

Package ‘scDbfFinder’

April 15, 2020

Type Package

Title scDbfFinder

Version 1.1.8

Depends R (>= 3.6)

URL <https://github.com/plger/scDbfFinder>

BugReports <https://github.com/plger/scDbfFinder/issues>

Description Efficient identification of doublets in single-cell RNAseq directly from counts using overclustering-based generation of artificial doublets.

License GPL-3

Imports igraph, Matrix, matrixStats, BiocParallel, BiocNeighbors, SummarizedExperiment, SingleCellExperiment, scran, scater, data.table, dplyr, ggplot2, randomForest, graphics, methods, stats, DelayedArray

Suggests BiocStyle, knitr, rmarkdown

VignetteBuilder knitr

Encoding UTF-8

RoxygenNote 6.1.1

biocViews Preprocessing, SingleCell, RNASeq

git_url <https://git.bioconductor.org/packages/scDbfFinder>

git_branch RELEASE_3_10

git_last_commit 78509bb

git_last_commit_date 2020-03-31

Date/Publication 2020-04-14

Author Pierre-Luc Germain [cre, aut] (<<https://orcid.org/0000-0003-3418-4218>>)

Maintainer Pierre-Luc Germain <pierre-luc.germain@hest.ethz.ch>

R topics documented:

data	2
doubletThresholding	2
fastClust	3
getArtificialDoublets	4

overcluster	4
plotROCs	5
rankTrans	6
resplitClusters	6
scDblFinder	7

Index	9
--------------	----------

data	<i>Comparison results</i>
------	---------------------------

Description

Results of the comparison of doublet detection methods. Each element of the list is a dataset, itself a list with scores for the different methods as well as the demuxlet SNP-based doublet calls ('demuxlet_cls').

Value

a list.

doubletThresholding	<i>doubletThresholding</i>
---------------------	----------------------------

Description

Sets the doublet scores threshold; typically called by `scDblFinder`.

Usage

```
doubletThresholding(scores, celltypes, clusters = NULL, dbr = 0.025,
  dbr.sd = 0.02, prop.fullyRandom = 0, do.plot = TRUE)
```

Arguments

scores	A vector of the doublet score for each cell (real and artificial); can be anything ranging from 0 to 1, with higher scores indicating higher change of being a doublet.
celltypes	A vector of the same length as 'scores' indicating, for each cell, whether it is a 'real' cell or an 'artificial' doublet. Missing values not allowed.
clusters	Optional vector of cluster assignment for each (real) cell, used for homotypic doublet correction.
dbr	The expected (mean) doublet rate.
dbr.sd	The standard deviation of the doublet rate, representing the uncertainty in the estimate.
prop.fullyRandom	The proportion of artificial doublets that are fully random, used for homotypic correction. Default 0.25 (the default value in 'getArtificialDoublets'). Ignored if 'clusters=NULL'
do.plot	Logical; whether to plot the thresholding data (default TRUE).

Value

A scaler indicating the decided threshold.

Examples

```
# random data
m <- t(sapply( seq(from=0, to=5, length.out=50),
              FUN=function(x) rpois(30,x) ) )
# generate doublets and merge them with real cells
doublets <- getArtificialDoublets(m, 30)
celltypes <- rep(c("real","artificial"), c(ncol(m), ncol(doublets)))
m <- cbind(m,doublets)
# dummy doublet scores:
scores <- abs(jitter(1:ncol(m),amount=10))
scores <- scores/max(scores)
# get threshold
doubletThresholding(scores, celltypes, do.plot=FALSE)
```

fastClust

fastClust

Description

fastClust

Usage

```
fastClust(sce, nfeatures = 1000, k = 10, dims = 20,
          graph.type = c("snn", "knn"), method = c("louvain", "fast_greedy",
          "overcluster"), BPPARAM = BiocParallel::SerialParam(), ...)
```

Arguments

sce	An object of class ‘SingleCellExperiment’
nfeatures	For the PCA
k	number of nearest neighbors
dims	number of PCA dimensions
graph.type	either snn or knn
method	Either ‘louvain’ or ‘fast_greedy’
BPPARAM	Passed to scan for KNN/SNN graph generation
...	passed to ‘overcluster’

Value

The ‘SingleCellExperiment’ object with an additional colData column scDbfFinder.clusters

`getArtificialDoublets` *getArtificialDoublets*

Description

Create expression profiles of random artificial doublets.

Usage

```
getArtificialDoublets(x, n = 3000, prop.fullyRandom = 0,
  clusters = NULL, n.meta.cells = 1, meta.triplets = TRUE)
```

Arguments

<code>x</code>	A count matrix, with features as rows and cells as columns.
<code>n</code>	The approximate number of doublet to generate (default 3000).
<code>prop.fullyRandom</code>	The proportion of the created doublets that are fully random (default 0); the rest will be doublets created across clusters. Ignored if 'clusters' is NULL.
<code>clusters</code>	The optional clusters labels to use to build cross-cluster doublets.
<code>n.meta.cells</code>	The number of meta-cell per cluster to create. If given, additional doublets will be created from cluster meta-cells.
<code>meta.triplets</code>	Logical; whether to create triplets from meta cells. Ignored if 'clusters' is missing.

Value

A count matrix for artificial doublets.

Examples

```
m <- t(sapply( seq(from=0, to=5, length.out=50),
  FUN=function(x) rpois(30,x) ) )
doublets <- getArtificialDoublets(m, 30)
```

`overcluster`

overcluster

Description

This function deliberately overclusters based on the desired range of cluster size. It first calculates a SNN network via `'scan::buildSNNGraph'`, then runs `'igraph::cluster_fast_greedy'` until no cluster is above the size limits, and merges clusters that are too small. By default, `'rankTrans'` is used on the counts before, because it tends to produce over-clustering influenced by library size, which is desirable for producing artificial doublets.

Usage

```
overcluster(x, rtrans = c("rankTrans", "scran", "none"), min.size = 50,
            max.size = NULL)
```

Arguments

<code>x</code>	A numeric matrix, with entities (e.g. cells) as columns and features (e.g. genes) as rows. Alternatively, an object of class 'igraph'.
<code>rtrans</code>	Transformation to apply, either 'rankTrans' (default, dense step-preserving rank transformation, see 'rankTrans'), 'scran' (default; see 'scran::scaledColRanks'), or 'none' (data taken as-is). Ignored if 'x' is an 'igraph'.
<code>min.size</code>	The minimum cluster size (applies after splitting, and hence overrides 'max.size')
<code>max.size</code>	The maximum cluster size. If omitted, will be calculated on the basis of the population size and initial number of clusters.

Value

A vector of cluster labels.

Examples

```
m <- t(sapply( seq(from=0, to=5, length.out=50),
              FUN=function(x) rpois(50,x) ) )
cc <- suppressWarnings(overcluster(m,min.size=5))
table(cc)
```

plotROCs

plotROCs

Description

Plot ROC curves for given scores.

Usage

```
plotROCs(scores, truth, called.class = NULL, nbT = TRUE,
         dot.size = 5)
```

Arguments

<code>scores</code>	A data.frame with the different types of scores as columns.
<code>truth</code>	A vector of the true class corresponding to each row of 'scores'
<code>called.class</code>	A numeric vector (with names corresponding to the columns of 'scores') indicating, for each method, the number of cases called as true (i.e. the threshold decided). Those will be plotted as points.
<code>nbT</code>	Logical; whether to add a dot for each score at the number of true positives (default TRUE).
<code>dot.size</code>	The size of the dots.

Value

a ggplot

Examples

```
myscores <- list( test=1:10 )
truth <- sample(c(TRUE,FALSE), 10, TRUE)
plotROCs( myscores, truth )
```

rankTrans

rankTrans

Description

A dense rank transformation that preserves 0 and rank step size in the presence of many ties.

Usage

```
rankTrans(x)
```

Arguments

x A matrix, with samples/cells as columns and genes/features as rows.

Value

rank-transformed *x*.

Examples

```
m <- t(sapply( seq(from=0, to=5, length.out=30),
              FUN=function(x) rpois(30,x) ) )
m2 <- rankTrans(m)
```

resplitClusters

resplitClusters

Description

Split (re-cluster) clusters of an existing graph-based clustering that are above a certain size

Usage

```
resplitClusters(g, cl = NULL, max.size = 500, min.size = 50,
               renameClusters = TRUE, iterative = TRUE)
```

Arguments

<code>g</code>	An object of class 'igraph'
<code>cl</code>	A vector of cluster labels corresponding to the nodes of 'g'. If omitted, a new clustering will be run using 'igraph::cluster_fast_greedy'.
<code>max.size</code>	The maximum cluster size
<code>min.size</code>	The minimum cluster size (default none). If given, this overrides 'max.size'.
<code>renameClusters</code>	Logical; whether to rename clusters
<code>iterative</code>	Logical; whether to resplit until no cluster is above the size limit or no improvement is made (default TRUE). If FALSE, splits each cluster once.

Value

A vector of cluster assignments.

Examples

```
m <- t(sapply( seq(from=0, to=5, length.out=50),
              FUN=function(x) rpois(50,x) ) )
g <- scan::buildSNNGraph(rankTrans(m))
table(resplitClusters(g, min.size=2, max.size=20))
```

scDbfFinder

scDbfFinder

Description

Identification of doublets in single-cell RNAseq directly from counts using overclustering-based generation of artificial doublets.

Usage

```
scDbfFinder(sce, artificialDoublets = NULL, clusters = NULL,
            clust.method = c("louvain", "overcluster", "fast_greedy"),
            samples = NULL, minClusSize = min(50, ncol(sce)/5),
            maxClusSize = NULL, nfeatures = 1000, dims = 20, dbr = NULL,
            dbr.sd = 0.015, k = 20, clust.graph.type = c("snn", "knn"),
            fullTable = FALSE, verbose = is.null(samples),
            score = c("weighted", "ratio", "hybrid"), BPPARAM = SerialParam())
```

Arguments

<code>sce</code>	A SummarizedExperiment-class
<code>artificialDoublets</code>	The approximate number of artificial doublets to create. If NULL, will be the maximum of the number of cells or '5*nbClusters^2'.
<code>clusters</code>	The optional cluster assignments (if omitted, will run clustering). This is used to make doublets more efficiently. 'clusters' should either be a vector of labels for each cell, or the name of a colData column of 'sce'.

<code>clust.method</code>	The clustering method if 'clusters' is not given.
<code>samples</code>	A vector of the same length as cells (or the name of a column of 'colData(sce)'), indicating to which sample each cell belongs. Here, a sample is understood as being processed independently. If omitted, doublets will be searched for with all cells together. If given, doublets will be searched for independently for each sample, which is preferable if they represent different captures.
<code>minClusSize</code>	The minimum cluster size for 'quickCluster'/'overcluster' (default 50); ignored if 'clusters' is given.
<code>maxClusSize</code>	The maximum cluster size for 'overcluster'. Ignored if 'clusters' is given. If NA, clustering will be performed using 'quickCluster', otherwise via 'overcluster'. If missing, the default value will be estimated by 'overcluster'.
<code>nfeatures</code>	The number of top features to use (default 1000)
<code>dims</code>	The number of dimensions used to build the network (default 20)
<code>dbr</code>	The expected doublet rate. By default this is assumed to be 1% per thousand cells captured (so 4% among 4000 thousand cells), which is appropriate for 10x datasets.
<code>dbr.sd</code>	The standard deviation of the doublet rate, defaults to 0.015.
<code>k</code>	Number of nearest neighbors (for KNN graph).
<code>clust.graph.type</code>	Either 'snn' or 'knn'.
<code>fullTable</code>	Logical; whether to return the full table including artificial doublets, rather than the table for real cells only (default).
<code>verbose</code>	Logical; whether to print messages and the thresholding plot.
<code>score</code>	Score to use for final classification; either 'weighted' (default), 'ratio' or 'hybrid' (includes information about library size and detection rate).
<code>BPPARAM</code>	Used for multithreading when splitting by samples (i.e. when 'samples!=NULL'); otherwise passed to eventual PCA and K/SNN calculations.

Value

The 'sce' object with the following additional colData columns: 'scDbfFinder.ratio' (ratio of artificial doublets among neighbors), 'scDbfFinder.weighted' (the ratio of artificial doublets among neighbors weighted by their distance), 'scDbfFinder.score' (the final score used, by default the same as 'scDbfFinder.weighted'), and 'scDbfFinder.class' (whether the cell is called as 'doublet' or 'singlet'). Alternatively, if 'fullTable=TRUE', a data.frame will be returned with information about real and artificial cells.

Examples

```
library(SingleCellExperiment)
m <- t(sapply( seq(from=0, to=5, length.out=50),
              FUN=function(x) rpois(50,x) ))
sce <- SingleCellExperiment( list(counts=m) )
sce <- scDbfFinder(sce, verbose=FALSE)
table(sce$scDbfFinder.class)
```


Index

data, [2](#)
doubletsComparison (data), [2](#)
doubletThresholding, [2](#)

fastClust, [3](#)

getArtificialDoublets, [4](#)

overcluster, [4](#)

plotROCs, [5](#)

rankTrans, [6](#)
resplitClusters, [6](#)

scDblFinder, [2, 7](#)