attached before running this function. The `delimiter` and `position` arguments are used to infer the individual ID from the cell ID. For example, a delimiter of ";" and position of "2" indicates that the individual ID for the cell ID `ACTG-2` would be `2`.

Value

`sc.eset` Expression set containing relevant phenotype and individual data, `cellType` and `SubjectName`.

Examples

```r
library(Seurat)
library(Biobase)

# We make a class to emulate a Seurat v2 object for illustration only
setClass("testSeuratv2", representation(cell.names = "character",
                                        ident = "character",
                                        raw.data = "matrix"))

sc.counts <- matrix(0,nrow=3,ncol=3)
# These barcodes correspond to a delimiter of "-" and position 2 for individual id.
test.cell.names <- c("ATCG-1", "TAGC-2", "GTCA-3")
test.ident <- c("cell type a", "cell type b", "cell type c")
```
SimulateBarcode

**Simulate barcode for decomposition illustration**

**Description**
Generates a nucleotide barcode similar to those generated by 10x chromium sequencing platforms for illustration purposes. Generates barcode and individual ID separated by `-` delimiter.

**Usage**
SimulateBarcode(index, individual, barcode.length)

**Arguments**
- **index** : Integer. Index of cell ID from 0 to barcode.length to the fourth power. Will generate a unique nucleotide barcode for each index.
- **individual** : Character. ID of individual that the cell is from.
- **barcode.length** : Integer. Length of nucleotide barcode.

**Value**
Simulated barcode for cell from an individual

SimulateData

**Simulate data for decomposition illustration**

**Description**
Simulates bulk and single-cell expression, as well as marker genes and true proportions that can be used as an example of decomposition.

**Usage**
SimulateData(n.ind, n.genes, n.cells, cell.types, avg.props)
**Arguments**

- **n.ind**: Integer. Number of individuals to simulate
- **n.genes**: Integer. Number of genes to simulate
- **n.cells**: Integer. Number of cells per individual for single-cell data
- **cell.types**: Character vector. List of cell types to simulate
- **avg.props**: Numeric vector. List of average proportions for given cell types. Should be same length as cell.types and sum to 1

**Value**

A list with simulated single-cell in slot 'sc.eset' and bulk in 'bulk.eset', as well as true proportions in 'props' and marker genes in 'markers'.

**Examples**

```r
library(Biobase)
sim.data <- SimulateData(n.ind=10, n.genes=100, n.cells=100,
cell.types=c("Neurons", "Astrocytes", "Microglia"),
avg.props=c(.5, .3, .2))
```

---

**SupervisedTransformBulk**

*Transforms bulk expression of a gene given overlapping data*

**Description**

For a specific gene, this function uses linear regression to learn a transformation of the bulk expression to match the values produced by the single-cell based reference and observed single-cell based cell proportions.

**Usage**

`SupervisedTransformBulk(gene, Y.train, X.train, X.pred)`

**Arguments**

- **gene**: Character string. Gene name that corresponds to row in Y.train
- **Y.train**: Numeric Matrix. Number of gene rows by number of overlapping individuals columns. Contains weighted sum of reference profile by single-cell based cell proportion estimates for each individual
- **X.train**: Numeric Matrix. Number of gene rows by number of overlapping individuals columns. Contains observed bulk expression for each individual
- **X.pred**: Numeric Matrix. Number of gene rows by number of remaining individuals columns. Contains observed bulk expression for each individual to be transformed.
Details

If a linear transformation cannot be learned for a gene (zero variance in observed bulk or single-cell based weighted sums), a vector of NaNs will be returned of the expected length (length of X.pred)

Value

Y.pred Numeric Matrix. One row for given gene by number of remaining individuals columns. Contains transformed bulk expression for each individual.
Index

CalculateSCCellProportions, 2
CorTri, 3
CountsToCPM, 3
EstimatePCACellTypeProportions, 4
FilterUnexpressedGenes, 4
FilterZeroVarianceGenes, 5
GenerateSCReference, 5
GetCTP, 6
GetNumGenes, 7
GetNumGenesWeighted, 7
GetOverlappingGenes, 8
GetOverlappingSamples, 8
GetUniqueMarkers, 9
MarkerBasedDecomposition, 9
ReferenceBasedDecomposition, 11
SemisupervisedTransformBulk, 12
SeuratToExpressionSet, 13
SimulateBarcode, 14
SimulateData, 14
SupervisedTransformBulk, 15