Package 'barbieQ'

July 15, 2025

Type Package

Title Analyze Barcode Data from Clonal Tracking Experiments

Version 1.1.2

Description The barbieQ package provides a series of robust statistical tools
for analysing barcode count data generated from cell clonal tracking (i.e., lineage tracing) experiments.
In these experiments, an initial cell and its offspring collectively form a clone (i.e., lineage).
A unique barcode sequence, incorporated into the DNA of the inital cell, is inherited within the clone.
This one-to-one mapping of barcodes to clones enables clonal tracking of their behaviors.
By counting barcodes, researchers can quantify the population abundance
of individual clones under specific experimental perturbations.
barbieQ supports barcode count data preprocessing, statistical testing, and visualization.

License GPL-3

LazyData FALSE

Encoding UTF-8

VignetteBuilder knitr

Suggests knitr, rmarkdown, testthat (>= 3.0.0), BiocStyle

Imports magrittr, tidyr, dplyr, grid, circlize, ComplexHeatmap,
ggplot2, logistf, limma, stats, igraph, utils, data.table,
S4Vectors, SummarizedExperiment

biocViews Sequencing, Software, Regression, Preprocessing,
Visualization

BugReports https://github.com/Oshlack/barbieQ

URL https://github.com/Oshlack/barbieQ/issues

RoxygenNote 7.3.2

Config/testthat/edition 3

Depends R (>= 4.5)

Roxygen list(markdown = TRUE)

git_url https://git.bioconductor.org/packages/barbieQ

git_branch devel

git_last_commit 62f8b28

git_last_commit_date 2025-04-24

Repository Bioconductor 3.22

Date/Publication 2025-07-15

Author Liyang Fei [aut, cre] (ORCID: <https://orcid.org/0000-0002-3293-2094>)

Maintainer Liyang Fei <liyang.fei@petermac.org>

Contents

`clusterCorrelatingBarcodes`

Cluster correlated Barcodes based on pairwise correlation

Description

`clusterCorrelatingBarcodes()` groups Barcode into clusters by identifying correlated Barcode pairs based on two criteria:

- The proportions of Barcodes across samples exhibit high correlation, exceeding the threshold specified by `corThresh`.
- The mean CPM of the Barcode pair exceeds the threshold specified by `cpmThresh`. These two parameters can be optimized up to the users using the visualization function plotBarcodePairCorrelation.

Usage

```
clusterCorrelatingBarcodes(
  barbieQ,
  method = "pearson",
  corThresh = 0.95,
  cpmThresh = 2^10,
  preDefinedCluster = NULL
)
```

Arguments

barbieQ	A barbieQ object created by the createBarbieQ function.
method	A string specifying the correlation method to use. Defaults to 'pearson'. Options include: 'pearson', 'kendall', 'spearman'.
corThresh	A numeric value that sets the threshold for high correlation Defaults to 0.95
cpmThresh	A numeric value that sets the minimum level of Barcode pair's mean CPM for a Barcode pair to be considered as highly correlated co-existing Barcodes. Defaults to 2^10.
preDefinedCluster	A list of known groups containing different Barcodes, or a vector/array indicating Barcode groups; or an equivalent matrix, data.frame, or DataFrame with a single column. Defaults to NULL.

Value

A barbieQ object updated with barcodeCorrelatedCluster as a DataFrame saved to the rowData of barbieQ, containing a single column of identified Barcode cluster groups, where the metadata saves a plot of the identified Barcode clusters.

Examples

```
nbarcodes <- 50
nsamples <- 12
count <- abs(matrix(rnorm(nbarcodes * nsamples), nbarcodes, nsamples))
rownames(count) <- paste0('Barcode', seq_len(nbarcodes))
barbieQ <- createBarbieQ(count)
clusterCorrelatingBarcodes(barbieQ, preDefinedCluster = c(rep(seq_len(10), 5)))
clusterCorrelatingBarcodes(barbieQ, preDefinedCluster = list(
 group1 = c('Barcode1', 'Barcode2', 'Barcode3'), group2 = c('Barcode4', 'Barcode5')))
```

---

createBarbieQ *Extract Barcode count data and build into a* barbieQ *object for subsequent analysis*

Description

createBarbieQ() constructs an object-centred list called barbieQ object, which is designed to process Barcode count data gained from cell clonal tracking experiments.

Usage

```
createBarbieQ(object, sampleMetadata = NULL, factorColors = NULL)
```

Arguments

object	A numeric matrix of Barcode counts, with Barcodes in rows and samples in columns; Alternatively, you can pass an existing barbieQ object, from which barcode counts, sample conditions, and the mapping colors will be inherited.

`sampleMetadata` A `matrix` or `data.frame` where rows represent samples and columns represent factors, each containing the specific condition assigned to a sample. If not specified, all samples are assigned the same condition. If a `barbieQ` object is passed to `object`, the sample conditions will be updated by specifying `sampleMetadata`.

`factorColors` A `list` mapping factors in `sampleMetadata` to color palettes, where each element is a named `vector` assigning colors to specific sample conditions. If not specified, defaults to `NULL`. If a `barbieQ` object is passed to `object`, the mapping colors will be updated by specifying `factorColors`.

Value

A `barbieQ` is a `SummarizedExperimnt` object, including several components:

- `assay`:A `data.frame` in `assays` storing Barcode counts, with Barcodes in rows and samples in columns.
- `sampleMetadata`: A `DataFrame` in `colData` storing sample conditions, with samples in rows and experimental factors in columns.
- `factorColors`: A `list` mapping factors in `sampleMetadata` to color palettes, where each element is a named `vector` assigning colors to specific sample conditions.
- Other matrices in `assays`: `proportion`, `CPM`, `occurrence`, and `rank` representing Barcode proportion, Counts Per Million (CPM), presence/absence, and ranking in each sample, based on the Barcode counts in `assay`.
- `isTopBarcode`: A `DataFrame` in `elementMetadata` (`rowData`) containing a single logical column indicating whether each Barcode is a major contributor. It is initially set to `TRUE` for all Barcodes, but should be refined by calling tagTopBarcodes.
- Additional processed information during further analysis.

Examples

```
## Sample conditions and color palettes
sampleConditions <- data.frame(
  Treat = factor(rep(c("ctrl", "drug"), each = 6)),
  Time = rep(rep(seq_len(2), each = 3), 2)
)
conditionColor <- list(
  Treat = c(ctrl = "#999999", drug = "#112233"),
  Time = c("1" = "#778899", "2" = "#998877")
)

## Barcode count data
nbarcodes <- 50
nsamples <- 12
barcodeCount <- abs(matrix(rnorm(nbarcodes * nsamples), nbarcodes, nsamples))
rownames(barcodeCount) <- paste0("Barcode", seq_len(nbarcodes))

## Passing `object`, `sampleMetadata` and `factorColors`
createBarbieQ(object = barcodeCount)
createBarbieQ(object = barcodeCount, sampleMetadata = sampleConditions)
createBarbieQ(
  object = barcodeCount, sampleMetadata = sampleConditions, factorColors = conditionColor
)
```

```
## Updating a `barbieQ` object by passing new `sampleMetadata` and `factorColors`
myBarbieQ <- createBarbieQ(barcodeCount, sampleConditions, conditionColor)
createBarbieQ(
  object = myBarbieQ, sampleMetadata = data.frame(Mouse = rep(seq_len(4), each = 3))
)
createBarbieQ(
  object = myBarbieQ, sampleMetadata = data.frame(Mouse = rep(seq_len(4), each = 3)),
  factorColors = list(
    Mouse = c("1" = "#111199", "2" = "#112200", "3" = "#441111", "4" = "#000000")
  )
)
```

---

monkeyHSPC *Monkey HSPC Cell Barcoding Data*

Description

A subset of data from a study on monkey HSPC cell expansion using barcoding technique. Unique barcodes were initially integrated into hematopoietic stem and progenitor cells (HSPCs) and subsequently passed to daughter cells. After a defined period of cell expansion, progeny cells were collected and sorted into various cell types as individual samples. Barcode counts within different samples were used to interpret the patterns of HSPC differentiation.

Usage

```
monkeyHSPC
```

Format

`monkeyHSPC`**:**

A `SummarizedExperiment` object containing:

- A barcode count matrix with 16,603 rows and 62 columns,
- A a data frame of sample metadata.
- other components associated with this `barbieQ` package.

Monkey Individual monkey ID

Months Time (in months) after cell expansion

Celltype Collected cell type ...

Details

The original monkey HSPC barcoding data were published in the following paper: Wu, Chuanfeng, et al. 'Clonal expansion and compartmentalized maintenance of rhesus macaque NK cell subsets.' Science Immunology (2018)

However, the barcode count data were analysed by the original authors using the barcodetrackR package and made available through the compatible barcodetrackRData repository on GitHub.

Here, we directly source the raw data from `barcodetrackRData`.

This dataset includes data from the 'ZG66' monkey individual only. Additional datasets are available at the source link.

Source

https://github.com/dunbarlabNIH/barcodetrackRData/tree/main/datasets/WuC_etal

`plotBarcodeHeatmap` *Plot Barcode CPM (or occurrence) across samples in a Heatmap*

Description

`plotBarcodeHeatmap()` visualizes Barcode output across samples with the option to include sample annotations. The Heatmap can display either:

- CPM: log2(CPM+1)
- occurrence: 0 or 1

Usage

```
plotBarcodeHeatmap(
  barbieQ,
  barcodeMetric = "CPM",
  splitSamples = FALSE,
  sampleMetadata = NULL,
  sampleGroup = NULL,
  barcodeAnnotation = NULL,
  sampleAnnotation = NULL
)
```

Arguments

`barbieQ`	A `barbieQ` object created by the createBarbieQ function.
`barcodeMetric`	A string indicating what to visualize. Defaults to 'CPM'. Options include: 'CPM' and 'occurrence'.
`splitSamples`	A logical barcodeMetric deciding whether to split samples into slices. Defaults to FALSE.
`sampleMetadata`	A `matrix`, `data.frame` or `DataFrame` of sample conditions, where each factor is represented in a separate column. Defaults to NULL, in which case sample conditions are inherited from `colData(barbieQ)$sampleMetadata`.
`sampleGroup`	A string representing the name of a factor from the sample conditions passed by `barbieQ` or `sampleMetadata`, or a vector of sample conditions, indicating the primary factor to split sample slices.
`barcodeAnnotation`	A row Annotation object created by the ComplexHeatmap::rowAnnotation function. Defaults to NULL, which means no Barcode annotation will be displayed in the Heatmap.
`sampleAnnotation`	A column Annotation object created by the ComplexHeatmap::HeatmapAnnotation function. Defaults to NULL, which means the sample annotations are generated from the sample conditions provided by `barbieQ` and `sampleMetadata`.

Value

A `Heatmap` S4 object displaying the heatmap of Barcode data across samples, optionally annotated with sample and Barcode information.

Examples

```
## sample conditions and color palettes
sampleConditions <- data.frame(
  Treat = factor(rep(c('ctrl', 'drug'), each = 6)),
  Time = rep(rep(seq_len(2), each = 3), 2)
)
conditionColor <- list(
  Treat = c(ctrl = '#999999', drug = '#112233'),
  Time = c('1' = '#778899', '2' = '#998877')
)
## Barcode count data
nbarcodes <- 50
nsamples <- 12
barcodeCount <- abs(matrix(10, nbarcodes, nsamples))
barcodeCount[seq(21, 50), ] <- 0.0001
rownames(barcodeCount) <- paste0('Barcode', seq_len(nbarcodes))
## create a `barbieQ` object
myBarbieQ <- createBarbieQ(barcodeCount, sampleConditions, conditionColor)
plotBarcodeHeatmap(myBarbieQ)
```

`plotBarcodeMA` *Plot Barcodes catogorized by significant change using a dot plot*

Description

`plotBarcodeMA()` use a dot plot to visualize the variation and mean of each Barcode in differential proportion or occurrence, as determined by the testBarcodeSignif function.

Usage

```
plotBarcodeMA(barbieQ)
```

Arguments

`barbieQ`	A `barbieQ` object created by the createBarbieQ function, updated with Barcode test results by calling the testBarcodeSignif function.

Value

A `ggplot` S3 class object displaying the significance level against other properties of Barcodes in a dot plot.

Examples

```
Block <- c(1, 1, 2, 3, 3, 4, 1, 1, 2, 3, 3, 4)
Treat <- factor(rep(c('ctrl', 'drug'), each = 6))
Time <- rep(rep(seq_len(2), each = 3), 2)
nbarcodes <- 50
nsamples <- 12
count <- abs(matrix(rnorm(nbarcodes * nsamples), nbarcodes, nsamples))
rownames(count) <- paste0('Barcode', seq_len(nbarcodes))
barbieQ <- createBarbieQ(count, data.frame(Treat = Treat, Time = Time))
testBB <- testBarcodeSignif(barbieQ, sampleGroup = 'Treat')
plotBarcodeMA(barbieQ = testBB)
```

plotBarcodePairCorrelation

Plot Barcode pairwise correlation

Description

plotBarcodePairCorrelation() visualizes the correlation of each pair of Barcodes in the barbieQ object using a dot plot. Correlated Barcodes can be identified as showing high correlation in Barcode proportions across samples using the clusterCorrelatingBarcodes function. Visualizing the pair wise correlation assists in determining the threshold for tagging the highly correlated Barcode clusters, which are considered multiple Barcodes existing in the same clone.

Usage

```
plotBarcodePairCorrelation(
  barbieQ,
  method = "pearson",
  yScaleMetric = "mean",
  corThresh = 0.95,
  cpmThresh = 2^10,
  preDefinedCluster = NULL
)
```

Arguments

barbieQ	A barbieQ object created by the createBarbieQ function.
method	A string specifying the correlation method to use. Defaults to 'pearson'. Options include: 'pearson', 'kendall', 'spearman'.
yScaleMetric	A string indicating what to present against correlation in the dot plot. Defaults to mean, representing the mean CPM for each pair. The alternative option is 'max'.
corThresh	A numeric value that sets the threshold for high correlation Defaults to 0.95
cpmThresh	A numeric value that sets the minimum level of Barcode pair's mean CPM for a Barcode pair to be considered as highly correlated co-existing Barcodes. Defaults to 2^10
preDefinedCluster	preDefinedCluster A list of known groups containing different Barcodes, or a vector/array indicating Barcode groups; or an equivalent matrix, data.frame, or DataFrame with a single column. Defaults to NULL.

Value

A ggplot S3 object displaying a dot plot of the correlation in CPM between each pair of Barcodes, plotted against the mean or max of their CPM.

Examples

```
nbarcodes <- 50
nsamples <- 12
count <- abs(matrix(rnorm(nbarcodes * nsamples), nbarcodes, nsamples))
rownames(count) <- paste0('Barcode', seq_len(nbarcodes))
```

```
barbieQ <- createBarbieQ(count)
plotBarcodePairCorrelation(barbieQ, preDefinedCluster = c(rep(seq_len(10), 5)))
plotBarcodePairCorrelation(barbieQ, preDefinedCluster = list(
 group1 = c('Barcode1', 'Barcode2', 'Barcode3'), group2 = c('Barcode4', 'Barcode5')))
```

`plotBarcodePareto` *Plot contributions of Barcodes split by* top *v.s.* bottom *in a circular bar plot.*

Description

After the tagTopBarcodes function tags Barcodes as either *top* or *bottom*, `plotBarcodePareto()` visualizes the proportion of each Barcode and separates them by these tags in a circular bar plot, also known as a Pareto plot.

Usage

```
plotBarcodePareto(barbieQ, absoluteProportion = FALSE)
```

Arguments

`barbieQ`	A `SummarizedExperiment` object created by the createBarbieQ function.
`absoluteProportion`	A logical value indicating whether to present absolute Barcode mean proportion (across samples) or relative values across Barcodes, Defaults to FALSE, which means it will present the percentage of (Barcode mean proportion) across Barcodes.

Value

A `ggplot` S3 class object displaying a circular bar plot, highlighting the relative total proportion of each Barcode across samples.

Note

To save the plot with its original aspect ratio, use the `ggplot2::ggsave` function and set `width = 8` and `height = 6`, or an equivalent size.

Examples

```
## sample conditions and color palettes
sampleConditions <- data.frame(
  Treat = factor(rep(c("ctrl", "drug"), each = 6)),
  Time = rep(rep(seq_len(2), each = 3), 2)
)
conditionColor <- list(
  Treat = c(ctrl = "#999999", drug = "#112233"),
  Time = c("1" = "#778899", "2" = "#998877")
)
## Barcode count data
nbarcodes <- 50
nsamples <- 12
barcodeCount <- abs(matrix(10, nbarcodes, nsamples))
```

```
barcodeCount[seq(21, 50), ] <- 0.0001
rownames(barcodeCount) <- paste0("Barcode", seq_len(nbarcodes))
## create a `barbieQ` object
myBarbieQ <- createBarbieQ(barcodeCount, sampleConditions, conditionColor)
myBarbieQ <- tagTopBarcodes(myBarbieQ)
plotBarcodePareto(myBarbieQ)
```

`plotBarcodeProportion` *Plot Barcode contributions as mean Barcode proportion across samples*

Description

`plotBarcodeProportion()` visualizes the mean proportion of each Barcode across samples using a bar plot, allowing for an easy comparison of Barcode contributions.

Usage

```
plotBarcodeProportion(
  barbieQ,
  absoluteProportion = FALSE,
  coordFixed = FALSE,
  colorGradient = FALSE
)
```

Arguments

`barbieQ`	A `SummarizedExperiment` object created by the createBarbieQ function.
`absoluteProportion`	A logical value indicating whether to present absolute Barcode mean proportion (across samples) or relative values across Barcodes, Defaults to FALSE, which means it will present the percentage of (Barcode mean proportion) across Barcodes.
`coordFixed`	A logical value indicating whether to coordinate the x and y scales. Defaults to FALSE, meaning the scales are not coordinated.
`colorGradient`	A logical value indicating whether to apply a gradient to the bar colors. Defaults to FALSE, which means colors will be consistent across all bars.

Value

A `ggplot` S3 class object displaying Barcode contributions in a bar plot.

Examples

```
## sample conditions and color palettes
sampleConditions <- data.frame(
  Treat = factor(rep(c('ctrl', 'drug'), each = 6)),
  Time = rep(rep(seq_len(2), each = 3), 2)
)
conditionColor <- list(
  Treat = c(ctrl = '#999999', drug = '#112233'),
  Time = c('1' = '#778899', '2' = '#998877')
```

```
)
## Barcode count data
nbarcodes <- 50
nsamples <- 12
barcodeCount <- abs(matrix(10, nbarcodes, nsamples))
barcodeCount[seq(21, 50), ] <- 0.0001
rownames(barcodeCount) <- paste0('Barcode', seq_len(nbarcodes))
## create a `barbieQ` object
myBarbieQ <- createBarbieQ(barcodeCount, sampleConditions, conditionColor)
plotBarcodeProportion(myBarbieQ)
```

`plotBarcodePValue` *Plot Barcodes categorized by significant change using a dot plot*

Description

`plotBarcodePValue()` use a dot plot to visualize the significance level of each Barcode in differential proportion or occurrence, as determined by the testBarcodeSignif function. P.values are plotted against other properties of Barcodes specified by `xAxis`.

Usage

```
plotBarcodePValue(barbieQ, xAxis = "avgRank")
```

Arguments

`barbieQ`	A `barbieQ` object created by the createBarbieQ function, updated with Barcode test results by calling the testBarcodeSignif function.
`xAxis`	A string indicating what to visualise on the x scale of the dot plot. Options include: 'avgRank', 'totalOcc', 'avgLogCPM', and 'avgProportion'. Defaults to 'avgRank', representing the average rank of Barcodes across samples.

Value

A `ggplot` S3 class object displaying the significance level against other properties of Barcodes in a dot plot.

Examples

```
Block <- c(1, 1, 2, 3, 3, 4, 1, 1, 2, 3, 3, 4)
Treat <- factor(rep(c('ctrl', 'drug'), each = 6))
Time <- rep(rep(seq_len(2), each = 3), 2)
nbarcodes <- 50
nsamples <- 12
count <- abs(matrix(rnorm(nbarcodes * nsamples), nbarcodes, nsamples))
rownames(count) <- paste0('Barcode', seq_len(nbarcodes))
barbieQ <- createBarbieQ(count, data.frame(Treat = Treat, Time = Time))
testBB <- testBarcodeSignif(barbieQ, sampleGroup = 'Treat')
plotBarcodePValue(barbieQ = testBB)
```

`plotBarcodeSankey` *Plot total contributions of Barcodes grouped by* top *vs.* bottom *in a stacked bar plot.*

Description

After the tagTopBarcodes function tags Barcodes as either *top* or *bottom*, `plotBarcodeSankey()` visualizes their relative frequency and total contribution across all samples using a stacked bar plot, resembling a Sankey plot.

Usage

```
plotBarcodeSankey(barbieQ)
```

Arguments

`barbieQ`	A `SummarizedExperiment` object created by the createBarbieQ function.

Value

A `ggplot` S3 class object displaying the Sankey-like stacked bar plot, where Barcodes are categorized as either *top* or *bottom*.

Examples

```
## sample conditions and color palettes
sampleConditions <- data.frame(
  Treat = factor(rep(c('ctrl', 'drug'), each = 6)),
  Time = rep(rep(seq_len(2), each = 3), 2)
)
conditionColor <- list(
  Treat = c(ctrl = '#999999', drug = '#112233'),
  Time = c('1' = '#778899', '2' = '#998877')
)
## Barcode count data
nbarcodes <- 50
nsamples <- 12
barcodeCount <- abs(matrix(10, nbarcodes, nsamples))
barcodeCount[seq(21, 50), ] <- 0.0001
rownames(barcodeCount) <- paste0('Barcode', seq_len(nbarcodes))
## create a `barbieQ` object
myBarbieQ <- createBarbieQ(barcodeCount, sampleConditions, conditionColor)
myBarbieQ <- tagTopBarcodes(myBarbieQ)
plotBarcodeSankey(myBarbieQ)
```

plotSamplePairCorrelation *Plot sample pairwise correlation in a Heatmap*

Description

plotSamplePairCorrelation() visualizes the pairwise correlation of CPM values across samples in a barbieQ object, using a heatmap with a checkerboard-like pattern.

Usage

```
plotSamplePairCorrelation(
  barbieQ,
  sampleOrder = NULL,
  sampleMetadata = NULL,
  sampleGroup = NULL,
  method = "pearson"
)
```

Arguments

barbieQ	A barbieQ object created by the createBarbieQ function.
sampleOrder	A character vector of names of the factors in the sample conditions provided by barbieQ or sampleMetadata, specifying the order in which samples should be arranged. Defaults to the original order of factors in the sample conditions.
sampleMetadata	A matrix, data.frame or DataFrame of sample conditions, where each factor is represented in a separate column. Defaults to NULL, in which case sample conditions are inherited from colData(barbieQ)$sampleMetadata.
sampleGroup	A string representing the name of a factor from the sample conditions passed by barbieQ or sampleMetadata, or a vector of sample conditions, indicating the primary factor to split sample slices.
method	A string specifying the correlation method to use. Defaults to 'pearson'. Options include: 'pearson', 'spearman'.

Value

A 'Heatmap' S4 object displaying the pairwise correlation between samples in a Heatmap.

Examples

```
## sample conditions and color palettes
sampleConditions <- data.frame(
  Treat = factor(rep(c('ctrl', 'drug'), each = 6)),
  Time = rep(rep(seq_len(2), each = 3), 2)
)
conditionColor <- list(
  Treat = c(ctrl = '#999999', drug = '#112233'),
  Time = c('1' = '#778899', '2' = '#998877')
)
## Barcode count data
nbarcodes <- 50
```

```
nsamples <- 12
barcodeCount <- abs(matrix(10, nbarcodes, nsamples))
barcodeCount[seq(21, 50), ] <- 0.0001
rownames(barcodeCount) <- paste0('Barcode', seq_len(nbarcodes))
## create a `barbieQ` object
myBarbieQ <- createBarbieQ(barcodeCount, sampleConditions, conditionColor)
plotSamplePairCorrelation(myBarbieQ)
```

`plotSignifBarcodeHeatmap`

Plot Barcodes categorized by significant change using a Heatmap

---

Description

`plotSignifBarcodeHeatmap()` uses the Heatmap annotations to visualize the significance level of each Barcode in differential proportion or occurrence, as determined by the testBarcodeSignif function.

Usage

```
plotSignifBarcodeHeatmap(
  barbieQ,
  barcodeMetric = "CPM",
  sampleAnnotation = NULL
)
```

Arguments

`barbieQ`	A `barbieQ` object created by the createBarbieQ function, updated with Barcode test results by calling the testBarcodeSignif function.
`barcodeMetric`	A string indicating what to visualize. Defaults to `CPM`. Options include: 'CPM' and 'occurrence'.
`sampleAnnotation`	A column Annotation object created by the ComplexHeatmap::HeatmapAnnotation function. Defaults to samples annotated by the groups to be compared.

Value

A `Heatmap` S4 class object displaying the significance level of Barcodes in Heatmap annotations.

Examples

```
Block <- c(1, 1, 2, 3, 3, 4, 1, 1, 2, 3, 3, 4)
Treat <- factor(rep(c('ctrl', 'drug'), each = 6))
Time <- rep(rep(seq_len(2), each = 3), 2)
nbarcodes <- 50
nsamples <- 12
count <- abs(matrix(rnorm(nbarcodes * nsamples), nbarcodes, nsamples))
rownames(count) <- paste0('Barcode', seq_len(nbarcodes))
barbieQ <- createBarbieQ(count, data.frame(Treat = Treat, Time = Time))
testBB <- testBarcodeSignif(barbieQ, sampleGroup = 'Treat')
plotSignifBarcodeHeatmap(barbieQ = testBB)
```

plotSignifBarcodeProportion *Plot Barcode contributions as mean Barcode proportion across samples*

Description

plotBarcodeProportion() visualizes the overall proportion of Barcodes within groups of significance, as determined by testBarcodeSignif, for each sample.

Usage

```
plotSignifBarcodeProportion(barbieQ)
```

Arguments

barbieQ	A SummarizedExperiment object created by the createBarbieQ function, updated with Barcode test results by calling the testBarcodeSignif function.

Value

A ggplot S3 class object displaying Barcode contributions in a bar plot.

Examples

```
Block <- c(1, 1, 2, 3, 3, 4, 1, 1, 2, 3, 3, 4)
Treat <- factor(rep(c('ctrl', 'drug'), each = 6))
Time <- rep(rep(seq_len(2), each = 3), 2)
nbarcodes <- 50
nsamples <- 12
count <- abs(matrix(rnorm(nbarcodes * nsamples), nbarcodes, nsamples))
rownames(count) <- paste0('Barcode', seq_len(nbarcodes))
barbieQ <- createBarbieQ(count, data.frame(Treat = Treat, Time = Time))
testBB <- testBarcodeSignif(barbieQ, sampleGroup = 'Treat')
plotSignifBarcodeProportion(testBB)
```

tagTopBarcodes *Tag each Barcode as being part of the major contributors or not*

Description

tagTopBarcodes() tags *top* Barcodes that collectively contribute to the majority of counts across the dataset. It is designed for subsequently filtering out the background noise, i.e., filtering out Barcodes that consistently have low contributions across samples. In each sample, Barcodes are tagged as major contributing Barcodes (*top* Barcodes) or not, based on whether their combined proportion passes a defined threshold in the sample. Across the entire dataset, a Barcode is considered *top* if it is tagged as *top* in a number of samples, meeting a specified appearance threshold across all selected samples.

Usage

```
tagTopBarcodes(
  barbieQ,
  activeSamples = NULL,
  proportionThreshold = 0.99,
  nSampleThreshold = 1
)
```

Arguments

barbieQ	A SummarizedExperiment object created by the createBarbieQ function.
activeSamples	A logical vector indicating individual samples (columns) to consider or avoid when determining the *top* Barcodes across the entire dataset. Default to considering all samples in the barbieQ object.
proportionThreshold	A numeric value ranging from 0 to 1, used as a threshold for determining *top* Barcodes in each sample based on their combined proportion in theat sample. Default to 0.99.
nSampleThreshold	An integer specifying the minimum number of times a Barcode must be tagged as *top* across the selected samples (specified by activeSamples) in order to be considered *top* for the entire dataset. Default to 1.

Value

A barbieQ object with slots isTopAssay and isTopBarcode added or updated, while inheriting other components from the barbieQ object passed into the function. See barbieQ structure (as a SummarizedExperiment object) in createBarbieQ.

- isTopAssay: a logical matrix stored in assays(barbieQ) tagging Barcodes as *top* in each sample.
- isTopBarcode: a DataFrame stored in rowData(barbieQ) with a single logical column tagging Barcodes as *top* or not across the entire dataset.

Examples

```
## create a `barbieQ` object
## sample conditions and color palettes
sampleConditions <- data.frame(
  Treat = factor(rep(c('ctrl', 'drug'), each = 6)),
  Time = rep(rep(seq_len(2), each = 3), 2)
)
conditionColor <- list(
  Treat = c(ctrl = '#999999', drug = '#112233'),
  Time = c('1' = '#778899', '2' = '#998877')
)
## Barcode count data
nbarcodes <- 50
nsamples <- 12
barcodeCount <- abs(matrix(rnorm(nbarcodes * nsamples), nbarcodes, nsamples))
rownames(barcodeCount) <- paste0('Barcode', seq_len(nbarcodes))
## create a `barbieQ` object
myBarbieQ <- createBarbieQ(barcodeCount, sampleConditions, conditionColor)
## tag top Barcodes
```

```
tagTopBarcodes(myBarbieQ)
```

`testBarcodeSignif` *Test significant change in Barcode proportion or occurrence between sample groups*

Description

`testBarcodeSignif()` tests differential proportion (diffProp) or occurrence (diffOcc) for each Barcode between sample groups, with the option account for multiple factors using regression models. The results can be visualized by plotBarcodePValue, plotSignifBarcodeHeatmap, plotBarcodeMA, and plotSignifBarcodeProportion.

Usage

```
testBarcodeSignif(
  barbieQ,
  method = "diffProp",
  sampleMetadata = NULL,
  sampleGroup = NULL,
  contrastFormula = NULL,
  designFormula = NULL,
  designMatrix = NULL,
  block = NULL,
  transformation = "asin-sqrt",
  regularization = "firth"
)
```

Arguments

`barbieQ`	A `barbieQ` object created by the createBarbieQ function.
`method`	A string specifying what is to be tested. Options: 'diffProp' and 'diffOcc'. Defaults to 'diffProp'.
`sampleMetadata`	A `matrix` or `data.frame` of sample conditions, where each factor is represented in a separate column. Defaults to NULL, in which case sample conditions are inherited from `barbieQ$metadata`.
`sampleGroup`	A string representing the name of a factor from the sample conditions passed by `barbieQ` or `sampleMetadata`, or a vector of sample conditions, indicating the primary factor to be tested. Defaults to the first factor in the sample conditions.
`contrastFormula`	A string indicating the contrast between sample conditions. Defaults to contrast bewteen the original levels of conditions in the specified factor.
`designFormula`	A formula to compute the `designMatrix`, generated by the stats::as.formula function. Defaults to include all factors in the sample conditions.
`designMatrix`	A numeric matrix standardizing `sampleMetadata`, generated by the stats::model.matrix function. Defaults to be generated by `designFormula`.
`block`	A vector (array) indicating sample duplicates. Defaults to no duplicates among the samples.

`transformation` A string specifying the transformation method when testing 'diffProp'. Options include: 'asin-sqrt', 'logit', and 'none'. Defaults to 'asin-sqrt'.

`regularization` A string specifying the regularization method when testing 'diffOcc'. Options: 'firth' and 'none'. Defaults to 'firth'.

Value

A `barbieQ` object updated with a `testBarcodes` component, adding a `list` named by the test name, containing:

- `results`: a `data.frame` of statistical test results, including p-values, etc.
- `methods`: a `list` indicating the specific statistical test method used.
- `targets`: a numeric `matrix` of the standardized design matrix used in the test.

Examples

```
Block <- c(1, 1, 2, 3, 3, 4, 1, 1, 2, 3, 3, 4)
Treat <- factor(rep(seq_len(2), each = 6))
Time <- rep(rep(seq_len(2), each = 3), 2)
nbarcodes <- 50
nsamples <- 12
count <- abs(matrix(rnorm(nbarcodes * nsamples), nbarcodes, nsamples))
rownames(count) <- paste0('Barcode', seq_len(nbarcodes))
barbieQ <- createBarbieQ(count, data.frame(Treat = Treat, Time = Time))
testBarcodeSignif(barbieQ, sampleGroup = 'Treat')
```

Index