

Package ‘singleCellTK’

March 30, 2021

Type Package

Title Comprehensive and Interactive Analysis of Single Cell RNA-Seq Data

Version 2.0.0

Depends R (>= 4.0), SummarizedExperiment, SingleCellExperiment, DelayedArray, Biobase

Description Run common single cell analysis in the R console or directly through your browser. Includes many functions for import, quality control, normalization, batch correction, clustering, differential expression, and visualization..

License MIT + file LICENSE

Encoding UTF-8

biocViews SingleCell, GeneExpression, DifferentialExpression, Alignment, Clustering, ImmunoOncology

LazyData TRUE

Imports ape, batchelor, BiocGenerics, BiocParallel, colourpicker, colorspace, cowplot, cluster, ComplexHeatmap, data.table, DelayedMatrixStats, DESeq2, dplyr, DT, ExperimentHub, fields, ggplot2, ggplotify, ggrepel, ggtree, gridExtra, GSVA (>= 1.26.0), GSVAdata, igraph, KernSmooth, limma, MAST, Matrix, matrixStats, methods, msigdb, multtest, plotly, RColorBrewer, ROCR, Rtsne, S4Vectors, scater, scMerge (>= 1.2.0), scran, Seurat (>= 3.1.3), shiny, shinyFiles, shinyWidgets, shinyjs, shinyBS, shinyjqui, sva, reshape2, AnnotationDbi, shinyalert, circlize, enrichR, celda, shinycssloaders, shinythemes, uwot, DropletUtils, scds (>= 1.2.0), reticulate (>= 1.14), tools, withr, GSEABase, R.utils, zinbwave, scRNAseq (>= 2.0.2), TENxPBMCDData, yaml, rmarkdown, magrittr, kableExtra, scDbfFinder

RoxygenNote 7.1.1

Suggests testthat, Rsubread, BiocStyle, knitr, lintr, bladderbatch, xtable, spelling, org.Mm.eg.db, stringr

VignetteBuilder knitr

URL https://combiomed.github.io/sctk_docs/

BugReports <https://github.com/combiomed/singleCellTK/issues>

Language en-US

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

git_branch RELEASE_3_12

git_last_commit e97b135

git_last_commit_date 2020-10-27

Date/Publication 2021-03-29

Author David Jenkins [aut] (<<https://orcid.org/0000-0002-7451-4288>>),
 Vidya Akavoor [aut],
 Salam Alabdullatif [aut],
 Shruthi Bandyadka [aut],
 Emma Briars [aut] (<<https://orcid.org/0000-0001-9350-5523>>),
 Xinyun Cao [aut],
 Sebastian Carrasco Pro [aut],
 Tyler Faits [aut],
 Rui Hong [aut],
 Mohammed Muzamil Khan [aut],
 Yusuke Koga [aut, cre],
 Anastasia Leshchik [aut],
 Irzam Sarfraz [aut],
 Yichen Wang [aut],
 Zhe Wang [aut],
 W. Evan Johnson [aut] (<<https://orcid.org/0000-0002-6247-6595>>),
 Joshua David Campbell [aut]

Maintainer Yusuke Koga <ykoga07@bu.edu>

R topics documented:

.addSeuratToMetaDataSCE	5
.checkDiffExpResultExists	6
.computeSignificantPC	7
.extractSCEAnnotation	7
.formatDEAList	8
.getComponentNames	9
.ggBar	9
.ggDensity	10
.ggScatter	11
.ggViolin	13
.sce2adata	14
.seuratGetVariableFeatures	15
.seuratInvalidate	15
.updateAssaySCE	16
calcEffectSizes	17
combineSCE	17
computeZScore	18
constructSCE	18
convertGeneIDs	19
convertSCEToSeurat	20
convertSeuratToSCE	20
dataAnnotationColor	21
discreteColorPalette	22
distinctColors	22

downSampleCells	23
downSampleDepth	24
enrichRSCE	25
exportSCE	26
exportSCEtoAnnData	27
exportSCEtoFlatFile	28
featureIndex	28
featureNameDedup	30
findMarkerDiffExp	30
generateMeta	31
generateSimulatedData	32
getBiomarker	33
getMSigDBTable	33
getPCA	34
getSceParams	35
getTopHVG	35
getTSNE	36
getUMAP	37
gsvaSCE	39
importAnnData	40
importBUStools	41
importCellRanger	42
importCellRangerV2Sample	46
importCellRangerV3Sample	47
importDropEst	48
importExampleData	49
importFromFiles	50
importGeneSetsFromCollection	51
importGeneSetsFromGMT	53
importGeneSetsFromList	54
importGeneSetsFromMSigDB	56
importMultipleSources	57
importOptimus	58
importSEQC	59
importSTARsolo	61
iterateSimulations	62
mergeSCEColData	63
mouseBrainSubsetSCE	64
msigdb_table	64
plotBarcodeRankDropsResults	65
plotBarcodeRankScatter	66
plotBatchVariance	67
plotBcdfsResults	68
plotBiomarker	71
plotCxlsResults	72
plotDecontXResults	74
plotDEGHeatmap	77
plotDEGRegression	79
plotDEGViolin	80
plotDoubletCellsResults	81
plotDoubletFinderResults	83
plotEmptyDropsResults	86

plotEmptyDropsScatter	87
plotMarkerDiffExp	89
plotMASTThresholdGenes	90
plotPCA	91
plotRunPerCellQCResults	92
plotScdsHybridResults	94
plotSCEBarAssayData	96
plotSCEBarColData	98
plotSCEBatchFeatureMean	99
plotSCEDensity	100
plotSCEDensityAssayData	102
plotSCEDensityColData	103
plotSCEDimReduceColData	104
plotSCEDimReduceFeatures	107
plotSCEHeatmap	109
plotSCEScatter	111
plotSCEViolin	114
plotSCEViolinAssayData	115
plotSCEViolinColData	117
plotScrubletResults	118
plotTSNE	121
plotUMAP	122
qcInputProcess	123
readSingleCellMatrix	124
reportCellQC	125
reportDiffExp	126
reportDropletQC	126
reportQCTool	127
retrieveSCEIndex	128
runANOVA	129
runBarcodeRankDrops	131
runBBKNN	132
runBcnds	133
runCellQC	134
runComBat	135
runCxds	136
runCxdsBcndsHybrid	137
runDEAnalysis	138
runDecontX	139
runDESeq2	140
runDoubletCells	142
runDoubletFinder	143
runDropletQC	144
runEmptyDrops	145
runFastMNN	146
runKMeans	147
runLimmaBC	148
runLimmaDE	149
runMAST	150
runMNNCorrect	152
runPerCellQC	153
runSCANORAMA	154

runSCMerge 156

runScranSNN 157

runScrublet 158

runZINBWaVE 160

sampleSummaryStats 162

scater_logNormCounts 162

sce 163

sceBatches 164

scran_modelGeneVar 164

sctkListGeneSetCollections 165

sctkPythonInstallConda 166

sctkPythonInstallVirtualEnv 167

SEG 168

selectSCTKConda 168

selectSCTKVirtualEnvironment 169

seuratComputeHeatmap 170

seuratComputeJackStraw 171

seuratElbowPlot 172

seuratFindClusters 172

seuratFindHVG 174

seuratHeatmapPlot 174

seuratICA 175

seuratIntegration 176

seuratJackStrawPlot 177

seuratNormalizeData 177

seuratPCA 178

seuratPlotHVG 179

seuratReductionPlot 180

seuratRunTSNE 180

seuratRunUMAP 181

seuratScaleData 182

seuratSCTransform 183

simpleLog 184

singleCellTK 184

subDiffEx 185

subsetSCECols 186

subsetSCERows 187

summarizeSCE 188

trimCounts 189

visPlot 190

Index **191**

.addSeuratToMetaDataSCE

.addSeuratToMetaDataSCE Adds the input seurat object to the meta-data slot of the input sce object (after removing the data matrices)

Description

.addSeuratToMetaDataSCE Adds the input seurat object to the metadata slot of the input sce object (after removing the data matrices)

Usage

```
.addSeuratToMetaDataSCE(inSCE, seuratObject)
```

Arguments

<code>inSCE</code>	(sce) object to which seurat object should be added in the metadata slot (copy to)
<code>seuratObject</code>	seurat object which should be added to the metadata slot of sce object (copy from)

Value

Updated `SingleCellExperiment` object which now contains the seurat object in its metadata slot (excluding data matrices)

```
.checkDiffExpResultExists
```

Check if the specified MAST result in SingleCellExperiment object is complete. But does not guarantee the biological correctness.

Description

Check if the specified MAST result in `SingleCellExperiment` object is complete. But does not guarantee the biological correctness.

Usage

```
.checkDiffExpResultExists(inSCE, useResult, labelBy = NULL)
```

Arguments

<code>inSCE</code>	SingleCellExperiment inherited object. a differential expression analysis function has to be run in advance.
<code>useResult</code>	character. A string specifying the <code>analysisName</code> used when running a differential expression analysis function.
<code>labelBy</code>	A single character for a column of <code>rowData(inSCE)</code> as where to search for the labeling text. Default <code>NULL</code> .

Value

Stop point if found

.computeSignificantPC *.computeSignificantPC Computes the significant principal components from an input sce object (must contain pca slot) using stdev*

Description

.computeSignificantPC Computes the significant principal components from an input sce object (must contain pca slot) using stdev

Usage

```
.computeSignificantPC(inSCE)
```

Arguments

inSCE (sce) object with pca computed

Value

A numerical value indicating how many number of components are considered significant

.extractSCEAnnotation *Extract columns from row/colData and transfer to factors*

Description

Extract columns from row/colData and transfer to factors

Usage

```
.extractSCEAnnotation(inSCE, axis = NULL, columns = NULL, index = NULL)
```

Arguments

inSCE [SingleCellExperiment](#) inherited object.
axis Choose from "col" or "row".
columns character vector. The columns needed to be extracted. If NULL, will return an empty data.frame with matched row names. Default NULL.
index Valid index to subset the col/row.

Value

A data.frame object.

.formatDEAList	<i>Helper function for differential expression analysis methods that accepts multiple ways of conditional subsetting and returns stable index format. Meanwhile it does all the input checkings.</i>
----------------	--

Description

Helper function for differential expression analysis methods that accepts multiple ways of conditional subsetting and returns stable index format. Meanwhile it does all the input checkings.

Usage

```
.formatDEAList(
  inSCE,
  useAssay,
  index1 = NULL,
  index2 = NULL,
  class = NULL,
  classGroup1 = NULL,
  classGroup2 = NULL,
  groupName1,
  groupName2,
  analysisName,
  covariates = NULL,
  overwrite = FALSE
)
```

Arguments

inSCE	SingleCellExperiment inherited object. Required.
useAssay	character. A string specifying which assay to use. Required.
index1	Any type of indices that can subset a SingleCellExperiment inherited object by cells. Specifies which cells are of interests. Default NULL.
index2	Any type of indices that can subset a SingleCellExperiment inherited object by cells. specifies the control group against those specified by index1. If NULL when using index specification, index1 cells will be compared with all other cells. Default NULL.
class	A vector/factor with <code>ncol(inSCE)</code> elements, or a character scalar that specifies a column name of <code>colData(inSCE)</code> . Default NULL.
classGroup1	a vector specifying which "levels" given in <code>class</code> are of interests. Default NULL.
classGroup2	a vector specifying which "levels" given in <code>class</code> is the control group against those specified by <code>classGroup1</code> . If NULL when using annotation specification, <code>classGroup1</code> cells will be compared with all other cells.
groupName1	A character scalar naming the group of interests. Required.
groupName2	A character scalar naming the control group. Required.
analysisName	A character scalar naming the DEG analysis. Required
covariates	A character vector of additional covariates used in linear regression methods such as Limma and DESeq2. Default NULL
overwrite	A logical scalar. Whether to overwrite result if exists. Default FALSE.

Value

A list object with part of formatted DE analysis information

Author(s)

Yichen Wang

`.getComponentNames` *.getComponentNames* Creates a list of PC/IC components to populate the picker for PC/IC heatmap generation

Description

`.getComponentNames` Creates a list of PC/IC components to populate the picker for PC/IC heatmap generation

Usage

```
.getComponentNames(maxComponents, component = c("PC", "IC"))
```

Arguments

`maxComponents` Number of components to return for the picker
`component` Which component to use. Choices are PC or IC.

Value

List of component names (appended with PC or IC)

`.ggBar` *Bar plot plotting tool.*

Description

Visualizes specified values via a violin plot.

Usage

```
.ggBar(  
  y,  
  groupBy = NULL,  
  xlab = NULL,  
  ylab = NULL,  
  axisSize = 10,  
  axisLabelSize = 10,  
  dotSize = 1,  
  transparency = 1,  
  defaultTheme = TRUE,  
  gridLine = FALSE,  
)
```

```

summary = NULL,
title = NULL,
titleSize = 15
)

```

Arguments

<code>y</code>	Numeric values to be plotted on y-axis.
<code>groupBy</code>	Groupings for each numeric value. A user may input a vector equal length to the number of the samples in the <code>SingleCellExperiment</code> object, or can be retrieved from the <code>colData</code> slot. Default <code>NULL</code> .
<code>xlab</code>	Character vector. Label for x-axis. Default <code>NULL</code> .
<code>ylab</code>	Character vector. Label for y-axis. Default <code>NULL</code> .
<code>axisSize</code>	Size of x/y-axis ticks. Default 10.
<code>axisLabelSize</code>	Size of x/y-axis labels. Default 10.
<code>dotSize</code>	Size of dots. Default 1.
<code>transparency</code>	Transparency of the dots, values will be 0-1. Default 1.
<code>defaultTheme</code>	Removes grid in plot and sets axis title size to 10 when <code>TRUE</code> . Default <code>TRUE</code> .
<code>gridLine</code>	Adds a horizontal grid line if <code>TRUE</code> . Will still be drawn even if <code>defaultTheme</code> is <code>TRUE</code> . Default <code>FALSE</code> .
<code>summary</code>	Adds a summary statistic, as well as a crossbar to the violin plot. Options are "mean" or "median". Default <code>NULL</code> .
<code>title</code>	Title of plot. Default <code>NULL</code> .
<code>titleSize</code>	Size of title of plot. Default 15.

Value

a `ggplot` of the reduced dimensions.

`.ggDensity`

Density plot plotting tool.

Description

Visualizes values stored in the specified slot of a `SingleCellExperiment` object via a density plot.

Usage

```

.ggDensity(
  value,
  groupBy = NULL,
  xlab = NULL,
  ylab = NULL,
  axisSize = 10,
  axisLabelSize = 10,
  defaultTheme = TRUE,
  title = NULL,
  titleSize = 18,
  cutoff = NULL
)

```

Arguments

<code>value</code>	Numeric value that will be plotted via density plot.
<code>groupBy</code>	Groupings for each numeric value. A user may input a vector equal length to the number of the samples in the <code>SingleCellExperiment</code> object, or can be retrieved from the <code>colData</code> slot. Default <code>NULL</code> .
<code>xlab</code>	Character vector. Label for x-axis. Default <code>NULL</code> .
<code>ylab</code>	Character vector. Label for y-axis. Default <code>NULL</code> .
<code>axisSize</code>	Size of x/y-axis ticks. Default 10.
<code>axisLabelSize</code>	Size of x/y-axis labels. Default 10.
<code>defaultTheme</code>	Removes grid in plot and sets axis title size to 10 when <code>TRUE</code> . Default <code>TRUE</code> .
<code>title</code>	Title of plot. Default <code>NULL</code> .
<code>titleSize</code>	Size of title of plot. Default 15.
<code>cutoff</code>	Numeric value. The plot will be annotated with a vertical line if set. Default <code>NULL</code> .

Value

density plot, in `ggplot`.

`.ggScatter`

Plot results of reduced dimensions data.

Description

Plot results of reduced dimensions data and colors the plots by the input vector.

Usage

```
.ggScatter(  
  inSCE,  
  sample = NULL,  
  colorBy = NULL,  
  groupBy = NULL,  
  shape = NULL,  
  reducedDimName,  
  conditionClass = NULL,  
  labelClusters = FALSE,  
  xlab = NULL,  
  ylab = NULL,  
  axisSize = 10,  
  axisLabelSize = 10,  
  dim1 = NULL,  
  dim2 = NULL,  
  bin = NULL,  
  binLabel = NULL,  
  dotSize = 2,  
  transparency = 1,  
  colorScale = NULL,  
)
```

```

colorLow = "white",
colorMid = "gray",
colorHigh = "blue",
defaultTheme = TRUE,
title = NULL,
titleSize = 15,
legendTitle = NULL,
legendTitleSize = 12,
legendSize = 10,
combinePlot = "none",
plotLabels = NULL
)

```

Arguments

inSCE	Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results. Required
sample	Character vector. Indicates which sample each cell belongs to.
colorBy	If provided, colors dots in the scatterplot based on value.
groupBy	If provided, facet wrap the scatterplot based on value.
shape	If provided, add shapes based on the value.
reducedDimName	Saved dimension reduction name in the SingleCellExperiment object. Required.
conditionClass	class of the annotation data used in colorBy. Options are NULL, "factor" or "numeric". If NULL, class will default to the original class. Default NULL.
labelClusters	Logical. Whether the cluster labels are plotted. Default FALSE.
xlab	Character vector. Label for x-axis. Default NULL.
ylab	Character vector. Label for y-axis. Default NULL.
axisSize	Size of x/y-axis ticks. Default 10.
axisLabelSize	Size of x/y-axis labels. Default 10.
dim1	1st dimension to be used for plotting. Can either be a string which specifies the name of the dimension to be plotted from reducedDims, or a numeric value which specifies the index of the dimension to be plotted. Default is NULL.
dim2	2nd dimension to be used for plotting. Can either be a string which specifies the name of the dimension to be plotted from reducedDims, or a numeric value which specifies the index of the dimension to be plotted. Default is NULL.
bin	Numeric vector. If single value, will divide the numeric values into the 'bin' groups. If more than one value, will bin numeric values using values as a cut point.
binLabel	Character vector. Labels for the bins created by the 'bin' parameter. Default NULL.
dotSize	Size of dots. Default 2.
transparency	Transparency of the dots, values will be 0-1. Default 1.
colorScale	Vector. Needs to be same length as the number of unique levels of 'colorBy'. Will be used only if conditionClass = "factor" or "character". Default NULL.
colorLow	Character. A color available from 'colors()'. The color will be used to signify the lowest values on the scale. Default 'white'. Will be used only if conditionClass = "numeric".

<code>colorMid</code>	Character. A color available from <code>'colors()'</code> . The color will be used to signify the midpoint on the scale. Default <code>'gray'</code> . Will be used only if <code>conditionClass = "numeric"</code> .
<code>colorHigh</code>	Character. A color available from <code>'colors()'</code> . The color will be used to signify the highest values on the scale. Default <code>'blue'</code> . Will be used only if <code>conditionClass = "numeric"</code> .
<code>defaultTheme</code>	Removes grid in plot and sets axis title size to 10 when TRUE. Default TRUE.
<code>title</code>	Title of plot. Default NULL.
<code>titleSize</code>	Size of title of plot. Default 15.
<code>legendTitle</code>	title of legend. Default NULL.
<code>legendTitleSize</code>	size of legend title. Default 12.
<code>legendSize</code>	size of legend. Default 10.
<code>combinePlot</code>	Must be either "all", "sample", or "none". "all" will combine all plots into a single <code>.ggplot</code> object, while "sample" will output a list of plots separated by sample. Default "none".
<code>plotLabels</code>	labels to each plot. If set to "default", will use the name of the samples as the labels. If set to "none", no label will be plotted.

Value

a `ggplot` of the reduced dimensions.

`.ggViolin`*Violin plot plotting tool.*

Description

Visualizes specified values via a violin plot.

Usage

```
.ggViolin(  
  y,  
  groupBy = NULL,  
  violin = TRUE,  
  boxplot = TRUE,  
  dots = TRUE,  
  xlab = NULL,  
  ylab = NULL,  
  axisSize = 10,  
  axisLabelSize = 10,  
  dotSize = 1,  
  transparency = 1,  
  defaultTheme = TRUE,  
  gridLine = FALSE,  
  summary = NULL,  
  title = NULL,  
  titleSize = 15  
)
```

Arguments

y	Numeric values to be plotted on y-axis.
groupBy	Groupings for each numeric value. A user may input a vector equal length to the number of the samples in the SingleCellExperiment object, or can be retrieved from the colData slot. Default NULL.
violin	Boolean. If TRUE, will plot the violin plot. Default TRUE.
boxplot	Boolean. If TRUE, will plot boxplots for each violin plot. Default TRUE.
dots	Boolean. If TRUE, will plot dots for each violin plot. Default TRUE.
xlab	Character vector. Label for x-axis. Default NULL.
ylab	Character vector. Label for y-axis. Default NULL.
axisSize	Size of x/y-axis ticks. Default 10.
axisLabelSize	Size of x/y-axis labels. Default 10.
dotSize	Size of dots. Default 1.
transparency	Transparency of the dots, values will be 0-1. Default 1.
defaultTheme	Removes grid in plot and sets axis title size to 10 when TRUE. Default TRUE.
gridLine	Adds a horizontal grid line if TRUE. Will still be drawn even if defaultTheme is TRUE. Default FALSE.
summary	Adds a summary statistic, as well as a crossbar to the violin plot. Options are "mean" or "median". Default NULL.
title	Title of plot. Default NULL.
titleSize	Size of title of plot. Default 15.

Value

a ggplot of the reduced dimensions.

.sce2adata	<i>Coverts SingleCellExperiment object from R to anndata.AnnData object in Python</i>
------------	---

Description

The AnnData object here can be saved to .h5ad file and read into Python interactive console. Mostly used scenario is when you want to apply reticulated Python function, which only works with an anndata.AnnData object.

Usage

```
.sce2adata(SCE, useAssay = "counts")
```

Arguments

SCE	A SingleCellExperiment object.
useAssay	Character, default "counts". The name of assay of interests that will be set as the primary matrix of the output AnnData. Available options can be listed by 'assayNames(SCE)'. The primary matrix will be saved in 'adata\$X', Other assays will be stored in 'adata\$obsm' together with the low-dimension representations (for now).

Value

A Python anndata.AnnData object

.seuratGetVariableFeatures *.seuratGetVariableFeatures* Retrieves the requested number of variable feature names

Description

.seuratGetVariableFeatures Retrieves the requested number of variable feature names

Usage

```
.seuratGetVariableFeatures(inSCE, numberOfFeatures)
```

Arguments

inSCE (sce) object from which to extract the variable feature names
numberOfFeatures numerical value indicating how many feature names should be retrieved (default is 100)

Value

list() of variable feature names

.seuratInvalidate *.seuratInvalidate* Removes seurat data from the input SingleCellExperiment object specified by the task in the Seurat workflow.

Description

.seuratInvalidate Removes seurat data from the input SingleCellExperiment object specified by the task in the Seurat workflow.

Usage

```
.seuratInvalidate(  
  inSCE,  
  scaleData = TRUE,  
  varFeatures = TRUE,  
  PCA = TRUE,  
  ICA = TRUE,  
  tSNE = TRUE,  
  UMAP = TRUE,  
  clusters = TRUE  
)
```

Arguments

<code>inSCE</code>	Input <code>SingleCellExperiment</code> object to remove Seurat data from.
<code>scaleData</code>	Remove scaled data from <code>seurat</code> . Default <code>TRUE</code> .
<code>varFeatures</code>	Remove variable features from <code>seurat</code> . Default <code>TRUE</code> .
<code>PCA</code>	Remove PCA from <code>seurat</code> . Default <code>TRUE</code> .
<code>ICA</code>	Remove ICA from <code>seurat</code> . Default <code>TRUE</code> .
<code>tSNE</code>	Remove tSNE from <code>seurat</code> . Default <code>TRUE</code> .
<code>UMAP</code>	Remove UMAP from <code>seurat</code> . Default <code>TRUE</code> .
<code>clusters</code>	Remove clusters from <code>seurat</code> . Default <code>TRUE</code> .

Value

Updated `SingleCellExperiment` object containing the Seurat object in the metadata slot with the data removed

<code>.updateAssaySCE</code>	<i>.updateAssaySCE Update/Modify/Add an assay in the provided <code>SingleCellExperiment</code> object from a Seurat object</i>
------------------------------	---

Description

`.updateAssaySCE` Update/Modify/Add an assay in the provided `SingleCellExperiment` object from a Seurat object

Usage

```
.updateAssaySCE(
  inSCE,
  seuratObject,
  assaySlotSCE,
  seuratDataSlot = "counts",
  seuratAssaySlot = "RNA"
)
```

Arguments

<code>inSCE</code>	Input <code>SingleCellExperiment</code> object
<code>seuratObject</code>	Input Seurat object
<code>assaySlotSCE</code>	Selected assay to update in the input <code>SingleCellExperiment</code> object
<code>seuratDataSlot</code>	Selected data slot from the Seurat object. Default <code>"counts"</code> .
<code>seuratAssaySlot</code>	Selected assay from Seurat object. Default <code>"RNA"</code> .

Value

A `SingleCellExperiment` object with data from Seurat object appended to the `assay` slot.

calcEffectSizes	<i>Finds the effect sizes for all genes in the original dataset, regardless of significance.</i>
-----------------	--

Description

Finds the effect sizes for all genes in the original dataset, regardless of significance.

Usage

```
calcEffectSizes(countMatrix, condition)
```

Arguments

countMatrix	Matrix. A simulated counts matrix, sans labels.
condition	Factor. The condition labels for the simulated cells. If more than 2 conditions are given, the first will be compared to all others by default.

Value

A vector of cohen's d effect sizes for each gene.

Examples

```
data("mouseBrainSubsetSCE")
res <- calcEffectSizes(assay(mouseBrainSubsetSCE, "counts"),
                      condition = colData(mouseBrainSubsetSCE)[, "level1class"])
```

combineSCE	<i>Combine a list of SingleCellExperiment objects as one SingleCellExperiment object</i>
------------	--

Description

Combine a list of SingleCellExperiment objects as one SingleCellExperiment object

Usage

```
combineSCE(sceList, by.r, by.c, combined)
```

Arguments

sceList	A list contains SingleCellExperiment objects. Currently, combineSCE function only support combining SCE objects with assay in dgCMatrx format. It does not support combining SCE with assay in delayedArray format.
by.r	Specifications of the columns used for merging rowData. See 'Details'.
by.c	Specifications of the columns used for merging colData. See 'Details'.
combined	logical; if TRUE, it will combine the list of SingleCellExperiment objects. See 'Details'.

Value

A [SingleCellExperiment](#) object which combines all objects in sceList. The colData is merged.

Examples

```
combinedsce <- combineSCE(list(sce,sce), by.r = NULL, by.c = NULL, combined = TRUE)
```

computeZScore	<i>Compute Z-Score</i>
---------------	------------------------

Description

Computes Z-Score from an input count matrix using the formula $((x - \text{mean}(x)) / \text{sd}(x))$ for each gene across all cells. The input count matrix can either be a base matrix, dgCMMatrix or a DelayedMatrix. Computations are performed using DelayedMatrixStats package to efficiently compute the Z-Score matrix.

Usage

```
computeZScore(counts)
```

Arguments

counts matrix (base matrix, dgCMMatrix or DelaymedMatrix)

Value

z-score computed counts matrix (DelayedMatrix)

Examples

```
data(sce_chc1, package = "scds")
assay(sce_chc1, "countsZScore") <- computeZScore(assay(sce_chc1, "counts"))
```

constructSCE	<i>Create SingleCellExperiment object from csv or txt input</i>
--------------	---

Description

Create SingleCellExperiment object from csv or txt input

Usage

```
constructSCE(data, samplename)
```

Arguments

data A [data.table](#) object containing the count matrix.
 samplename The sample name of the data.

Value

A [SingleCellExperiment](#) object containing the count matrix.

convertGeneIDs	<i>Convert Gene IDs</i>
----------------	-------------------------

Description

Convert the gene IDs in a [SingleCellExperiment](#) object using Bioconductor `org.*.eg.db` data packages. Because annotation databases do not have a 1:1 relationship, this tool removes rows with no corresponding annotation in your desired annotation, and remove any duplicate annotations after conversion.

Usage

```
convertGeneIDs(inSCE, inSymbol, outSymbol, database = "org.Hs.eg.db")
```

Arguments

<code>inSCE</code>	Input SingleCellExperiment object.
<code>inSymbol</code>	The input symbol type
<code>outSymbol</code>	The output symbol type
<code>database</code>	The <code>org.*.eg.db</code> database to use. The default is <code>org.Hs.eg.db</code>

Value

A [SingleCellExperiment](#) object with converted gene IDs.

Examples

```
if(requireNamespace("org.Mm.eg.db", quietly = TRUE)) {
  #convert mouse gene symbols to ensembl IDs
  library("org.Mm.eg.db")
  sample(rownames(mouseBrainSubsetSCE), 50)
  mouseBrainSubsetSymbol <- convertGeneIDs(inSCE = mouseBrainSubsetSCE,
                                           inSymbol = "SYMBOL",
                                           outSymbol = "ENSEMBL",
                                           database = "org.Mm.eg.db")
  sample(rownames(mouseBrainSubsetSymbol), 50)
}
```

convertSCEToSeurat	<i>convertSCEToSeurat Converts sce object to seurat while retaining all assays and metadata</i>
--------------------	---

Description

convertSCEToSeurat Converts sce object to seurat while retaining all assays and metadata

Usage

```
convertSCEToSeurat(
  inSCE,
  countsAssay = NULL,
  normAssay = NULL,
  scaledAssay = NULL
)
```

Arguments

inSCE	A SingleCellExperiment object to convert to a Seurat object.
countsAssay	Which assay to use from sce object for raw counts. Default NULL.
normAssay	Which assay to use from sce object for normalized data. Default NULL.
scaledAssay	Which assay to use from sce object for scaled data. Default NULL.

Value

Updated seurat object that contains all data from the input sce object

Examples

```
data(scExample, package = "singleCellTK")
seurat <- convertSCEToSeurat(sce)
```

convertSeuratToSCE	<i>convertSeuratToSCE Converts the input seurat object to a sce object</i>
--------------------	--

Description

convertSeuratToSCE Converts the input seurat object to a sce object

Usage

```
convertSeuratToSCE(
  seuratObject,
  normAssayName = "seuratNormData",
  scaledAssayName = "seuratScaledData"
)
```

Arguments

seuratObject Input Seurat object

normAssayName Name of assay to store the normalized data. Default "seuratNormData".

scaledAssayName Name of assay to store the scaled data. Default "seuratScaledData".

Value

SingleCellExperiment output object

Examples

```
data(scExample, package = "singleCellTK")
seurat <- convertSCEToSeurat(sce)
sce <- convertSeuratToSCE(seurat)
```

dataAnnotationColor *Generate distinct colors for all categorical col/rowData entries. Character columns will be considered as well as all-integer columns. Any column with all-distinct values will be excluded.*

Description

Generate distinct colors for all categorical col/rowData entries. Character columns will be considered as well as all-integer columns. Any column with all-distinct values will be excluded.

Usage

```
dataAnnotationColor(inSCE, axis = NULL, colorGen = distinctColors)
```

Arguments

inSCE [SingleCellExperiment](#) inherited object.

axis Choose from "col" or "row".

colorGen A function that generates color code vector by giving an integer for the number of colors. Alternatively, [rainbow](#). Default [distinctColors](#).

Value

A list object containing distinct colors mapped to all possible categorical entries in `rowData(inSCE)` or `colData(inSCE)`.

Author(s)

Yichen Wang

discreteColorPalette *Generate given number of color codes*

Description

Three different generation methods are wrapped, including `distinctColors`, `[randomcolorR](SCTK_PerformingQC_Ce` and the ggplot default color generation.

Usage

```
discreteColorPalette(
  n,
  palette = c("random", "ggplot", "celda"),
  seed = 12345,
  ...
)
```

Arguments

n	An integer, the number of color codes to generate.
palette	A single character string. Select the method, available options are "ggplot", "celda" and "random". Default "random".
seed	An integer. Set the seed for random process that happens only in "random" generation. Default 12345.
...	Other arguments that are passed to the internal function, according to the method selected.

Value

A character vector of n hex color codes.

Examples

```
discreteColorPalette(n = 3)
```

distinctColors *Generate a distinct palette for coloring different clusters*

Description

Generate a distinct palette for coloring different clusters

Usage

```
distinctColors(
  n,
  hues = c("red", "cyan", "orange", "blue", "yellow", "purple", "green", "magenta"),
  saturation.range = c(0.7, 1),
  value.range = c(0.7, 1)
)
```

Arguments

n	Integer; Number of colors to generate
hues	Character vector of R colors available from the colors() function. These will be used as the base colors for the clustering scheme. Different saturations and values (i.e. darkness) will be generated for each hue.
saturation.range	Numeric vector of length 2 with values between 0 and 1. Default: c(0.25, 1)
value.range	Numeric vector of length 2 with values between 0 and 1. Default: c(0.5, 1)

Value

A vector of distinct colors that have been converted to HEX from HSV.

Examples

```
distinctColors(10)
```

downSampleCells	<i>Estimate numbers of detected genes, significantly differentially expressed genes, and median significant effect size</i>
-----------------	---

Description

Estimate numbers of detected genes, significantly differentially expressed genes, and median significant effect size

Usage

```
downSampleCells(
  originalData,
  useAssay = "counts",
  minCountDetec = 10,
  minCellsDetec = 3,
  minCellnum = 10,
  maxCellnum = 1000,
  realLabels,
  depthResolution = 10,
  iterations = 10,
  totalReads = 1e+06
)
```

Arguments

originalData	The SingleCellExperiment object storing all assay data from the shiny app.
useAssay	Character. The name of the assay to be used for subsampling.
minCountDetec	Numeric. The minimum number of reads found for a gene to be considered detected.
minCellsDetec	Numeric. The minimum number of cells a gene must have at least 1 read in for it to be considered detected.

minCellnum	Numeric. The minimum number of virtual cells to include in the smallest simulated dataset.
maxCellnum	Numeric. The maximum number of virtual cells to include in the largest simulated dataset
realLabels	Character. The name of the condition of interest. Must match a name from sample data. If only two factors present in the corresponding colData, will default to t-test. If multiple factors, will default to ANOVA.
depthResolution	Numeric. How many different read depth should the script simulate? Will simulate a number of experimental designs ranging from 10 reads to maxReadDepth, with logarithmic spacing.
iterations	Numeric. How many times should each experimental design be simulated?
totalReads	Numeric. How many aligned reads to put in each simulated dataset.

Value

A 3-dimensional array, with dimensions = c(iterations, depthResolution, 3). [.,1] contains the number of detected genes in each simulated dataset, [.,2] contains the number of significantly differentially expressed genes in each simulation, and [.,3] contains the mediansignificant effect size in each simulation. If no genes are significantly differentially expressed, the median effect size defaults to infinity.

Examples

```
data("mouseBrainSubsetSCE")
subset <- mouseBrainSubsetSCE[1:1000,]
res <- downSampleCells(subset,
                       realLabels = "level1class",
                       iterations=2)
```

downSampleDepth	<i>Estimate numbers of detected genes, significantly differentially expressed genes, and median significant effect size</i>
-----------------	---

Description

Estimate numbers of detected genes, significantly differentially expressed genes, and median significant effect size

Usage

```
downSampleDepth(
  originalData,
  useAssay = "counts",
  minCount = 10,
  minCells = 3,
  maxDepth = 1e+07,
  realLabels,
  depthResolution = 10,
  iterations = 10
)
```


Arguments

originalData	SingleCellExperiment object storing all assay data from the shiny app.
useAssay	Character. The name of the assay to be used for subsampling.
minCount	Numeric. The minimum number of reads found for a gene to be considered detected.
minCells	Numeric. The minimum number of cells a gene must have at least 1 read in for it to be considered detected.
maxDepth	Numeric. The highest number of total reads to be simulated.
realLabels	Character. The name of the condition of interest. Must match a name from sample data.
depthResolution	Numeric. How many different read depth should the script simulate? Will simulate a number of experimental designs ranging from 10 reads to maxReadDepth, with logarithmic spacing.
iterations	Numeric. How many times should each experimental design be simulated?

Value

A 3-dimensional array, with dimensions = c(iterations, depthResolution, 3). [,,1] contains the number of detected genes in each simulated dataset, [,,2] contains the number of significantly differentially expressed genes in each simulation, and [,,3] contains the median significant effect size in each simulation. If no genes are significantly differentially expressed, the median effect size defaults to infinity.

Examples

```
data("mouseBrainSubsetSCE")
subset <- mouseBrainSubsetSCE[1:1000,]
res <- downSampleDepth(subset,
  realLabels = "level1class",
  iterations=2)
```

enrichRSCE	<i>enrichR</i> Given a list of genes this function runs the <code>enrichR()</code> to perform Gene enrichment
------------	---

Description

`enrichR` Given a list of genes this function runs the `enrichR()` to perform Gene enrichment

Usage

```
enrichRSCE(inSCE, glist, db = NULL)
```

Arguments

inSCE	Input SingleCellExperiment object.
glist	selected genes for enrichment analysis using <code>enrichR()</code> . Required
db	selected database name from the <code>enrichR</code> database list. if <code>NULL</code> then <code>enrichR</code> will be run on all the available databases on the <code>enrichR</code> database.

Value

enrichRSCE(): returns a data.frame of enrichment terms overlapping in the respective databases along with p-values, z-scores etc.,

Examples

```
## Not run:
enrichRSCE(mouseBrainSubsetSCE, "Cmtm5", "GO_Cellular_Component_2017")

## End(Not run)
```

 exportSCE

Export data in SingleCellExperiment object

Description

Export data in SingleCellExperiment object

Usage

```
exportSCE(
  inSCE,
  samplename = "sample",
  directory = "./",
  type = "Cells",
  format = c("SCE", "AnnData", "FlatFile", "HTAN")
)
```

Arguments

inSCE	A SingleCellExperiment object that contains the data. QC metrics are stored in colData of the singleCellExperiment object.
samplename	Sample name. This will be used as name of subdirectories and the prefix of flat file output. Default is 'sample'.
directory	Output directory. Default is './'.
type	Type of data. The type of data stored in SingleCellExperiment object. It can be 'Droplets' (raw droplets matrix) or 'Cells' (cells matrix).
format	The format of output. It currently supports flat files, rds files and python h5 files. It can output multiple formats. Default: c("SCE", "AnnData", "FlatFile", "HTAN").

Value

Generates a file containing data from inSCE, in specified format.

Examples

```
data(scExample)
## Not run:
exportSCE(sce, format = "SCE")

## End(Not run)
```

exportSCEtoAnnData *Export a [SingleCellExperiment](#) R object as Python `annData` object*

Description

Writes all assays, colData, rowData, reducedDims, and altExps objects in a [SingleCellExperiment](#) to a Python `annData` object in the `.h5ad` format. All parameters of `AnnData.write_h5ad` function (https://icb-anndata.readthedocs-hosted.com/en/stable/anndata.AnnData.write_h5ad.html) are available as parameters to this export function and set to defaults. Defaults can be overridden at function call.

Usage

```
exportSCEtoAnnData(
  sce,
  useAssay = "counts",
  outputDir = "./",
  prefix = "sample",
  overwrite = TRUE,
  compression = c("None", "lzf", "gzip"),
  compressionOpts = NULL,
  forceDense = c("False", "True")
)
```

Arguments

<code>sce</code>	SingleCellExperiment R object to be exported.
<code>useAssay</code>	Character. The name of assay of interests that will be set as the primary matrix of the output <code>AnnData</code> . Default "counts".
<code>outputDir</code>	Path to the directory where <code>.h5ad</code> outputs will be written. Default is the current working directory.
<code>prefix</code>	Prefix to use for the name of the output file. Default "sample".
<code>overwrite</code>	Boolean. Default TRUE.
<code>compression</code>	If output file compression is required, this variable accepts 'gzip' or 'lzf' as inputs. Default None.
<code>compressionOpts</code>	Integer. Sets the compression level
<code>forceDense</code>	Default False Write sparse data as a dense matrix. Refer <code>anndata.write_h5ad</code> documentation for details. Default NULL.

Value

Generates a Python `anndata` object containing data from `inSCE`.

Examples

```
data(sce_chc1, package = "scds")
## Not run:
exportSCEtoAnnData(sce=sce_chc1, compression="gzip")

## End(Not run)
```

exportSCEtoFlatFile *Export a [SingleCellExperiment](#) object to flat text files*

Description

Writes all assays, colData, rowData, reducedDims, and altExps objects in a [SingleCellExperiment](#) to text files. The items in the 'metadata' slot remain stored in list and are saved in an RDS file.

Usage

```
exportSCEtoFlatFile(
  sce,
  outputDir = "./",
  overwrite = TRUE,
  gzipped = TRUE,
  sample = "sample"
)
```

Arguments

sce	SingleCellExperiment object to be exported.
outputDir	Name of the directory to store the exported file(s).
overwrite	Boolean. Whether to overwrite the output files. Default TRUE.
gzipped	Boolean. TRUE if the output files are to be gzip compressed. FALSE otherwise. Default TRUE.
sample	Name of the sample. It will be used as the prefix of file names.

Value

Generates text files containing data from inSCE.

Examples

```
data(sce_chcl, package = "scds")
## Not run:
exportSCEtoFlatFile(sce_chcl, "sce_chcl")

## End(Not run)
```

featureIndex *Retrieve row index for a set of features*

Description

This will return indices of features among the rownames or rowData of a data.frame, matrix, or a [SummarizedExperiment](#) object including a [SingleCellExperiment](#). Partial matching (i.e. grepping) can be used by setting exactMatch = FALSE.

Usage

```
featureIndex(
  features,
  inSCE,
  by = "rownames",
  exactMatch = TRUE,
  removeNA = FALSE,
  errorOnNoMatch = TRUE,
  warningOnPartialMatch = TRUE
)
```

Arguments

features	Character vector of feature names to find in the rows of inSCE.
inSCE	A data.frame, matrix, or SingleCellExperiment object to search.
by	Character. Where to search for features in inSCE. If set to "rownames" then the features will be searched for among rownames(inSCE). If inSCE inherits from class SummarizedExperiment , then by can be one of the fields in the row annotation data.frame (i.e. one of colnames(rowData(inSCE))).
exactMatch	Boolean. Whether to only identify exact matches or to identify partial matches using grep .
removeNA	Boolean. If set to FALSE, features not found in inSCE will be given NA and the returned vector will be the same length as features. If set to TRUE, then the NA values will be removed from the returned vector. Default FALSE.
errorOnNoMatch	Boolean. If TRUE, an error will be given if no matches are found. If FALSE, an empty vector will be returned if removeNA is set to TRUE or a vector of NA if removeNA is set to FALSE. Default TRUE.
warningOnPartialMatch	Boolean. If TRUE, a warning will be given if some of the entries in features were not found in inSCE. The warning will list the features not found. Default TRUE.

Value

A vector of row indices for the matching features in inSCE.

Author(s)

Yusuke Koga, Joshua D. Campbell

See Also

[`retrieveFeatureInfo`](#) from package 'scater' and [link{regex}](#) for how to use regular expressions when exactMatch = FALSE.

Examples

```
data(scExample)
ix <- featureIndex(features = c("MT-CYB", "MT-ND2"),
  inSCE = sce,
  by = "feature_name")
```

featureNameDedup	<i>Resolve duplicated feature names in a matrix</i>
------------------	---

Description

Adds '-1', '-2', ... '-i' to multiple duplicated feature names

Usage

```
featureNameDedup(countmat)
```

Arguments

countmat matrix, with row names as feature names that need to be resolved.

Value

The same matrix as input with rowname duplication resolved.

findMarkerDiffExp	<i>Find the marker gene set for each cluster With an input SingleCellExperiment object and specifying the clustering labels, this function iteratively call the differential expression analysis on each cluster against all the others.</i>
-------------------	--

Description

Find the marker gene set for each cluster With an input SingleCellExperiment object and specifying the clustering labels, this function iteratively call the differential expression analysis on each cluster against all the others.

Usage

```
findMarkerDiffExp(
  inSCE,
  useAssay = "logcounts",
  method = "MAST",
  cluster = "cluster",
  covariates = NULL,
  log2fcThreshold = 0.25,
  fdrThreshold = 0.05
)
```

Arguments

inSCE	SingleCellExperiment inherited object.
useAssay	character. A string specifying which assay to use for the MAST calculations. Default "logcounts".
method	A single character for specific differential expression analysis method. Choose from 'MAST', 'DESeq2', 'Limma', 'ANOVA'. Default "MAST".
cluster	One single character to specify a column in colData(inSCE) for the clustering label. Alternatively, a vector or a factor is also acceptable. Default "cluster".
covariates	A character vector of additional covariates to use when building the model. All covariates must exist in names(colData(inSCE)). Not applicable when method is "MAST" method. Default NULL.
log2fcThreshold	Only out put DEGs with the absolute values of log2FC larger than this value. Default NULL
fdrThreshold	Only out put DEGs with FDR value smaller than this value. Default 1

Value

The input [SingleCellExperiment](#) object with `metadata(inSCE)$findMarker` updated with a `data.table` of the up- regulated DEGs for each cluster.

Author(s)

Yichen Wang

generateMeta	<i>Generate manifest file for droplet and cell count data</i>
--------------	---

Description

Generate manifest file for droplet and cell count data

Usage

```
generateMeta(
  dropletSCE = NULL,
  cellSCE = NULL,
  samplename,
  dir,
  HTAN = TRUE,
  dataType = c("Droplet", "Cell", "Both")
)
```

Arguments

dropletSCE	A SingleCellExperiment object containing droplet count matrix data
cellSCE	A SingleCellExperiment object containing cell count matrix data
samplename	The sample name of the SingleCellExperiment objects

dir	The output directory of the SCTK QC pipeline.
HTAN	Whether generate manifest file with the format required by HTAN. Default is TRUE.
dataType	Type of the input data. It can be one of "Droplet", "Cell" or "Both".

Value

A [SingleCellExperiment](#) object which combines all objects in sceList. The colData is merged.

Examples

```
data(scExample, package = "singleCellTK")
generateMeta(sce, dir = ".", samplename = "Sample",
  HTAN = TRUE)
```

generateSimulatedData *Generates a single simulated dataset, bootstrapping from the input counts matrix.*

Description

Generates a single simulated dataset, bootstrapping from the input counts matrix.

Usage

```
generateSimulatedData(totalReads, cells, originalData, realLabels)
```

Arguments

totalReads	Numeric. The total number of reads in the simulated dataset, to be split between all simulated cells.
cells	Numeric. The number of virtual cells to simulate.
originalData	Matrix. The original raw read count matrix. When used within the Shiny app, this will be assay(SCEsetObject, "counts").
realLabels	Factor. The condition labels for differential expression. If only two factors present, will default to t-test. If multiple factors, will default to ANOVA.

Value

A simulated counts matrix, the first row of which contains the 'true' labels for each virtual cell.

Examples

```
data("mouseBrainSubsetSCE")
res <- generateSimulatedData(
  totalReads = 1000, cells=10,
  originalData = assay(mouseBrainSubsetSCE, "counts"),
  realLabels = colData(mouseBrainSubsetSCE)[, "level1class"])
```

getBiomarker	<i>Given a list of genes and a SingleCellExperiment object, return the binary or continuous expression of the genes.</i>
--------------	--

Description

Given a list of genes and a SingleCellExperiment object, return the binary or continuous expression of the genes.

Usage

```
getBiomarker(inSCE, gene, binary = "Binary", useAssay = "counts")
```

Arguments

inSCE	Input SingleCellExperiment object.
gene	gene list
binary	"Binary" for binary expression or "Continuous" for a gradient. Default: "Binary"
useAssay	Indicate which assay to use. The default is "counts".

Value

getBiomarker(): A data.frame of expression values

Examples

```
getBiomarker(mouseBrainSubsetSCE, gene="C1qa")
```

getMSigDBTable	<i>Shows MSigDB categories</i>
----------------	--------------------------------

Description

Returns a data.frame that shows MSigDB categories and subcategories as well as descriptions for each. The entries in the ID column in this table can be used as input for [importGeneSetsFromMSigDB](#).

Usage

```
getMSigDBTable()
```

Value

data.frame, containing MSigDB categories

Author(s)

Joshua D. Campbell

See Also

[importGeneSetsFromMSigDB](#) for importing MSigDB gene sets.

getPCA	<i>Perform PCA on a SingleCellExperiment Object A wrapper to runPCA function to compute principal component analysis (PCA) from a given SingleCellExperiment object.</i>
--------	--

Description

Perform PCA on a SingleCellExperiment Object A wrapper to [runPCA](#) function to compute principal component analysis (PCA) from a given [SingleCellExperiment](#) object.

Usage

```
getPCA(
  inSCE,
  useAssay = "logcounts",
  useAltExp = NULL,
  reducedDimName = "PCA",
  ndim = 50,
  scale = TRUE,
  ntop = NULL
)
```

Arguments

inSCE	Input SingleCellExperiment object.
useAssay	Assay to use for PCA computation. If useAltExp is specified, useAssay has to exist in assays(altExp(inSCE, useAltExp)). Default "logcounts"
useAltExp	The subset to use for PCA computation, usually for the selected.variable features. Default NULL.
reducedDimName	Name to use for the reduced output assay. Default "PCA".
ndim	Number of principal components to obtain from the PCA computation. Default 50.
scale	Logical scalar, whether to standardize the expression values. Default TRUE.
ntop	Number of top features to use as a further variable feature selection. Default NULL.

Value

A [SingleCellExperiment](#) object with PCA computation updated in reducedDim(inSCE, reducedDimName).

Examples

```
data(scExample, package = "singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
logcounts(sce) <- log(counts(sce) + 1)
sce <- getPCA(sce, ntop = 500)
```

getSceParams	<i>Extract QC parameters from the SingleCellExperiment object</i>
--------------	---

Description

Extract QC parameters from the SingleCellExperiment object

Usage

```
getSceParams(
  inSCE,
  skip = c("scrublet", "runDecontX", "runBarcodeRanksMetaOutput"),
  ignore = c("algorithms", "estimates", "contamination", "z", "sample", "rank",
    "BPPARAM", "batch", "geneSetCollection", "barcodeArgs"),
  directory = "./",
  samplename = "",
  writeYAML = TRUE
)
```

Arguments

inSCE	A SingleCellExperiment object.
skip	Skip extracting the parameters of the provided QC functions.
ignore	Skip extracting the content within QC functions.
directory	The output directory of the SCTK_runQC.R pipeline.
samplename	The sample name of the SingleCellExperiment objects.
writeYAML	Whether output yaml file to store parameters. Default if TRUE. If FALSE, return character object.

Value

If writeYAML TRUE, a yaml object will be generated. If FALSE, character object.

getTopHVG	<i>getTopHVG Extracts the top variable genes from an input singleCell-Experiment object</i>
-----------	---

Description

getTopHVG Extracts the top variable genes from an input singleCellExperiment object

Usage

```
getTopHVG(inSCE, method, n = 2000)
```

Arguments

inSCE	an input <code>singleCellExperiment</code> object
method	represents which method to use for variable gene extraction from either Seurat "vst", "mean.var.plot", "dispersion" or Scrn "modelGeneVar"
n	number of top variable genes to extract

Value

list of top variable gene names

Author(s)

Irzam Sarfraz

Examples

```
data(sce_chcl, package = "scds")
sce_chcl <- scran_modelGeneVar(sce_chcl, "counts")
# return top 10 variable genes
topGenes <- getTopHVG(sce_chcl, "modelGeneVar", 10)
```

getTSNE

Run t-SNE dimensionality reduction method on a SingleCellExperiment Object

Description

Run t-SNE dimensionality reduction method on a `SingleCellExperiment` Object

Usage

```
getTSNE(
  inSCE,
  useAssay = "logcounts",
  useAltExp = NULL,
  reducedDimName = "TSNE",
  n_iterations = 1000,
  perplexity = NULL,
  run_pca = TRUE,
  ntop = NULL
)
```

Arguments

inSCE	Input SingleCellExperiment object.
useAssay	Assay to use for tSNE computation. If useAltExp is specified, useAssay has to exist in <code>assays(altExp(inSCE, useAltExp))</code> . Default "logcounts"
useAltExp	The subset to use for tSNE computation, usually for the selected.variable features. Default NULL.
reducedDimName	a name to store the results of the dimension reductions. Default "TSNE".

n_iterations	maximum iterations. Default 1000.
perplexity	perplexity parameter. Default NULL.
run_pca	run tSNE on PCA components? Default TRUE.
ntop	Number of top features to use as a further variable feature selection. Default NULL.

Value

A [SingleCellExperiment](#) object with tSNE computation updated in `reducedDim(inSCE, reducedDimName)`.

Examples

```
data("mouseBrainSubsetSCE")
#add a CPM assay
assay(mouseBrainSubsetSCE, "cpm") <- apply(
  assay(mouseBrainSubsetSCE, "counts"), 2, function(x) {
    x / (sum(x) / 1000000)
  })
mouseBrainSubsetSCE <- getTSNE(mouseBrainSubsetSCE, useAssay = "cpm",
  reducedDimName = "TSNE_cpm")
reducedDims(mouseBrainSubsetSCE)
```

getUMAP	<i>Uniform Manifold Approximation and Projection(UMAP) algorithm for dimension reduction.</i>
---------	---

Description

Uniform Manifold Approximation and Projection(UMAP) algorithm for dimension reduction.

Usage

```
getUMAP(
  inSCE,
  useAssay = "counts",
  useAltExp = NULL,
  sample = NULL,
  reducedDimName = "UMAP",
  logNorm = TRUE,
  nNeighbors = 30,
  nIterations = 200,
  alpha = 1,
  minDist = 0.01,
  spread = 1,
  pca = TRUE,
  initialDims = 50
)
```

Arguments

<code>inSCE</code>	Input SingleCellExperiment object.
<code>useAssay</code>	Assay to use for UMAP computation. If <code>useAltExp</code> is specified, <code>useAssay</code> has to exist in <code>assays(altExp(inSCE, useAltExp))</code> . Default "counts".
<code>useAltExp</code>	The subset to use for UMAP computation, usually for the selected.variable features. Default NULL.
<code>sample</code>	Character vector. Indicates which sample each cell belongs to. If given a single character, will take the annotation from <code>colData</code> . Default NULL.
<code>reducedDimName</code>	A name to store the results of the dimension reduction coordinates obtained from this method. Default "UMAP".
<code>logNorm</code>	Whether the counts will need to be log-normalized prior to generating the UMAP via <code>logNormCounts</code> . Default TRUE.
<code>nNeighbors</code>	The size of local neighborhood used for manifold approximation. Larger values result in more global views of the manifold, while smaller values result in more local data being preserved. Default 30. See <code>'?uwot::umap'</code> for more information.
<code>nIterations</code>	The number of iterations performed during layout optimization. Default is 200.
<code>alpha</code>	The initial value of "learning rate" of layout optimization. Default is 1.
<code>minDist</code>	The effective minimum distance between embedded points. Smaller values will result in a more clustered/clumped embedding where nearby points on the manifold are drawn closer together, while larger values will result on a more even dispersal of points. Default 0.01. See <code>'?uwot::umap'</code> for more information.
<code>spread</code>	The effective scale of embedded points. In combination with <code>minDist</code> , this determines how clustered/clumped the embedded points are. Default 1. See <code>'?uwot::umap'</code> for more information.
<code>pca</code>	Logical. Whether to perform dimension reduction with PCA before UMAP. Default TRUE
<code>initialDims</code>	Number of dimensions from PCA to use as input in UMAP. Default 50.

Value

A [SingleCellExperiment](#) object with UMAP computation updated in `reducedDim(inSCE, reducedDimName)`.

Examples

```
data(scExample, package = "singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
umap_res <- getUMAP(inSCE = sce, useAssay = "counts",
  reducedDimName = "UMAP", logNorm = TRUE,
  nNeighbors = 30, alpha = 1,
  nIterations = 200, spread = 1, pca = TRUE,
  initialDims = 50)
reducedDims(umap_res)
```

 gsvaSCE

 Run GSVA analysis on a *SingleCellExperiment* object

Description

Run GSVA analysis on a [SingleCellExperiment](#) object

Usage

```
gsvaSCE(inSCE, useAssay = "logcounts", pathwayNames, ...)
```

```
gsvaPlot(
  inSCE,
  gsvaData,
  plotType,
  condition = NULL,
  show_column_names = TRUE,
  show_row_names = TRUE,
  text_size = 12
)
```

Arguments

inSCE	Input SingleCellExperiment object.
useAssay	Indicate which assay to use. The default is "logcounts"
pathwayNames	List of pathway names to run, depending on pathwaySource parameter.
...	Parameters to pass to gsva()
gsvaData	GSVA data to plot. Required.
plotType	The type of plot to use, "Violin" or "Heatmap". Required.
condition	The condition(s) to use for the Violin plot, or the condition(s) to add as color bars above the Heatmap. Required for Violin, optional for Heatmap.
show_column_names	Display the column labels on the heatmap. The default is TRUE
show_row_names	Display the row labels on the heatmap. The default is TRUE.
text_size	Text size for plots. The default is 12

Value

gsvaSCE(): A data.frame of pathway activity scores from GSVA.

gsvaPlot(): The requested plot of the GSVA results.

Functions

- gsvaPlot: Plot GSVA results.
Plot GSVA Results

importAnnData	<i>Create a SingleCellExperiment Object from Python AnnData .h5ad files</i>
---------------	---

Description

This function reads in one or more Python AnnData files in the .h5ad format and returns a single [SingleCellExperiment](#) object containing all the AnnData samples by concatenating their counts matrices and related information slots.

Usage

```
importAnnData(sampleDirs = NULL, sampleNames = NULL, delayedArray = TRUE)
```

Arguments

sampleDirs	Folder containing the .h5ad file. Can be one of - <ul style="list-style-type: none"> • Default current working directory. • Full path to the directory containing the .h5ad file. E.g. sampleDirs = '/path/to/sample' • A vector of folder paths for the samples to import. E.g. sampleDirs = c('/path/to/sample1', '/path/to/sample2', '/path/to/sample3') importAnnData will return a single SCE object containing all the samples with the sample name appended to each colname in colData
sampleNames	The prefix/name of the .h5ad file without the .h5ad extension e.g. if 'sample.h5ad' is the filename, pass sampleNames = 'sample'. Can be one of - <ul style="list-style-type: none"> • Default sample. • A vector of samples to import. Length of vector must be equal to length of sampleDirs vector E.g. sampleDirs = c('sample1', 'sample2', 'sample3') importAnnData will return a single SCE object containing all the samples with the sample name appended to each colname in colData
delayedArray	Boolean. Whether to read the expression matrix as DelayedArray object. Default TRUE.

Details

importAnnData converts scRNA-seq data in the AnnData format to the SingleCellExperiment object. The .X slot in AnnData is transposed to the features x cells format and becomes the 'counts' matrix in the assay slot. The .vars AnnData slot becomes the SCE rowData and the .obs AnnData slot becomes the SCE colData. Multidimensional data in the .obsm AnnData slot is ported over to the SCE reducedDims slot. Additionally, unstructured data in the .uns AnnData slot is available through the SCE metadata slot. There are 2 currently known minor issues - Anndata python module depends on another python module h5py to read hd5 format files. If there are errors reading the .h5ad files, such as "ValueError: invalid shape in fixed-type tuple." the user will need to do downgrade h5py by running `pip3 install --user h5py==2.9.0` Additionally there might be errors in converting some python objects in the unstructured data slots. There are no known R solutions at present. Refer <https://github.com/rstudio/reticulate/issues/209>

Value

A SingleCellExperiment object.

Examples

```
file.path <- system.file("extdata/annData_pbmc_3k", package = "singleCellTK")
## Not run:
sce <- importAnnData(sampleDirs = file.path,
                    sampleNames = 'pbmc3k_20by20')

## End(Not run)
```

importBUStools	<i>Construct SCE object from BUStools output</i>
----------------	--

Description

Read the barcodes, features (genes), and matrix from BUStools output. Import them as one [Single-CellExperiment](#) object. Note the cells in the output files for BUStools 0.39.4 are not filtered.

Usage

```
importBUStools(
  BUStoolsDirs,
  samples,
  matrixFileNames = "genes.mtx",
  featuresFileNames = "genes.genes.txt",
  barcodesFileNames = "genes.barcodes.txt",
  gzipped = "auto",
  class = c("Matrix", "matrix"),
  delayedArray = TRUE
)
```

Arguments

BUStoolsDirs	A vector of paths to BUStools output files. Each sample should have its own path. For example: ./genecount. Must have the same length as samples.
samples	A vector of user-defined sample names for the samples to be imported. Must have the same length as BUStoolsDirs.
matrixFileNames	Filenames for the Market Exchange Format (MEX) sparse matrix files (.mtx files). Must have length 1 or the same length as samples.
featuresFileNames	Filenames for the feature annotation files. Must have length 1 or the same length as samples.
barcodesFileNames	Filenames for the cell barcode list file. Must have length 1 or the same length as samples.
gzipped	Boolean. TRUE if the BUStools output files (barcodes.txt, genes.txt, and genes.mtx) were gzip compressed. FALSE otherwise. This is FALSE in BUStools 0.39.4. Default "auto" which automatically detects if the files are gzip compressed. Must have length 1 or the same length as samples.

class	Character. The class of the expression matrix stored in the SCE object. Can be one of "Matrix" (as returned by readMM function), or "matrix" (as returned by matrix function). Default "Matrix".
delayedArray	Boolean. Whether to read the expression matrix as DelayedArray-class object or not. Default TRUE.

Value

A `SingleCellExperiment` object containing the count matrix, the gene annotation, and the cell annotation.

Examples

```
# Example #1
# FASTQ files were downloaded from
# https://support.10xgenomics.com/single-cell-gene-expression/datasets/3.0.0
# /pbmc_1k_v3
# They were concatenated as follows:
# cat pbmc_1k_v3_S1_L001_R1_001.fastq.gz pbmc_1k_v3_S1_L002_R1_001.fastq.gz >
# pbmc_1k_v3_R1.fastq.gz
# cat pbmc_1k_v3_S1_L001_R2_001.fastq.gz pbmc_1k_v3_S1_L002_R2_001.fastq.gz >
# pbmc_1k_v3_R2.fastq.gz
# The following BUStools command generates the gene, cell, and
# matrix files

# bustools correct -w ./3M-february-2018.txt -p output.bus | \
# bustools sort -T tmp/ -t 4 -p - | \
# bustools count -o genecount/genes \
# -g ./transcripts_to_genes.txt \
# -e matrix.ec \
# -t transcripts.txt \
# --genecounts -

# The top 20 genes and the first 20 cells are included in this example.
sce <- importBUStools(
  BUStoolsDirs = system.file("extdata/BUStools_PBMC_1k_v3_20x20/genecount/"),
  package = "singleCellTK"),
  samples = "PBMC_1k_v3_20x20")
```

importCellRanger	<i>Construct SCE object from Cell Ranger output</i>
------------------	---

Description

Read the filtered barcodes, features, and matrices for all samples from (preferably a single run of) Cell Ranger output. Import and combine them as one big [SingleCellExperiment](#) object.

Usage

```
importCellRanger(
  cellRangerDirs = NULL,
  sampleDirs = NULL,
  sampleNames = NULL,
```

```

    cellRangerOuts = NULL,
    dataType = c("filtered", "raw"),
    matrixFileNames = "matrix.mtx.gz",
    featuresFileNames = "features.tsv.gz",
    barcodesFileNames = "barcodes.tsv.gz",
    gzipped = "auto",
    class = c("Matrix", "matrix"),
    delayedArray = TRUE
)

importCellRangerV2(
  cellRangerDirs = NULL,
  sampleDirs = NULL,
  sampleNames = NULL,
  dataTypeV2 = c("filtered", "raw"),
  class = c("Matrix", "matrix"),
  delayedArray = TRUE,
  reference = NULL,
  cellRangerOutsV2 = NULL
)

importCellRangerV3(
  cellRangerDirs = NULL,
  sampleDirs = NULL,
  sampleNames = NULL,
  dataType = c("filtered", "raw"),
  class = c("Matrix", "matrix"),
  delayedArray = TRUE
)

```

Arguments

cellRangerDirs The root directories where Cell Ranger was run. These folders should contain sample specific folders. Default NULL, meaning the paths for each sample will be specified in *samples* argument.

sampleDirs Default NULL. Can be one of

- NULL. All samples within cellRangerDirs will be imported. The order of samples will be first determined by the order of cellRangerDirs and then by [list.dirs](#). This is only for the case where cellRangerDirs is specified.
- A list of vectors containing the folder names for samples to import. Each vector in the list corresponds to samples from one of cellRangerDirs. These names are the same as the folder names under cellRangerDirs. This is only for the case where cellRangerDirs is specified.
- A vector of folder paths for the samples to import. This is only for the case where cellRangerDirs is NULL.

The cells in the final SCE object will be ordered in the same order of sampleDirs.

sampleNames A vector of user-defined sample names for the samples to be imported. Must have the same length as `length(unlist(sampleDirs))` if sampleDirs is not NULL. Otherwise, make sure the length and order match the output of `unlist(lapply(cellRangerDir, FUN = FALSE))`. Default NULL, in which case the folder names will be used as sample names.

cellRangerOuts	Character vector. The intermediate paths to filtered or raw cell barcode, feature, and matrix files for each sample. Supersedes <code>dayaType</code> . If NULL, <code>dayaType</code> will be used to determine Cell Ranger output directory. If not NULL, <code>dayaType</code> will be ignored and <code>cellRangerOuts</code> specifies the paths. Must have length 1 or the same length as <code>length(unlist(sampleDirs))</code> if <code>sampleDirs</code> is not NULL. Otherwise, make sure the length and order match the output of <code>unlist(lapply(cellRangerDirs, list.dirs, recursive = FALSE))</code> . Reference genome names might need to be appended for CellRanger version below 3.0.0 if reads were mapped to multiple genomes when running Cell Ranger pipeline. Probable options include "outs/filtered_feature_bc_matrix/", "outs/raw_feature_bc_matrix/", "outs/filtered_gene_bc_matrix/", "outs/raw_gene_bc_matrix/".
dayaType	Character. The type of data to import. Can be one of "filtered" (which is equivalent to <code>cellRangerOuts = "outs/filtered_feature_bc_matrix/"</code> or <code>cellRangerOuts = "outs/filtered_gene_bc_matrix/"</code>) or "raw" (which is equivalent to <code>cellRangerOuts = "outs/raw_feature_bc_matrix/"</code> or <code>cellRangerOuts = "outs/raw_gene_bc_matrix/"</code>). Default "filtered" which imports the counts for filtered cell barcodes only.
matrixFileNames	Character vector. Filenames for the Market Exchange Format (MEX) sparse matrix files (matrix.mtx or matrix.mtx.gz files). Must have length 1 or the same length as <code>length(unlist(sampleDirs))</code> if <code>sampleDirs</code> is not NULL. Otherwise, make sure the length and order match the output of <code>unlist(lapply(cellRangerDirs, list.dirs, recursive = FALSE))</code> .
featuresFileNames	Character vector. Filenames for the feature annotation files. They are usually named <i>features.tsv.gz</i> or <i>genes.tsv</i> . Must have length 1 or the same length as <code>length(unlist(sampleDirs))</code> if <code>sampleDirs</code> is not NULL. Otherwise, make sure the length and order match the output of <code>unlist(lapply(cellRangerDirs, list.dirs, recursive = FALSE))</code> .
barcodesFileNames	Character vector. Filename for the cell barcode list files. They are usually named <i>barcodes.tsv.gz</i> or <i>barcodes.tsv</i> . Must have length 1 or the same length as <code>length(unlist(sampleDirs))</code> if <code>sampleDirs</code> is not NULL. Otherwise, make sure the length and order match the output of <code>unlist(lapply(cellRangerDirs, list.dirs, recursive = FALSE))</code> .
gzipped	TRUE if the Cell Ranger output files (barcodes.tsv, features.tsv, and matrix.mtx) were gzip compressed. FALSE otherwise. This is true after Cell Ranger 3.0.0 update. Default "auto" which automatically detects if the files are gzip compressed. If not "auto", <code>gzipped</code> must have length 1 or the same length as <code>length(unlist(sampleDirs))</code> if <code>sampleDirs</code> is not NULL. Otherwise, make sure the length and order match the output of <code>unlist(lapply(cellRangerDirs, list.dirs, recursive = FALSE))</code> .
class	Character. The class of the expression matrix stored in the SCE object. Can be one of "Matrix" (as returned by readMM function), or "matrix" (as returned by matrix function). Default "Matrix".
delayedArray	Boolean. Whether to read the expression matrix as DelayedArray object or not. Default TRUE.
dayaTypeV2	Character. The type of output to import for Cellranger version below 3.0.0. Whether to import the filtered or the raw data. Can be one of 'filtered' or 'raw'. Default 'filtered'. When <code>cellRangerOuts</code> is specified, <code>dayaTypeV2</code> and reference will be ignored.

reference Character vector. The reference genome names. Default NULL. If not NULL, it must give the length and order as `length(unlist(sampleDirs))` if `sampleDirs` is not NULL. Otherwise, make sure the length and order match the output of `unlist(lapply(cellRangerDirs,list.dirs,recursive = FALSE))`. Only needed for Cellranger version below 3.0.0.

cellRangerOutsV2 Character vector. The intermediate paths to filtered or raw cell barcode, feature, and matrix files for each sample for Cellranger version below 3.0.0. If NULL, `reference` and `dataTypeV2` will be used to determine Cell Ranger output directory. If it has length 1, it assumes that all samples use the same genome reference and the function will load only filtered or raw data.

Details

`importCellRangerV2` imports output from Cell Ranger V2. `importCellRangerV2Sample` imports output from one sample from Cell Ranger V2. `importCellRangerV3` imports output from Cell Ranger V3. `importCellRangerV3Sample` imports output from one sample from Cell Ranger V3. Some implicit assumptions which match the output structure of Cell Ranger V2 & V3 are made in these 4 functions including `cellRangerOuts`, `matrixFileName`, `featuresFileName`, `barcodesFileName`, and `gzipped`. Alternatively, user can call `importCellRanger` to explicitly specify these arguments.

Value

A `SingleCellExperiment` object containing the combined count matrix, the feature annotations, and the cell annotation.

Examples

```
# Example #1
# The following filtered feature, cell, and matrix files were downloaded from
# https://support.10xgenomics.com/single-cell-gene-expression/datasets/
# 3.0.0/hgmm_1k_v3
# The top 10 hg19 & mm10 genes are included in this example.
# Only the first 20 cells are included.
sce <- importCellRanger(
  cellRangerDirs = system.file("extdata/", package = "singleCellTK"),
  sampleDirs = "hgmm_1k_v3_20x20",
  sampleNames = "hgmm1kv3",
  dataType = "filtered")
# The following filtered feature, cell, and matrix files were downloaded from
# https://support.10xgenomics.com/single-cell-gene-expression/datasets/
# 2.1.0/pbmc4k
# Top 20 genes are kept. 20 cell barcodes are extracted.
sce <- importCellRangerV2(
  cellRangerDirs = system.file("extdata/", package = "singleCellTK"),
  sampleDirs = "pbmc_4k_v2_20x20",
  sampleNames = "pbmc4k_20",
  reference = 'GRCh38',
  dataTypeV2 = "filtered")
sce <- importCellRangerV3(
  cellRangerDirs = system.file("extdata/", package = "singleCellTK"),
  sampleDirs = "hgmm_1k_v3_20x20",
  sampleNames = "hgmm1kv3",
  dataType = "filtered")
```

```
importCellRangerV2Sample
```

Construct SCE object from Cell Ranger V2 output for a single sample

Description

Read the filtered barcodes, features, and matrices for all samples from Cell Ranger V2 output. Files are assumed to be named "matrix.mtx", "genes.tsv", and "barcodes.tsv".

Usage

```
importCellRangerV2Sample(  
  dataDir = NULL,  
  sampleName = NULL,  
  class = c("Matrix", "matrix"),  
  delayedArray = TRUE  
)
```

Arguments

dataDir	A path to the directory containing the data files. Default ".".
sampleName	A User-defined sample name. This will be prepended to all cell barcode IDs. Default "sample".
class	Character. The class of the expression matrix stored in the SCE object. Can be one of "Matrix" (as returned by readMM function), or "matrix" (as returned by matrix function). Default "Matrix".
delayedArray	Boolean. Whether to read the expression matrix as DelayedArray object or not. Default TRUE.

Value

A `SingleCellExperiment` object containing the count matrix, the feature annotations, and the cell annotation for the sample.

Examples

```
sce <- importCellRangerV2Sample(  
  dataDir = system.file("extdata/pbmc_4k_v2_20x20/outs/",  
    "filtered_gene_bc_matrices/GRCh38", package = "singleCellTK"),  
  sampleName = "pbmc4k_20")
```

`importCellRangerV3Sample`*Construct SCE object from Cell Ranger V3 output for a single sample*

Description

Read the filtered barcodes, features, and matrices for all samples from Cell Ranger V3 output. Files are assumed to be named "matrix.mtx.gz", "features.tsv.gz", and "barcodes.tsv.gz".

Usage

```
importCellRangerV3Sample(  
  dataDir = "./",  
  sampleName = "sample",  
  class = c("Matrix", "matrix"),  
  delayedArray = TRUE  
)
```

Arguments

<code>dataDir</code>	A path to the directory containing the data files. Default ".".
<code>sampleName</code>	A User-defined sample name. This will be prepended to all cell barcode IDs. Default "sample".
<code>class</code>	Character. The class of the expression matrix stored in the SCE object. Can be one of "Matrix" (as returned by readMM function), or "matrix" (as returned by matrix function). Default "Matrix".
<code>delayedArray</code>	Boolean. Whether to read the expression matrix as DelayedArray object or not. Default TRUE.

Value

A `SingleCellExperiment` object containing the count matrix, the feature annotations, and the cell annotation for the sample.

Examples

```
sce <- importCellRangerV3Sample(  
  dataDir = system.file("extdata/hgmm_1k_v3_20x20/outs/",  
    "filtered_feature_bc_matrix", package = "singleCellTK"),  
  sampleName = "hgmm1kv3")
```

<code>importDropEst</code>	<i>Create a SingleCellExperiment Object from DropEst output</i>
----------------------------	---

Description

imports the RDS file created by DropEst (<https://github.com/hms-dbmi/dropEst>) and create a `SingleCellExperiment` object from either the raw or filtered counts matrix. Additionally parse through the RDS to obtain appropriate feature annotations as SCE coldata, in addition to any metadata.

Usage

```
importDropEst(
  sampleDirs = NULL,
  dataType = c("filtered", "raw"),
  rdsFileName = "cell.counts",
  sampleNames = NULL,
  delayedArray = TRUE
)
```

Arguments

<code>sampleDirs</code>	A path to the directory containing the data files. Default <code>"/"</code> .
<code>dataType</code>	can be <code>"filtered"</code> or <code>"raw"</code> . Default <code>"filtered"</code> .
<code>rdsFileName</code>	File name prefix of the DropEst RDS output. default is <code>"cell.counts"</code>
<code>sampleNames</code>	A User-defined sample name. This will be prepended to all cell barcode IDs. Default <code>"sample"</code> .
<code>delayedArray</code>	Boolean. Whether to read the expression matrix as DelayedArray object or not. Default <code>TRUE</code> .

Details

`importDropEst` expects either raw counts matrix stored as `"cm_raw"` or filtered counts matrix stored as `"cm"` in the DropEst rds output. `ColData` is obtained from the DropEst corresponding to `"mean_reads_per_umi", "aligned_reads_per_cell", "aligned_umis_per_cell", "requested_umis_per_cb", "requested_reads"`. If using filtered counts matrix, the `colData` dataframe is subset to contain features from the filtered counts matrix alone. If any annotations of (`"saturation_info", "merge_targets", "reads_per_umi_per_cell"`) are found in the DropEst rds, they will be added to the SCE metadata field

Value

A `SingleCellExperiment` object containing the count matrix, the feature annotations from DropEst as `ColData`, and any metadata from DropEst

Examples

```
# Example results were generated as per instructions from the developers of dropEst described in
# https://github.com/hms-dbmi/dropEst/blob/master/examples/EXAMPLES.md
sce <- importDropEst(sampleDirs = system.file("extdata/dropEst_scg71", package = "singleCellTK"),
  sampleNames = 'scg71')
```

importExampleData *Retrieve example datasets*

Description

Retrieves published example datasets stored in [SingleCellExperiment](#) using the [scRNAseq](#) and [TENxPBMCDData](#) packages. See 'Details' for a list of available datasets.

Usage

```
importExampleData(dataset, class = c("Matrix", "matrix"), delayedArray = FALSE)
```

Arguments

dataset	Character. Name of the dataset to retrieve.
class	Character. The class of the expression matrix stored in the SCE object. Can be one of "Matrix" or "matrix". "Matrix" will store the data as a sparse matrix from package Matrix while "matrix" will store the data in a standard matrix. Default "Matrix".
delayedArray	Boolean. Whether to read the expression matrix as DelayedArray object or not. Default FALSE.

Details

See the list below for the available datasets and their descriptions.

"fluidigm_pollen" Retrieved with [ReprocessedFluidigmData](#). Returns a dataset of 65 human neural cells from Pollen et al. (2014), each sequenced at high and low coverage (SRA accession SRP041736).

"allen_tasic" Retrieved with [ReprocessedAllenData](#). Returns a dataset of 379 mouse brain cells from Tasic et al. (2016).

"pbmc3k" Retrieved with [TENxPBMCDData](#). 2,700 peripheral blood mononuclear cells (PBMCs) from 10X Genomics.

"pbmc4k" Retrieved with [TENxPBMCDData](#). 4,340 peripheral blood mononuclear cells (PBMCs) from 10X Genomics.

"pbmc6k" Retrieved with [TENxPBMCDData](#). 5,419 peripheral blood mononuclear cells (PBMCs) from 10X Genomics.

"pbmc8k" Retrieved with [TENxPBMCDData](#). 8,381 peripheral blood mononuclear cells (PBMCs) from 10X Genomics.

"pbmc33k" Retrieved with [TENxPBMCDData](#). 33,148 peripheral blood mononuclear cells (PBMCs) from 10X Genomics.

"pbmc68k" Retrieved with [TENxPBMCDData](#). 68,579 peripheral blood mononuclear cells (PBMCs) from 10X Genomics.

Value

The specified [SingleCellExperiment](#) object.

Author(s)

Joshua D. Campbell, David Jenkins

Examples

```
sce <- importExampleData("pbmc3k")
```

<code>importFromFiles</code>	<i>Create a SingleCellExperiment object from files</i>
------------------------------	--

Description

Creates a `SingleCellExperiment` object from a counts file in various formats. and a file of annotation information, .

Usage

```
importFromFiles(
  assayFile,
  annotFile = NULL,
  featureFile = NULL,
  assayName = "counts",
  inputDataFrames = FALSE,
  class = c("Matrix", "matrix"),
  delayedArray = FALSE,
  annotFileHeader = FALSE,
  annotFileRowName = 1,
  annotFileSep = "\t",
  featureHeader = FALSE,
  featureRowName = 1,
  featureSep = "\t",
  gzipped = "auto"
)
```

Arguments

<code>assayFile</code>	The path to a file in .mtx, .txt, .csv, .tab, or .tsv format.
<code>annotFile</code>	The path to a text file that contains columns of annotation information for each sample in the <code>assayFile</code> . This file should have the same number of rows as there are columns in the <code>assayFile</code> . If multiple samples are represented in these files, this should be denoted by a column called 'sample' within the <code>annotFile</code> .
<code>featureFile</code>	The path to a text file that contains columns of annotation information for each gene in the count matrix. This file should have the same genes in the same order as <code>assayFile</code> . This is optional.
<code>assayName</code>	The name of the assay that you are uploading. The default is "counts".
<code>inputDataFrames</code>	If TRUE, <code>assayFile</code> and <code>annotFile</code> are read as data frames instead of file paths. The default is FALSE.

class	Character. The class of the expression matrix stored in the SCE object. Can be one of "Matrix" (as returned by readMM function), or "matrix" (as returned by matrix function). Default "Matrix".
delayedArray	Boolean. Whether to read the expression matrix as DelayedArray object or not. Default TRUE.
annotFileHeader	Whether there's a header (colnames) in the cell annotation file. Default is FALSE
annotFileRowName	Which column is used as the rownames for the cell annotation file. Default is 1 (first column).
annotFileSep	Separator used for the cell annotation file. Default is "\t".
featureHeader	Whether there's a header (colnames) in the feature annotation file. Default is FALSE
featureRowName	Which column is used as the rownames for the feature annotation file. Default is 1 (first column).
featureSep	Separator used for the feature annotation file. Default is "\t".
gzipped	Whether the input file is gzipped. Default is "auto" and it will automatically detect whether the file is gzipped. Other options is TRUE or FALSE.

Value

a [SingleCellExperiment](#) object

importGeneSetsFromCollection

Imports gene sets from a [GeneSetCollection](#) object

Description

Converts a list of gene sets stored in a [GeneSetCollection](#) object and stores it in the metadata of the [SingleCellExperiment](#) object. These gene sets can be used in downstream quality control and analysis functions in [singleCellTK](#).

Usage

```
importGeneSetsFromCollection(
  inSCE,
  geneSetCollection,
  collectionName = "GeneSetCollection",
  by = "rownames"
)
```

Arguments

inSCE Input [SingleCellExperiment](#) object.

geneSetCollection

A [GeneSetCollection](#) object. See [GeneSetCollection](#) for more details.

collectionName Character. Name of collection to add gene sets to. If this collection already exists in `inSCE`, then these gene sets will be added to that collection. Any gene sets within the collection with the same name will be overwritten. Default `GeneSetCollection`.

by Character, character vector, or `NULL`. Describes the location within `inSCE` where the gene identifiers in `geneSetCollection` should be mapped. If set to `"rownames"` then the features will be searched for among `rownames(inSCE)`. This can also be set to one of the column names of `rowData(inSCE)` in which case the gene identifiers will be mapped to that column in the `rowData` of `inSCE`. `by` can be a vector the same length as the number of gene sets in the `GeneSetCollection` and the elements of the vector can point to different locations within `inSCE`. Finally, `by` can be `NULL`. In this case, the location of the gene identifiers in `inSCE` should be saved in the description slot for each gene set in the `GeneSetCollection`. See [featureIndex](#) for more information. Default `"rownames"`.

Details

The gene identifiers in gene sets in the `GeneSetCollection` will be mapped to the `rownames` of `inSCE` using the `by` parameter and stored in a [GeneSetCollection](#) object from package [GSEABase](#). This object is stored in `metadata(inSCE)$sctk$genesets`, which can be accessed in downstream analysis functions such as [runCellQC](#).

Value

A [SingleCellExperiment](#) object with gene set from `collectionName` output stored to the `metadata` slot.

Author(s)

Joshua D. Campbell

See Also

[importGeneSetsFromList](#) for importing from lists, [importGeneSetsFromGMT](#) for importing from GMT files, and [importGeneSetsFromMSigDB](#) for importing MSigDB gene sets.

Examples

```
data(scExample)
library(GSEABase)
gs1 <- GeneSet(setName = "geneset1", geneIds = rownames(sce)[1:10])
gs2 <- GeneSet(setName = "geneset2", geneIds = rownames(sce)[11:20])
gsc <- GeneSetCollection(list(gs1, gs2))
sce <- importGeneSetsFromCollection(inSCE = sce,
                                  geneSetCollection = gsc,
                                  by = "rownames")
```

importGeneSetsFromGMT *Imports gene sets from a GMT file*

Description

Converts a list of gene sets stored in a GMT file into a [GeneSetCollection](#) and stores it in the metadata of the [SingleCellExperiment](#) object. These gene sets can be used in downstream quality control and analysis functions in [singleCellTK](#).

Usage

```
importGeneSetsFromGMT(
  inSCE,
  file,
  collectionName = "GeneSetCollection",
  by = "rownames",
  sep = "\t"
)
```

Arguments

inSCE	Input SingleCellExperiment object.
file	Character. Path to GMT file. See getGmt for more information on reading GMT files.
collectionName	Character. Name of collection to add gene sets to. If this collection already exists in inSCE, then these gene sets will be added to that collection. Any gene sets within the collection with the same name will be overwritten. Default GeneSetCollection .
by	Character, character vector, or NULL. Describes the location within inSCE where the gene identifiers in geneSetList should be mapped. If set to "rownames" then the features will be searched for among rownames(inSCE). This can also be set to one of the column names of rowData(inSCE) in which case the gene identifiers will be mapped to that column in the rowData of inSCE. by can be a vector the same length as the number of gene sets in the GMT file and the elements of the vector can point to different locations within inSCE. Finally, by can be NULL. In this case, the location of the gene identifiers in inSCE should be saved in the description (2nd column) of the GMT file. See featureIndex for more information. Default "rownames".
sep	Character. Delimiter of the GMT file. Default "\t".

Details

The gene identifiers in gene sets in the GMT file will be mapped to the rownames of inSCE using the by parameter and stored in a [GeneSetCollection](#) object from package [GSEABase](#). This object is stored in metadata(inSCE)\$sctk\$genesets, which can be accessed in downstream analysis functions such as [runCellQC](#).

Value

A [SingleCellExperiment](#) object with gene set from collectionName output stored to the [metadata](#) slot.

Author(s)

Joshua D. Campbell

See Also

[importGeneSetsFromList](#) for importing from lists, [importGeneSetsFromCollection](#) for importing from [GeneSetCollection](#) objects, and [importGeneSetsFromMSigDB](#) for importing MSigDB gene sets.

Examples

```
data(scExample)

# GMT file containing gene symbols for a subset of human mitochondrial genes
gmt <- system.file("extdata/mito_subset.gmt", package = "singleCellTK")

# "feature_name" is the second column in the GMT file, so the ids will
# be mapped using this column in the 'rowData' of 'sce'. This
# could also be accomplished by setting by = "feature_name" in the
# function call.
sce <- importGeneSetsFromGMT(inSCE = sce, file = gmt, by = NULL)
```

```
importGeneSetsFromList
```

Imports gene sets from a list

Description

Converts a list of gene sets into a [GeneSetCollection](#) and stores it in the metadata of the [SingleCellExperiment](#) object. These gene sets can be used in downstream quality control and analysis functions in [singleCellTK](#).

Usage

```
importGeneSetsFromList(
  inSCE,
  geneSetList,
  collectionName = "GeneSetCollection",
  by = "rownames"
)
```

Arguments

<code>inSCE</code>	Input SingleCellExperiment object.
<code>geneSetList</code>	Named List. A list containing one or more gene sets. Each element of the list should be a character vector of gene identifiers. The names of the list will become the gene set names in the GeneSetCollection object.
<code>collectionName</code>	Character. Name of collection to add gene sets to. If this collection already exists in <code>inSCE</code> , then these gene sets will be added to that collection. Any gene sets within the collection with the same name will be overwritten. Default <code>GeneSetCollection</code> .

by Character or character vector. Describes the location within `inSCE` where the gene identifiers in `geneSetList` should be mapped. If set to "rownames" then the features will be searched for among `rownames(inSCE)`. This can also be set to one of the column names of `rowData(inSCE)` in which case the gene identifiers will be mapped to that column in the `rowData` of `inSCE`. Finally, `by` can be a vector the same length as the number of gene sets in `geneSetList` and the elements of the vector can point to different locations within `inSCE`. See [featureIndex](#) for more information. Default "rownames".

Details

The gene identifiers in gene sets in `geneSetList` will be mapped to the `rownames` of `inSCE` using the `by` parameter and stored in a [GeneSetCollection](#) object from package [GSEABase](#). This object is stored in `metadata(inSCE)$sctk$genesets`, which can be accessed in downstream analysis functions such as [runCellQC](#).

Value

A [SingleCellExperiment](#) object with gene set from `collectionName` output stored to the `metadata` slot.

Author(s)

Joshua D. Campbell

See Also

[importGeneSetsFromCollection](#) for importing from [GeneSetCollection](#) objects, [importGeneSetsFromGMT](#) for importing from GMT files, and [importGeneSetsFromMSigDB](#) for importing MSigDB gene sets.

Examples

```
data(scExample)

# Generate gene sets from 'rownames'
gs1 <- rownames(sce)[1:10]
gs2 <- rownames(sce)[11:20]
gs <- list("geneset1" = gs1, "geneset2" = gs2)
sce <- importGeneSetsFromList(inSCE = sce,
                             geneSetList = gs,
                             by = "rownames")

# Generate a gene set for mitochondrial genes using
# Gene Symbols stored in 'rowData'
mito.ix <- grep("^MT-", rowData(sce)$feature_name)
mito <- list(mito = rowData(sce)$feature_name[mito.ix])
sce <- importGeneSetsFromList(inSCE = sce,
                             geneSetList = mito,
                             by = "feature_name")
```

```
importGeneSetsFromMSigDB
```

Imports gene sets from MSigDB

Description

Gets a list of MSigDB gene sets stores it in the metadata of the [SingleCellExperiment](#) object. These gene sets can be used in downstream quality control and analysis functions in [singleCellTK](#).

Usage

```
importGeneSetsFromMSigDB(
  inSCE,
  categoryIDs,
  species = "Homo sapiens",
  mapping = c("gene_symbol", "human_gene_symbol", "entrez_gene"),
  by = "rownames",
  verbose = TRUE
)
```

Arguments

inSCE	Input SingleCellExperiment object.
categoryIDs	Character vector containing the MSigDB gene set ids. The column ID in the table returned by <code>getMSigDBTable()</code> shows the list of possible gene set IDs that can be obtained.
species	Character. Species available can be found using the function msigdb_show_species . Default "Homo sapiens".
mapping	Character. One of "gene_symbol", "human_gene_symbol", or "entrez_gene". Gene identifiers to be used for MSigDB gene sets. IDs denoted by the by parameter must be either in gene symbol or Entrez gene id format to match IDs from MSigDB.
by	Character. Describes the location within inSCE where the gene identifiers in the MSigDB gene sets should be mapped. If set to "rownames" then the features will be searched for among <code>rownames(inSCE)</code> . This can also be set to one of the column names of <code>rowData(inSCE)</code> in which case the gene identifiers will be mapped to that column in the <code>rowData</code> of inSCE. See featureIndex for more information. Default "rownames".
verbose	Boolean. Whether to display progress. Default TRUE.

Details

The gene identifiers in gene sets from MSigDB will be retrieved using the [msigdb](#) package. They will be mapped to the IDs in inSCE using the by parameter and stored in a [GeneSetCollection](#) object from package [GSEABase](#). This object is stored in `metadata(inSCE)$sctk$genesets`, which can be accessed in downstream analysis functions such as [runCellQC](#).

Value

A [SingleCellExperiment](#) object with gene set from collectionName output stored to the `metadata` slot.

Author(s)

Joshua D. Campbell

See Also

[importGeneSetsFromList](#) for importing from lists, [importGeneSetsFromGMT](#) for importing from GMT files, and [GeneSetCollection](#) objects.

Examples

```
data(scExample)
sce <- importGeneSetsFromMSigDB(inSCE = sce,
                                categoryIDs = "H",
                                species = "Homo sapiens",
                                mapping = "gene_symbol",
                                by = "feature_name")
```

`importMultipleSources` *Imports samples from different sources and compiles them into a list of SCE objects*

Description

Imports samples from different sources and compiles them into a list of SCE objects

Usage

```
importMultipleSources(allImportEntries, delayedArray = FALSE)
```

Arguments

<code>allImportEntries</code>	object containing the sources and parameters of all the samples being imported (from the UI)
<code>delayedArray</code>	Boolean. Whether to read the expression matrix as DelayedArray object or not. Default TRUE.

Value

A list of [SingleCellExperiment](#) object containing the droplet or cell data or both, depending on the `dataType` that users provided.

importOptimus	<i>Construct SCE object from Optimus output</i>
---------------	---

Description

Read the barcodes, features (genes), and matrices from Optimus outputs. Import them as one [SingleCellExperiment](#) object.

Usage

```
importOptimus(
  OptimusDirs,
  samples,
  matrixLocation = "call-MergeCountFiles/sparse_counts.npz",
  colIndexLocation = "call-MergeCountFiles/sparse_counts_col_index.npy",
  rowIndexLocation = "call-MergeCountFiles/sparse_counts_row_index.npy",
  cellMetricsLocation = "call-MergeCellMetrics/merged-cell-metrics.csv.gz",
  geneMetricsLocation = "call-MergeGeneMetrics/merged-gene-metrics.csv.gz",
  emptyDropsLocation = "call-RunEmptyDrops/empty_drops_result.csv",
  class = c("Matrix", "matrix"),
  delayedArray = TRUE
)
```

Arguments

OptimusDirs	A vector of root directories of Optimus output files. The paths should be something like this: /PATH/T0/bb4a2a5e-ff34-41b6-97d2-0c0c0c534530. Each entry in OptimusDirs is considered a sample and should have its own path. Must have the same length as samples.
samples	A vector of user-defined sample names for the sample to be imported. Must have the same length as OptimusDirs.
matrixLocation	Character. It is the intermediate path to the filtered count matrix file saved in sparse matrix format (.npz). Default call-MergeCountFiles/sparse_counts.npz which works for optimus_v1.4.0.
colIndexLocation	Character. The intermediate path to the barcode index file. Default call-MergeCountFiles/sparse_
rowIndexLocation	Character. The intermediate path to the feature (gene) index file. Default call-MergeCountFiles/sp
cellMetricsLocation	Character. It is the intermediate path to the cell metrics file (merged-cell-metrics.csv.gz). Default call-MergeCellMetrics/merged-cell-metrics.csv.gz which works for optimus_v1.4.0.
geneMetricsLocation	Character. It is the intermediate path to the feature (gene) metrics file (merged-gene-metrics.csv.g). Default call-MergeGeneMetrics/merged-gene-metrics.csv.gz which works for optimus_v1.4.0.
emptyDropsLocation	Character. It is the intermediate path to emptyDrops metrics file (empty_drops_result.csv). Default call-RunEmptyDrops/empty_drops_result.csv which works for optimus_v1.4.0.

class	Character. The class of the expression matrix stored in the SCE object. Can be one of "Matrix" (as returned by readMM function), or "matrix" (as returned by matrix function). Default "Matrix".
delayedArray	Boolean. Whether to read the expression matrix as DelayedArray object or not. Default TRUE.

Value

A [SingleCellExperiment](#) object containing the count matrix, the gene annotation, and the cell annotation.

Examples

```
file.path <- system.file("extdata/Optimus_20x1000",
  package = "singleCellTK")
## Not run:
sce <- importOptimus(OptimusDirs = file.path,
  samples = "Optimus_20x1000")

## End(Not run)
```

importSEQC

Construct SCE object from seqc output

Description

Read the filtered barcodes, features, and matrices for all samples from (preferably a single run of) seqc output. Import and combine them as one big [SingleCellExperiment](#) object.

Usage

```
importSEQC(
  seqcDirs = NULL,
  samples = NULL,
  prefix = NULL,
  gzipped = FALSE,
  class = c("Matrix", "matrix"),
  delayedArray = TRUE,
  cbNotFirstCol = TRUE,
  feNotFirstCol = TRUE,
  combinedSample = TRUE
)
```

Arguments

seqcDirs	A vector of paths to seqc output files. Each sample should have its own path. For example: ./pbmc_1k_50x50. Must have the same length as samples.
samples	A vector of user-defined sample names for the samples to be imported. Must have the same length as seqcDirs.
prefix	A vector containing the prefix of file names within each sample directory. It cannot be null and the vector should have the same length as <i>samples</i> .

<code>gzipped</code>	Boolean. TRUE if the seqc output files (<code>sparse_counts_barcode.csv</code> , <code>sparse_counts_genes.csv</code> , and <code>sparse_molecule_counts.mtx</code>) were gzip compressed. FALSE otherwise. Default seqc outputs are not gzipped. Default FALSE.
<code>class</code>	Character. The class of the expression matrix stored in the SCE object. Can be one of "Matrix" (as returned by <code>readMM</code> function), or "matrix" (as returned by <code>matrix</code> function). Default "Matrix".
<code>delayedArray</code>	Boolean. Whether to read the expression matrix as <code>DelayedArray</code> object or not. Default TRUE.
<code>cbNotFirstCol</code>	Boolean. TRUE if first column of <code>sparse_counts_barcode.csv</code> is row index and it will be removed. FALSE the first column will be kept.
<code>feNotFirstCol</code>	Boolean. TRUE if first column of <code>sparse_counts_genes.csv</code> is row index and it will be removed. FALSE the first column will be kept.
<code>combinedSample</code>	Boolean. If TRUE, <code>importSEQC</code> returns a <code>SingleCellExperiment</code> object containing the combined count matrix, feature annotations and the cell annotations. If FALSE, <code>importSEQC</code> returns a list containing multiple <code>SingleCellExperiment</code> objects. Each <code>SingleCellExperiment</code> contains count matrix, feature annotations and cell annotations for each sample.

Details

`importSEQC` imports output from seqc. The default `sparse_counts_barcode.csv` or `sparse_counts_genes.csv` from seqc output contains two columns. The first column is row index and the second column is cell-barcode or gene symbol. `importSEQC` will remove first column. Alternatively, user can call `cbNotFirstCol` or `feNotFirstCol` as FALSE to keep the first column of these files. When `combinedSample` is TRUE, `importSEQC` will combined count matrix with genes detected in at least one sample.

Value

A `SingleCellExperiment` object containing the combined count matrix, the feature annotations, and the cell annotation.

Examples

```
# Example #1
# The following filtered feature, cell, and matrix files were downloaded from
# https://support.10xgenomics.com/single-cell-gene-expression/datasets/
# 3.0.0/pbmc_1k_v3
# The top 50 hg38 genes are included in this example.
# Only the top 50 cells are included.
sce <- importSEQC(
  seqcDirs = system.file("extdata/pbmc_1k_50x50", package = "singleCellTK"),
  samples = "pbmc_1k_50x50",
  prefix = "pbmc_1k",
  combinedSample = FALSE)
```

importSTARsolo	<i>Construct SCE object from STARsolo outputs</i>
----------------	---

Description

Read the barcodes, features (genes), and matrices from STARsolo outputs. Import them as one [SingleCellExperiment](#) object.

Usage

```
importSTARsolo(
  STARsoloDirs,
  samples,
  STARsoloOuts = "Gene/filtered",
  matrixFileNames = "matrix.mtx",
  featuresFileNames = "features.tsv",
  barcodesFileNames = "barcodes.tsv",
  gzipped = "auto",
  class = c("Matrix", "matrix"),
  delayedArray = TRUE
)
```

Arguments

STARsoloDirs	A vector of root directories of STARsolo output files. The paths should be something like this: <i>/PATH/TO/prefixSolo.out</i> . For example: <i>./Solo.out</i> . Each sample should have its own path. Must have the same length as <code>samples</code> .
samples	A vector of user-defined sample names for the sample to be imported. Must have the same length as <code>STARsoloDirs</code> .
STARsoloOuts	Character vector. The intermediate paths to filtered or raw cell barcode, feature, and matrix files for each of samples. Default "Gene/filtered" which works for STAR 2.7.3a. Must have length 1 or the same length as <code>samples</code> .
matrixFileNames	Filenames for the Market Exchange Format (MEX) sparse matrix file (.mtx file). Must have length 1 or the same length as <code>samples</code> .
featuresFileNames	Filenames for the feature annotation file. Must have length 1 or the same length as <code>samples</code> .
barcodesFileNames	Filenames for the cell barcode list file. Must have length 1 or the same length as <code>samples</code> .
gzipped	Boolean. TRUE if the STARsolo output files (barcodes.tsv, features.tsv, and matrix.mtx) were gzip compressed. FALSE otherwise. This is FALSE in STAR 2.7.3a. Default "auto" which automatically detects if the files are gzip compressed. Must have length 1 or the same length as <code>samples</code> .
class	Character. The class of the expression matrix stored in the SCE object. Can be one of "Matrix" (as returned by readMM function), or "matrix" (as returned by matrix function). Default "Matrix".
delayedArray	Boolean. Whether to read the expression matrix as DelayedArray object or not. Default TRUE.

Value

A SingleCellExperiment object containing the count matrix, the gene annotation, and the cell annotation.

Examples

```
# Example #1
# FASTQ files were downloaded from
# https://support.10xgenomics.com/single-cell-gene-expression/datasets/3.0.0
# /pbmc_1k_v3
# They were concatenated as follows:
# cat pbmc_1k_v3_S1_L001_R1_001.fastq.gz pbmc_1k_v3_S1_L002_R1_001.fastq.gz >
# pbmc_1k_v3_R1.fastq.gz
# cat pbmc_1k_v3_S1_L001_R2_001.fastq.gz pbmc_1k_v3_S1_L002_R2_001.fastq.gz >
# pbmc_1k_v3_R2.fastq.gz
# The following STARsolo command generates the filtered feature, cell, and
# matrix files
# STAR \
# --genomeDir ./index \
# --readFilesIn ./pbmc_1k_v3_R2.fastq.gz \
#                ./pbmc_1k_v3_R1.fastq.gz \
# --readFilesCommand zcat \
# --outSAMtype BAM Unsorted \
# --outBAMcompression -1 \
# --soloType CB_UMI_Simple \
# --soloCBwhitelist ../737K-august-2016.txt \
# --soloUMIlen 12

# The top 20 genes and the first 20 cells are included in this example.
sce <- importSTARsolo(
  STARsoloDirs = system.file("extdata/STARsolo_PBMC_1k_v3_20x20",
    package = "singleCellTK"),
  samples = "PBMC_1k_v3_20x20")
```

iterateSimulations *Returns significance data from a snapshot.*

Description

Returns significance data from a snapshot.

Usage

```
iterateSimulations(
  originalData,
  useAssay = "counts",
  realLabels,
  totalReads,
  cells,
  iterations
)
```

Arguments

originalData	The SingleCellExperiment object storing all assay data from the shiny app.
useAssay	Character. The name of the assay to be used for subsampling.
realLabels	Character. The name of the condition of interest. Must match a name from sample data.
totalReads	Numeric. The total number of reads in the simulated dataset, to be split between all simulated cells.
cells	Numeric. The number of virtual cells to simulate.
iterations	Numeric. How many times should each experimental design be simulated.

Value

A matrix of significance information from a snapshot

Examples

```
data("mouseBrainSubsetSCE")
res <- iterateSimulations(mouseBrainSubsetSCE, realLabels = "level1class",
                          totalReads = 1000, cells = 10, iterations = 2)
```

mergeSCEColData	<i>Merging colData from two singleCellExperiment objects</i>
-----------------	--

Description

Merges colData of the singleCellExperiment objects obtained from the same dataset which contain differing colData. (i.e. raw data and filtered data)

Usage

```
mergeSCEColData(inSCE1, inSCE2, id1 = "column_name", id2 = "column_name")
```

Arguments

inSCE1	Input SingleCellExperiment object. The function will output this singleCellExperiment object with a combined colData from inSCE1 and inSCE2.
inSCE2	Input SingleCellExperiment object. colData from this object will be merged with colData from inSCE1 and loaded into inSCE1.
id1	Character vector. Column in colData of inSCE1 that will be used to combine inSCE1 and inSCE2. Default "column_name"
id2	Character vector. Column in colData of inSCE2 that will be used to combine inSCE1 and inSCE2. Default "column_name"

Value

SingleCellExperiment object containing combined colData from both singleCellExperiment for samples in inSCE1.

Examples

```
sce1 <- importCellRanger(
  cellRangerDirs = system.file("extdata/", package = "singleCellTK"),
  sampleDirs = "hgmm_1k_v3_20x20",
  sampleNames = "hgmm1kv3",
  dataType = "filtered")
data(sceExample)
sce2 <- sce
sce <- mergeSCEColData(inSCE1 = sce1, inSCE2 = sce2, id1 = "column_name", id2 = "column_name")
```

mouseBrainSubsetSCE	<i>Example Single Cell RNA-Seq data in SingleCellExperiment Object, GSE60361 subset</i>
---------------------	---

Description

A subset of 30 cells from a single cell RNA-Seq experiment from Zeisel, et al. Science 2015. The data was produced from cells from the mouse somatosensory cortex (S1) and hippocampus (CA1). 15 of the cells were identified as oligodendrocytes and 15 of the cell were identified as microglia.

Usage

```
mouseBrainSubsetSCE
```

Format

```
SingleCellExperiment
```

Source

```
DOI: 10.1126/science.aaa1934
```

Examples

```
data("mouseBrainSubsetSCE")
```

msigdb_table	<i>MSigDB gene set Category table</i>
--------------	---------------------------------------

Description

A table of gene set categories that can be download from MSigDB. The categories and descriptions can be found here: <https://www.gsea-msigdb.org/gsea/msigdb/collections.jsp>. The IDs in the first column can be used to retrieve the gene sets for these categories using the [importGeneSetsFromMSigDB](#) function.

Usage

```
msigdb_table
```


Format

A data.frame.

Examples

```
data("msigdb_table")
```

```
plotBarcodeRankDropsResults
```

Plots for runEmptyDrops outputs.

Description

A wrapper function which visualizes outputs from the runEmptyDrops function stored in the col-Data slot of the SingleCellExperiment object via plots.

Usage

```
plotBarcodeRankDropsResults(
  inSCE,
  sample = NULL,
  defaultTheme = TRUE,
  dotSize = 1,
  titleSize = 18,
  axisLabelSize = 18,
  axisSize = 15,
  legendSize = 15
)
```

Arguments

inSCE	Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results from runBarcodeRankDrops . Required.
sample	Character vector. Indicates which sample each cell belongs to. Default NULL.
defaultTheme	Removes grid in plot and sets axis title size to 10 when TRUE. Default TRUE.
dotSize	Size of dots. Default 1.
titleSize	Size of title of plot. Default 18.
axisLabelSize	Size of x/y-axis labels. Default 18.
axisSize	Size of x/y-axis ticks. Default 15.
legendSize	size of legend. Default 15.

Value

list of .ggplot objects

Examples

```
data(scExample, package="singleCellTK")
sce <- runBarcodeRankDrops(inSCE=sce)
plotBarcodeRankDropsResults(inSCE=sce)
```

plotBarcodeRankScatter

Plots for runBarcodeRankDrops outputs.

Description

A plotting function which visualizes outputs from the runBarcodeRankDrops function stored in the colData slot of the SingleCellExperiment object via scatterplot.

Usage

```
plotBarcodeRankScatter(
  inSCE,
  sample = NULL,
  defaultTheme = TRUE,
  dotSize = 1,
  title = NULL,
  titleSize = 18,
  xlab = NULL,
  ylab = NULL,
  axisSize = 12,
  axisLabelSize = 15,
  legendSize = 10,
  combinePlot = "none",
  sampleRelHeights = 1,
  sampleRelWidths = 1
)
```

Arguments

inSCE	Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results from runBarcodeRankDrops . Required.
sample	Character vector. Indicates which sample each cell belongs to. Default NULL.
defaultTheme	Removes grid in plot and sets axis title size to 10 when TRUE. Default TRUE.
dotSize	Size of dots. Default 1.
title	Title of plot. Default NULL.
titleSize	Size of title of plot. Default 18.
xlab	Character vector. Label for x-axis. Default NULL.
ylab	Character vector. Label for y-axis. Default NULL.
axisSize	Size of x/y-axis ticks. Default 12.
axisLabelSize	Size of x/y-axis labels. Default 15.
legendSize	size of legend. Default 10.
combinePlot	Boolean. If multiple plots are generated (multiple samples, etc.), will combined plots using 'cowplot::plot_grid'.
sampleRelHeights	If there are multiple samples and combining by "all", the relative heights for each plot.

sampleRelWidths

If there are multiple samples and combining by "all", the relative widths for each plot. Default TRUE.

Value

a ggplot object of the scatter plot.

Examples

```
data(scExample, package="singleCellTK")
sce <- runBarcodeRankDrops(inSCE=sce)
plotBarcodeRankScatter(inSCE=sce)
```

plotBatchVariance	<i>Plot the percent of the variation that is explained by batch and condition in the data</i>
-------------------	---

Description

Visualize the percent variation in the data that is explained by batch and condition, individually, and that explained by combining both annotations. Plotting only the variation explained by batch is supported but not recommended, because this can be confounded by potential condition.

Usage

```
plotBatchVariance(
  inSCE,
  useAssay = NULL,
  useReddim = NULL,
  useAltExp = NULL,
  batch = "batch",
  condition = NULL,
  title = NULL
)
```

Arguments

inSCE	SingleCellExperiment inherited object.
useAssay	A single character. The name of the assay that stores the value to plot. For useReddim and useAltExp also. Default NULL.
useReddim	A single character. The name of the dimension reduced matrix that stores the value to plot. Default NULL.
useAltExp	A single character. The name of the alternative experiment that stores an assay of the value to plot. Default NULL.
batch	A single character. The name of batch annotation column in colData(inSCE). Default "batch".
condition	A single character. The name of an additional condition annotation column in colData(inSCE). Default NULL.
title	A single character. The title text on the top. Default NULL.

Details

When condition and batch both are causing some variation, if the difference between full variation and condition variation is close to batch variation, this might imply that batches are causing some effect; if the difference is much less than batch variation, then the batches are likely to be confounded by the conditions.

Value

A ggplot object of a boxplot of variation explained by batch, condition, and batch+condition.

Examples

```
## Not run:
  if(requireNamespace("bladderbatch", quietly = TRUE)) {
    library(bladderbatch)
    data(bladderdata)
    dat <- as(as(bladderEset, "SummarizedExperiment"),
              "SingleCellExperiment")
    plotBatchVariance(dat,
                      useAssay="exprs",
                      batch="batch",
                      condition = "cancer")
  }

## End(Not run)
```

plotBcdsResults

Plots for runBcds outputs.

Description

A wrapper function which visualizes outputs from the runBcds function stored in the colData slot of the SingleCellExperiment object via various plots.

Usage

```
plotBcdsResults(
  inSCE,
  sample = NULL,
  shape = NULL,
  groupBy = NULL,
  combinePlot = "all",
  violin = TRUE,
  boxplot = FALSE,
  dots = TRUE,
  reducedDimName = NULL,
  xlab = NULL,
  ylab = NULL,
  dim1 = NULL,
  dim2 = NULL,
  bin = NULL,
```

```

binLabel = NULL,
defaultTheme = TRUE,
dotSize = 1,
transparency = 1,
titleSize = 18,
axisLabelSize = 18,
axisSize = 15,
legendSize = 15,
legendTitleSize = 16,
relHeights = c(1.5, 1, 1),
relWidths = c(1, 1, 1),
plotNcols = NULL,
plotNrows = NULL,
plotLabels = "default",
plotLabelSize = 20,
plotLabelPositionX = NULL,
plotLabelPositionY = NULL,
samplePerColumn = TRUE,
sampleRelHeights = 1,
sampleRelWidths = 1
)

```

Arguments

inSCE	Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results from runBcde . Required.
sample	Character vector. Indicates which sample each cell belongs to. Default NULL.
shape	If provided, add shapes based on the value.
groupBy	Groupings for each numeric value. A user may input a vector equal length to the number of the samples in the SingleCellExperiment object, or can be retrieved from the colData slot. Default NULL.
combinePlot	Must be either "all", "sample", or "none". "all" will combine all plots into a single .ggplot object, while "sample" will output a list of plots separated by sample. Default "all".
violin	Boolean. If TRUE, will plot the violin plot. Default TRUE.
boxplot	Boolean. If TRUE, will plot boxplots for each violin plot. Default TRUE.
dots	Boolean. If TRUE, will plot dots for each violin plot. Default TRUE.
reducedDimName	Saved dimension reduction name in the SingleCellExperiment object. Required.
xlab	Character vector. Label for x-axis. Default NULL.
ylab	Character vector. Label for y-axis. Default NULL.
dim1	1st dimension to be used for plotting. Can either be a string which specifies the name of the dimension to be plotted from reducedDims, or a numeric value which specifies the index of the dimension to be plotted. Default is NULL.
dim2	2nd dimension to be used for plotting. Can either be a string which specifies the name of the dimension to be plotted from reducedDims, or a numeric value which specifies the index of the dimension to be plotted. Default is NULL.
bin	Numeric vector. If single value, will divide the numeric values into the 'bin' groups. If more than one value, will bin numeric values using values as a cut point.

binLabel	Character vector. Labels for the bins created by the 'bin' parameter. Default NULL.
defaultTheme	Removes grid in plot and sets axis title size to 10 when TRUE. Default TRUE.
dotSize	Size of dots. Default 1.
transparency	Transparency of the dots, values will be 0-1. Default 1.
titleSize	Size of title of plot. Default 18.
axisLabelSize	Size of x/y-axis labels. Default 18.
axisSize	Size of x/y-axis ticks. Default 15.
legendSize	size of legend. Default 15.
legendTitleSize	size of legend title. Default 16.
relHeights	Relative heights of plots when combine is set.
relWidths	Relative widths of plots when combine is set.
plotNCols	Number of columns when plots are combined in a grid.
plotNRows	Number of rows when plots are combined in a grid.
plotLabels	labels to each plot. If set to "default", will use the name of the samples as the labels. If set to "none", no label will be plotted.
plotLabelSize	size of labels
plotLabelPositionX	Numeric vector. The X position of the plot label.
plotLabelPositionY	Numeric vector. The Y position of the plot label.
samplePerColumn	If TRUE, when there are multiple samples and combining by "all", the output .ggplot will have plots from each sample on a single column. Default TRUE.
sampleRelHeights	If there are multiple samples and combining by "all", the relative heights for each plot.
sampleRelWidths	If there are multiple samples and combining by "all", the relative widths for each plot.

Value

list of .ggplot objects

Examples

```
data(scExample, package="singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
sce <- getUMAP(inSCE=sce, useAssay="counts", reducedDimName="UMAP")
sce <- runBcde(sce)
plotBcdeResults(inSCE=sce, reducedDimName="UMAP")
```

plotBiomarker	<i>Given a set of genes, return a ggplot of expression values.</i>
---------------	--

Description

Given a set of genes, return a ggplot of expression values.

Usage

```
plotBiomarker(
  inSCE,
  gene,
  binary = "Binary",
  shape = "No Shape",
  useAssay = "counts",
  reducedDimName = "PCA",
  x = NULL,
  y = NULL,
  comp1 = NULL,
  comp2 = NULL
)
```

Arguments

inSCE	Input SingleCellExperiment object.
gene	genelist to run the method on.
binary	binary/continuous color for the expression.
shape	shape parameter for the ggplot.
useAssay	Indicate which assay to use. The default is "logcounts".
reducedDimName	results name of dimension reduction coordinates obtained from this method. This is stored in the SingleCellExperiment object in the reducedDims slot. Required.
x	PCA component to be used for plotting(if applicable). Default is first PCA component for PCA data and NULL otherwise.
y	PCA component to be used for plotting(if applicable). Default is second PCA component for PCA data and NULL otherwise.
comp1	label for x-axis
comp2	label for y-axis

Value

A Biomarker plot

Examples

```
data("mouseBrainSubsetSCE")
plotBiomarker(mouseBrainSubsetSCE, gene="C1qa", shape="level1class", reducedDimName="TSNE_counts")
```

plotCxlsResults	<i>Plots for runCxls outputs.</i>
-----------------	-----------------------------------

Description

A wrapper function which visualizes outputs from the runCxls function stored in the colData slot of the SingleCellExperiment object via various plots.

Usage

```
plotCxlsResults(  
  inSCE,  
  sample = NULL,  
  shape = NULL,  
  groupBy = NULL,  
  combinePlot = "all",  
  violin = TRUE,  
  boxplot = FALSE,  
  dots = TRUE,  
  reducedDimName = NULL,  
  xlab = NULL,  
  ylab = NULL,  
  dim1 = NULL,  
  dim2 = NULL,  
  bin = NULL,  
  binLabel = NULL,  
  defaultTheme = TRUE,  
  dotSize = 1,  
  transparency = 1,  
  titleSize = 18,  
  axisLabelSize = 18,  
  axisSize = 15,  
  legendSize = 15,  
  legendTitleSize = 16,  
  relHeights = c(1.5, 1, 1),  
  relWidths = c(1, 1, 1),  
  plotNcols = NULL,  
  plotNrows = NULL,  
  plotLabels = "default",  
  plotLabelSize = 20,  
  plotLabelPositionX = NULL,  
  plotLabelPositionY = NULL,  
  samplePerColumn = TRUE,  
  sampleRelHeights = 1,  
  sampleRelWidths = 1  
)
```

Arguments

inSCE	Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results from runCxls .
-------	---

sample	Character vector. Indicates which sample each cell belongs to. Default NULL.
shape	If provided, add shapes based on the value.
groupBy	Groupings for each numeric value. A user may input a vector equal length to the number of the samples in the SingleCellExperiment object, or can be retrieved from the colData slot. Default NULL.
combinePlot	Must be either "all", "sample", or "none". "all" will combine all plots into a single .ggplot object, while "sample" will output a list of plots separated by sample. Default "all".
violin	Boolean. If TRUE, will plot the violin plot. Default TRUE.
boxplot	Boolean. If TRUE, will plot boxplots for each violin plot. Default TRUE.
dots	Boolean. If TRUE, will plot dots for each violin plot. Default TRUE.
reducedDimName	Saved dimension reduction name in the SingleCellExperiment object. Required.
xlab	Character vector. Label for x-axis. Default NULL.
ylab	Character vector. Label for y-axis. Default NULL.
dim1	1st dimension to be used for plotting. Can either be a string which specifies the name of the dimension to be plotted from reducedDims, or a numeric value which specifies the index of the dimension to be plotted. Default is NULL.
dim2	2nd dimension to be used for plotting. Can either be a string which specifies the name of the dimension to be plotted from reducedDims, or a numeric value which specifies the index of the dimension to be plotted. Default is NULL.
bin	Numeric vector. If single value, will divide the numeric values into the 'bin' groups. If more than one value, will bin numeric values using values as a cut point.
binLabel	Character vector. Labels for the bins created by the 'bin' parameter. Default NULL.
defaultTheme	Removes grid in plot and sets axis title size to 10 when TRUE. Default TRUE.
dotSize	Size of dots. Default 1.
transparency	Transparency of the dots, values will be 0-1. Default 1.
titleSize	Size of title of plot. Default 18.
axisLabelSize	Size of x/y-axis labels. Default 18.
axisSize	Size of x/y-axis ticks. Default 15.
legendSize	size of legend. Default 15.
legendTitleSize	size of legend title. Default 16.
relHeights	Relative heights of plots when combine is set.
relWidths	Relative widths of plots when combine is set.
plotNCols	Number of columns when plots are combined in a grid.
plotNRows	Number of rows when plots are combined in a grid.
plotLabels	labels to each plot. If set to "default", will use the name of the samples as the labels. If set to "none", no label will be plotted.
plotLabelSize	size of labels
plotLabelPositionX	Numeric vector. The X position of the plot label.

plotLabelPositionY	Numeric vector. The Y position of the plot label.
samplePerColumn	If TRUE, when there are multiple samples and combining by "all", the output .ggplot will have plots from each sample on a single column. Default TRUE.
sampleRelHeights	If there are multiple samples and combining by "all", the relative heights for each plot.
sampleRelWidths	If there are multiple samples and combining by "all", the relative widths for each plot.

Value

list of .ggplot objects

Examples

```
data(scExample, package="singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
sce <- getUMAP(inSCE=sce, useAssay="counts", reducedDimName="UMAP")
sce <- runCxds(sce)
plotCxdsResults(inSCE=sce, reducedDimName="UMAP")
```

plotDecontXResults *Plots for runDecontX outputs.*

Description

A wrapper function which visualizes outputs from the runDecontX function stored in the colData slot of the SingleCellExperiment object via various plots.

Usage

```
plotDecontXResults(
  inSCE,
  sample = NULL,
  shape = NULL,
  groupBy = NULL,
  combinePlot = "all",
  violin = TRUE,
  boxplot = FALSE,
  dots = TRUE,
  reducedDimName = NULL,
  xlab = NULL,
  ylab = NULL,
  dim1 = NULL,
  dim2 = NULL,
  bin = NULL,
  binLabel = NULL,
  defaultTheme = TRUE,
```

```

dotSize = 1,
transparency = 1,
titleSize = 18,
axisLabelSize = 18,
axisSize = 15,
legendSize = 15,
legendTitleSize = 16,
relHeights = c(1.5, 1, 1),
relWidths = c(1, 1, 1),
plotNcols = NULL,
plotNrows = NULL,
plotLabels = "default",
plotLabelSize = 20,
plotLabelPositionX = NULL,
plotLabelPositionY = NULL,
samplePerColumn = TRUE,
sampleRelHeights = 1,
sampleRelWidths = 1
)

```

Arguments

inSCE	Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results from runDecontX . Required.
sample	Character vector. Indicates which sample each cell belongs to. Default NULL.
shape	If provided, add shapes based on the value.
groupBy	Groupings for each numeric value. A user may input a vector equal length to the number of the samples in the SingleCellExperiment object, or can be retrieved from the colData slot. Default NULL.
combinePlot	Must be either "all", "sample", or "none". "all" will combine all plots into a single <code>.ggplot</code> object, while "sample" will output a list of plots separated by sample. Default "all".
violin	Boolean. If TRUE, will plot the violin plot. Default TRUE.
boxplot	Boolean. If TRUE, will plot boxplots for each violin plot. Default TRUE.
dots	Boolean. If TRUE, will plot dots for each violin plot. Default TRUE.
reducedDimName	Saved dimension reduction name in the SingleCellExperiment object. Required.
xlab	Character vector. Label for x-axis. Default NULL.
ylab	Character vector. Label for y-axis. Default NULL.
dim1	1st dimension to be used for plotting. Can either be a string which specifies the name of the dimension to be plotted from <code>reducedDims</code> , or a numeric value which specifies the index of the dimension to be plotted. Default is NULL.
dim2	2nd dimension to be used for plotting. Can either be a string which specifies the name of the dimension to be plotted from <code>reducedDims</code> , or a numeric value which specifies the index of the dimension to be plotted. Default is NULL.
bin	Numeric vector. If single value, will divide the numeric values into the 'bin' groups. If more than one value, will bin numeric values using values as a cut point.
binLabel	Character vector. Labels for the bins created by the 'bin' parameter. Default NULL.

defaultTheme	Removes grid in plot and sets axis title size to 10 when TRUE. Default TRUE.
dotSize	Size of dots. Default 1.
transparency	Transparency of the dots, values will be 0-1. Default 1.
titleSize	Size of title of plot. Default 18.
axisLabelSize	Size of x/y-axis labels. Default 18.
axisSize	Size of x/y-axis ticks. Default 15.
legendSize	size of legend. Default 15.
legendTitleSize	size of legend title. Default 16.
relHeights	Relative heights of plots when combine is set.
relWidths	Relative widths of plots when combine is set.
plotNCols	Number of columns when plots are combined in a grid.
plotNRows	Number of rows when plots are combined in a grid.
plotLabels	labels to each plot. If set to "default", will use the name of the samples as the labels. If set to "none", no label will be plotted.
plotLabelSize	size of labels
plotLabelPositionX	Numeric vector. The X position of the plot label.
plotLabelPositionY	Numeric vector. The Y position of the plot label.
samplePerColumn	If TRUE, when there are multiple samples and combining by "all", the output .ggplot will have plots from each sample on a single column. Default TRUE.
sampleRelHeights	If there are multiple samples and combining by "all", the relative heights for each plot.
sampleRelWidths	If there are multiple samples and combining by "all", the relative widths for each plot.

Value

list of .ggplot objects

Examples

```
data(scExample, package="singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
sce <- getUMAP(inSCE=sce, useAssay="counts", reducedDimName="UMAP")
sce <- runDecontX(sce)
plotDecontXResults(inSCE=sce, reducedDimName="UMAP")
```

plotDEGHeatmap

*Heatmap visualization of DEG result***Description**

A differential expression analysis function has to be run in advance so that information is stored in the metadata of the input SCE object. This function wraps plotSCEHeatmap. A feature annotation basing on the log2FC level called "regulation" will be automatically added. A cell annotation basing on the condition selection while running the analysis called "condition", and the annotations used from colData(inSCE) while setting the condition and covariates will also be added.

Usage

```
plotDEGHeatmap(
  inSCE,
  useResult,
  doLog = FALSE,
  onlyPos = FALSE,
  log2fcThreshold = 0.25,
  fdrThreshold = 0.05,
  useAssay = NULL,
  featureAnnotations = NULL,
  cellAnnotations = NULL,
  featureAnnotationColor = NULL,
  cellAnnotationColor = NULL,
  rowDataName = NULL,
  colDataName = NULL,
  colSplitBy = "condition",
  rowSplitBy = "regulation",
  title = paste0("MAST Result: ", useResult),
  ...
)
```

Arguments

inSCE	SingleCellExperiment inherited object. runMAST() has to be run in advance.
useResult	character. A string specifying the analysisName used when running a differential expression analysis function.
doLog	Logical scalar. Whether to do log(assay + 1) transformation on the assay used for the analysis. Default FALSE.
onlyPos	logical. Whether to only plot DEG with positive log2_FC value. Default FALSE.
log2fcThreshold	numeric. Only plot DEGs with the absolute values of log2FC larger than this value. Default 1.
fdrThreshold	numeric. Only plot DEGs with FDR value smaller than this value. Default 0.05.
useAssay	character. A string specifying an assay of expression value to plot. By default the assay used for runMAST() will be used. Default NULL.
featureAnnotations	data.frame, with rownames containing all the features going to be plotted. Character columns should be factors. Default NULL.

cellAnnotations	data.frame, with rownames containing all the cells going to be plotted. Character columns should be factors. Default NULL.
featureAnnotationColor	A named list. Customized color settings for feature labeling. Should match the entries in the featureAnnotations or rowDataName. For each entry, there should be a list/vector of colors named with categories. Default NULL.
cellAnnotationColor	A named list. Customized color settings for cell labeling. Should match the entries in the cellAnnotations or colDataName. For each entry, there should be a list/vector of colors named with categories. Default NULL.
rowDataName	character. The column name(s) in rowData that need to be added to the annotation. Default NULL.
colDataName	character. The column name(s) in colData that need to be added to the annotation. Default NULL.
colSplitBy	character. Do semi-heatmap based on the grouping of this(these) annotation(s). Should exist in either colDataName or names(cellAnnotations). Default "condition".
rowSplitBy	character. Do semi-heatmap based on the grouping of this(these) annotation(s). Should exist in either rowDataName or names(featureAnnotations). Default "regulation".
title	character. Main title of the heatmap. Default "MAST Result: <useResult>".
...	Other arguments passed to plotSCEHeatmap

Value

A ComplexHeatmap::Heatmap object

Author(s)

Yichen Wang

Examples

```
data(scExample, package = "singleCellTK")
## Not run:
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
sce <- runDEAnalysis(inSCE = sce, method = "DESeq2",
                    groupName1 = "Sample1", groupName2 = "Sample2",
                    index1 = 1:100, index2 = 101:190,
                    analysisName = "DESeq2")
plotDEGHeatmap(sce, useResult = "DESeq2", fdrThreshold = 1, doLog = TRUE)

## End(Not run)
```

plotDEGRegression	<i>plot the linear regression to show visualize the expression the of DEGs identified by differential expression analysis</i>
-------------------	---

Description

plot the linear regression to show visualize the expression the of DEGs identified by differential expression analysis

Usage

```
plotDEGRegression(
  inSCE,
  useResult,
  threshP = FALSE,
  labelBy = NULL,
  nrow = 6,
  ncol = 6,
  defaultTheme = TRUE,
  isLogged = TRUE,
  check_sanity = TRUE
)
```

Arguments

inSCE	SingleCellExperiment inherited object. runMAST() has to be run in advance.
useResult	character. A string specifying the analysisName used when running a differential expression analysis function.
threshP	logical. Whether to plot threshold values from adaptive thresholding, instead of using the assay used by runMAST(). Default FALSE.
labelBy	A single character for a column of rowData(inSCE) as where to search for the labeling text. Default NULL.
nrow	Integer. Number of rows in the plot grid. Default 6.
ncol	Integer. Number of columns in the plot grid. Default 6.
defaultTheme	Logical scalar. Whether to use default SCTK theme in ggplot. Default TRUE.
isLogged	Logical scalar. Whether the assay used for the analysis is logged. If not, will do a log(assay + 1) transformation. Default TRUE.
check_sanity	Logical scalar. Whether to perform MAST's sanity check to see if the counts are logged. Default TRUE

Value

A ggplot object of linear regression

plotDEGViolin	<i>plot the violin plot to show visualize the expression distribution of DEGs identified by differential expression analysis</i>
---------------	--

Description

plot the violin plot to show visualize the expression distribution of DEGs identified by differential expression analysis

Usage

```
plotDEGViolin(
  inSCE,
  useResult,
  threshP = FALSE,
  labelBy = NULL,
  nrow = 6,
  ncol = 6,
  defaultTheme = TRUE,
  isLogged = TRUE,
  check_sanity = TRUE
)
```

Arguments

inSCE	SingleCellExperiment inherited object. runMAST() has to be run in advance.
useResult	character. A string specifying the analysisName used when running a differential expression analysis function.
threshP	logical. Whether to plot threshold values from adaptive thresholding, instead of using the assay used by runMAST(). Default FALSE.
labelBy	A single character for a column of rowData(inSCE) as where to search for the labeling text. Default NULL.
nrow	Integer. Number of rows in the plot grid. Default 6.
ncol	Integer. Number of columns in the plot grid. Default 6.
defaultTheme	Logical scalar. Whether to use default SCTK theme in ggplot. Default TRUE.
isLogged	Logical scalar. Whether the assay used for the analysis is logged. If not, will do a log(assay + 1) transformation. Default TRUE.
check_sanity	Logical scalar. Whether to perform MAST's sanity check to see if the counts are logged. Default TRUE

Value

A ggplot object of violin plot

`plotDoubletCellsResults`*Plots for runDoubletCells outputs.*

Description

A wrapper function which visualizes outputs from the `runDoubletCells` function stored in the `col-Data` slot of the `SingleCellExperiment` object via various plots.

Usage

```
plotDoubletCellsResults(  
  inSCE,  
  sample = NULL,  
  shape = NULL,  
  groupBy = NULL,  
  combinePlot = "all",  
  violin = TRUE,  
  boxplot = FALSE,  
  dots = TRUE,  
  reducedDimName = NULL,  
  xlab = NULL,  
  ylab = NULL,  
  dim1 = NULL,  
  dim2 = NULL,  
  bin = NULL,  
  binLabel = NULL,  
  defaultTheme = TRUE,  
  dotSize = 1,  
  transparency = 1,  
  titleSize = 18,  
  axisLabelSize = 18,  
  axisSize = 15,  
  legendSize = 15,  
  legendTitleSize = 16,  
  relHeights = c(1.5, 1, 1),  
  relWidths = c(1, 1, 1),  
  plotNcols = NULL,  
  plotNrows = NULL,  
  plotLabels = "default",  
  plotLabelSize = 20,  
  plotLabelPositionX = NULL,  
  plotLabelPositionY = NULL,  
  samplePerColumn = TRUE,  
  sampleRelHeights = 1,  
  sampleRelWidths = 1  
)
```

Arguments

<code>inSCE</code>	Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results from runDoubletCells . Required.
<code>sample</code>	Character vector. Indicates which sample each cell belongs to. Default NULL.
<code>shape</code>	If provided, add shapes based on the value.
<code>groupBy</code>	Groupings for each numeric value. A user may input a vector equal length to the number of the samples in the SingleCellExperiment object, or can be retrieved from the <code>colData</code> slot. Default NULL.
<code>combinePlot</code>	Must be either "all", "sample", or "none". "all" will combine all plots into a single <code>.ggplot</code> object, while "sample" will output a list of plots separated by sample. Default "all".
<code>violin</code>	Boolean. If TRUE, will plot the violin plot. Default TRUE.
<code>boxplot</code>	Boolean. If TRUE, will plot boxplots for each violin plot. Default TRUE.
<code>dots</code>	Boolean. If TRUE, will plot dots for each violin plot. Default TRUE.
<code>reducedDimName</code>	Saved dimension reduction name in the SingleCellExperiment object. Required.
<code>xlab</code>	Character vector. Label for x-axis. Default NULL.
<code>ylab</code>	Character vector. Label for y-axis. Default NULL.
<code>dim1</code>	1st dimension to be used for plotting. Can either be a string which specifies the name of the dimension to be plotted from <code>reducedDims</code> , or a numeric value which specifies the index of the dimension to be plotted. Default is NULL.
<code>dim2</code>	2nd dimension to be used for plotting. Can either be a string which specifies the name of the dimension to be plotted from <code>reducedDims</code> , or a numeric value which specifies the index of the dimension to be plotted. Default is NULL.
<code>bin</code>	Numeric vector. If single value, will divide the numeric values into the 'bin' groups. If more than one value, will bin numeric values using values as a cut point.
<code>binLabel</code>	Character vector. Labels for the bins created by the 'bin' parameter. Default NULL.
<code>defaultTheme</code>	Removes grid in plot and sets axis title size to 10 when TRUE. Default TRUE.
<code>dotSize</code>	Size of dots. Default 1.
<code>transparency</code>	Transparency of the dots, values will be 0-1. Default 1.
<code>titleSize</code>	Size of title of plot. Default 18.
<code>axisLabelSize</code>	Size of x/y-axis labels. Default 18.
<code>axisSize</code>	Size of x/y-axis ticks. Default 15.
<code>legendSize</code>	size of legend. Default 15.
<code>legendTitleSize</code>	size of legend title. Default 16.
<code>relHeights</code>	Relative heights of plots when combine is set.
<code>relWidths</code>	Relative widths of plots when combine is set.
<code>plotNcols</code>	Number of columns when plots are combined in a grid.
<code>plotNrows</code>	Number of rows when plots are combined in a grid.
<code>plotLabels</code>	labels to each plot. If set to "default", will use the name of the samples as the labels. If set to "none", no label will be plotted.

`plotLabelSize` size of labels
`plotLabelPositionX`
 Numeric vector. The X position of the plot label.
`plotLabelPositionY`
 Numeric vector. The Y position of the plot label.
`samplePerColumn`
 If TRUE, when there are multiple samples and combining by "all", the output .ggplot will have plots from each sample on a single column. Default TRUE.
`sampleRelHeights`
 If there are multiple samples and combining by "all", the relative heights for each plot.
`sampleRelWidths`
 If there are multiple samples and combining by "all", the relative widths for each plot.

Value

list of .ggplot objects

Examples

```

data(scExample, package="singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
sce <- getUMAP(inSCE=sce, useAssay="counts", reducedDimName="UMAP")
sce <- runDoubletCells(sce)
plotDoubletCellsResults(inSCE=sce, reducedDimName="UMAP")
  
```

plotDoubletFinderResults

Plots for runDoubletFinder outputs.

Description

A wrapper function which visualizes outputs from the runDoubletFinder function stored in the colData slot of the SingleCellExperiment object via various plots.

Usage

```

plotDoubletFinderResults(
  inSCE,
  sample = NULL,
  shape = NULL,
  groupBy = NULL,
  combinePlot = "all",
  violin = TRUE,
  boxplot = FALSE,
  dots = TRUE,
  reducedDimName = NULL,
  xlab = NULL,
  ylab = NULL,
  dim1 = NULL,
  )
  
```

```

dim2 = NULL,
bin = NULL,
binLabel = NULL,
defaultTheme = TRUE,
dotSize = 1,
transparency = 1,
titleSize = 18,
axisLabelSize = 18,
axisSize = 15,
legendSize = 15,
legendTitleSize = 16,
relHeights = c(1.5, 1, 1),
relWidths = c(1, 1, 1),
plotNcols = NULL,
plotNrows = NULL,
plotLabels = "default",
plotLabelSize = 20,
plotLabelPositionX = NULL,
plotLabelPositionY = NULL,
samplePerColumn = TRUE,
sampleRelHeights = 1,
sampleRelWidths = 1
)

```

Arguments

inSCE	Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results from runDoubletFinder . Required.
sample	Character vector. Indicates which sample each cell belongs to. Default NULL.
shape	If provided, add shapes based on the value.
groupBy	Groupings for each numeric value. A user may input a vector equal length to the number of the samples in the SingleCellExperiment object, or can be retrieved from the colData slot. Default NULL.
combinePlot	Must be either "all", "sample", or "none". "all" will combine all plots into a single .ggplot object, while "sample" will output a list of plots separated by sample. Default "all".
violin	Boolean. If TRUE, will plot the violin plot. Default TRUE.
boxplot	Boolean. If TRUE, will plot boxplots for each violin plot. Default TRUE.
dots	Boolean. If TRUE, will plot dots for each violin plot. Default TRUE.
reducedDimName	Saved dimension reduction name in the SingleCellExperiment object. Required.
xlab	Character vector. Label for x-axis. Default NULL.
ylab	Character vector. Label for y-axis. Default NULL.
dim1	1st dimension to be used for plotting. Can either be a string which specifies the name of the dimension to be plotted from reducedDims, or a numeric value which specifies the index of the dimension to be plotted. Default is NULL.
dim2	2nd dimension to be used for plotting. Can either be a string which specifies the name of the dimension to be plotted from reducedDims, or a numeric value which specifies the index of the dimension to be plotted. Default is NULL.

bin	Numeric vector. If single value, will divide the numeric values into the 'bin' groups. If more than one value, will bin numeric values using values as a cut point.
binLabel	Character vector. Labels for the bins created by the 'bin' parameter. Default NULL.
defaultTheme	Removes grid in plot and sets axis title size to 10 when TRUE. Default TRUE.
dotSize	Size of dots. Default 1.
transparency	Transparency of the dots, values will be 0-1. Default 1.
titleSize	Size of title of plot. Default 18.
axisLabelSize	Size of x/y-axis labels. Default 18.
axisSize	Size of x/y-axis ticks. Default 15.
legendSize	size of legend. Default 15.
legendTitleSize	size of legend title. Default 16.
relHeights	Relative heights of plots when combine is set.
relWidths	Relative widths of plots when combine is set.
plotNCols	Number of columns when plots are combined in a grid.
plotNRows	Number of rows when plots are combined in a grid.
plotLabels	labels to each plot. If set to "default", will use the name of the samples as the labels. If set to "none", no label will be plotted.
plotLabelSize	size of labels
plotLabelPositionX	Numeric vector. The X position of the plot label.
plotLabelPositionY	Numeric vector. The Y position of the plot label.
samplePerColumn	If TRUE, when there are multiple samples and combining by "all", the output .ggplot will have plots from each sample on a single column. Default TRUE.
sampleRelHeights	If there are multiple samples and combining by "all", the relative heights for each plot.
sampleRelWidths	If there are multiple samples and combining by "all", the relative widths for each plot.

Value

list of .ggplot objects

Examples

```
data(scExample, package="singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
sce <- getUMAP(inSCE=sce, useAssay="counts", reducedDimName="UMAP")
sce <- runDoubletFinder(sce)
plotDoubletFinderResults(inSCE=sce, reducedDimName="UMAP")
```

plotEmptyDropsResults *Plots for runEmptyDrops outputs.*

Description

A wrapper function which visualizes outputs from the runEmptyDrops function stored in the col-Data slot of the SingleCellExperiment object via plots.

Usage

```
plotEmptyDropsResults(
  inSCE,
  sample = NULL,
  combinePlot = "all",
  fdrCutoff = 0.01,
  defaultTheme = TRUE,
  dotSize = 1,
  titleSize = 18,
  axisLabelSize = 18,
  axisSize = 15,
  legendSize = 15,
  legendTitleSize = 16,
  relHeights = 1,
  relWidths = 1,
  samplePerColumn = TRUE,
  sampleRelHeights = 1,
  sampleRelWidths = 1
)
```

Arguments

inSCE	Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results from runScrublet . Required.
sample	Character vector. Indicates which sample each cell belongs to. Default NULL.
combinePlot	Must be either "all", "sample", or "none". "all" will combine all plots into a single .ggplot object, while "sample" will output a list of plots separated by sample. Default "all".
fdrCutoff	Numeric. Thresholds barcodes based on the FDR values from runEmptyDrops as "Empty Droplet" or "Putative Cell". Default 0.01.
defaultTheme	Removes grid in plot and sets axis title size to 10 when TRUE. Default TRUE.
dotSize	Size of dots. Default 1.
titleSize	Size of title of plot. Default 18.
axisLabelSize	Size of x/y-axis labels. Default 18.
axisSize	Size of x/y-axis ticks. Default 15.
legendSize	size of legend. Default 15.
legendTitleSize	size of legend title. Default 16.
relHeights	Relative heights of plots when combine is set.

relWidths	Relative widths of plots when combine is set.
samplePerColumn	If TRUE, when there are multiple samples and combining by "all", the output .ggplot will have plots from each sample on a single column. Default TRUE.
sampleRelHeights	If there are multiple samples and combining by "all", the relative heights for each plot.
sampleRelWidths	If there are multiple samples and combining by "all", the relative widths for each plot.

Value

list of .ggplot objects

Examples

```
data(scExample, package="singleCellTK")
sce <- runEmptyDrops(inSCE=sce)
plotEmptyDropsResults(inSCE=sce)
```

plotEmptyDropsScatter *Plots for runEmptyDrops outputs.*

Description

A plotting function which visualizes outputs from the runEmptyDrops function stored in the colData slot of the SingleCellExperiment object via scatterplot.

Usage

```
plotEmptyDropsScatter(
  inSCE,
  sample = NULL,
  fdrCutoff = 0.01,
  defaultTheme = TRUE,
  dotSize = 1,
  title = NULL,
  titleSize = 18,
  xlab = NULL,
  ylab = NULL,
  axisSize = 12,
  axisLabelSize = 15,
  legendTitle = NULL,
  legendTitleSize = 12,
  legendSize = 10,
  combinePlot = "none",
  relHeights = 1,
  relWidths = 1,
  samplePerColumn = TRUE,
  sampleRelHeights = 1,
  sampleRelWidths = 1
)
```

Arguments

<code>inSCE</code>	Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results from runEmptyDrops . Required.
<code>sample</code>	Character vector. Indicates which sample each cell belongs to. Default NULL.
<code>fdrCutoff</code>	Numeric. Thresholds barcodes based on the FDR values from runEmptyDrops as "Empty Droplet" or "Putative Cell". Default 0.01.
<code>defaultTheme</code>	Removes grid in plot and sets axis title size to 10 when TRUE. Default TRUE.
<code>dotSize</code>	Size of dots. Default 1.
<code>title</code>	Title of plot. Default NULL.
<code>titleSize</code>	Size of title of plot. Default 18.
<code>xlab</code>	Character vector. Label for x-axis. Default NULL.
<code>ylab</code>	Character vector. Label for y-axis. Default NULL.
<code>axisSize</code>	Size of x/y-axis ticks. Default 12.
<code>axisLabelSize</code>	Size of x/y-axis labels. Default 15.
<code>legendTitle</code>	Title of legend. Default NULL.
<code>legendTitleSize</code>	size of legend title. Default 12.
<code>legendSize</code>	size of legend. Default 10.
<code>combinePlot</code>	Boolean. If multiple plots are generated (multiple samples, etc.), will combined plots using 'cowplot::plot_grid'. Default TRUE.
<code>relHeights</code>	Relative heights of plots when combine is set.
<code>relWidths</code>	Relative widths of plots when combine is set.
<code>samplePerColumn</code>	If TRUE, when there are multiple samples and combining by "all", the output .ggplot will have plots from each sample on a single column. Default TRUE.
<code>sampleRelHeights</code>	If there are multiple samples and combining by "all", the relative heights for each plot.
<code>sampleRelWidths</code>	If there are multiple samples and combining by "all", the relative widths for each plot.

Value

a ggplot object of the scatter plot.

Examples

```
data(scExample, package="singleCellTK")
sce <- runEmptyDrops(inSCE=sce)
plotEmptyDropsScatter(inSCE=sce)
```

plotMarkerDiffExp *Plot a heatmap to visualize the result of `findMarkerDiffExp`*

Description

This function will first reads the result saved in metadata slot, named by "findMarker" and generated by `findMarkerDiffExp`. Then it do the filtering on the statistics based on the input parameters and get unique genes to plot. We choose the genes that are identified as up-regulated only. As for the genes identified as up-regulated for multiple clusters, we only keep the belonging towards the one they have the highest Log2FC value. In the heatmap, there will always be a cell annotation for the cluster labeling used when finding the markers, and a feature annotation for which cluster each gene belongs to. And by default we split the heatmap by these two annotations. Additional legends can be added and the splitting can be canceled.

Usage

```
plotMarkerDiffExp(
  inSCE,
  useAssay = "logcounts",
  orderBy = "size",
  log2fcThreshold = 1,
  fdrThreshold = 0.05,
  topN = 10,
  decreasing = TRUE,
  rowDataName = NULL,
  colDataName = NULL,
  featureAnnotations = NULL,
  cellAnnotations = NULL,
  featureAnnotationColor = NULL,
  cellAnnotationColor = NULL,
  colSplitBy = ifelse(is.null(orderBy), NULL, colnames(inSCE@metadata$findMarker)[5]),
  rowSplitBy = "marker",
  ...
)
```

Arguments

<code>inSCE</code>	SingleCellExperiment inherited object.
<code>useAssay</code>	character. A string specifying which assay to use for the expression values. Default "logcounts".
<code>orderBy</code>	The ordering method of the clusters on the splitted heatmap. Can be chosen from "size" or "name", specified with vector of ordered unique cluster labels, or set as NULL for unsplitted heatmap. Default "size".
<code>log2fcThreshold</code>	Only use DEGs with the absolute values of log2FC larger than this value. Default 1
<code>fdrThreshold</code>	Only use DEGs with FDR value smaller than this value. Default 0.05
<code>topN</code>	An integer. Only to plot this number of top markers for each cluster in maximum, in terms of log2FC value. Use NULL to cancel the top N subscription. Default 10.

decreasing	Order the cluster decreasingly. Default TRUE.
rowDataName	character. The column name(s) in rowData that need to be added to the annotation. Default NULL.
colDataName	character. The column name(s) in colData that need to be added to the annotation. Default NULL.
featureAnnotations	data.frame, with rownames containing all the features going to be plotted. Character columns should be factors. Default NULL.
cellAnnotations	data.frame, with rownames containing all the cells going to be plotted. Character columns should be factors. Default NULL.
featureAnnotationColor	A named list. Customized color settings for feature labeling. Should match the entries in the featureAnnotations or rowDataName. For each entry, there should be a list/vector of colors named with categories. Default NULL.
cellAnnotationColor	A named list. Customized color settings for cell labeling. Should match the entries in the cellAnnotations or colDataName. For each entry, there should be a list/vector of colors named with categories. Default NULL.
colSplitBy	character vector. Do semi-heatmap based on the grouping of this(these) annotation(s). Should exist in either colDataName or names(cellAnnotations). Default is the value of cluster in <code>findMarkerDiffExp</code> when orderBy is not NULL, or NULL otherwise.
rowSplitBy	character vector. Do semi-heatmap based on the grouping of this(these) annotation(s). Should exist in either rowDataName or names(featureAnnotations). Default "marker", which indicates an auto generated annotation for this plot.
...	Other arguments passed to <code>plotSCEHeatmap</code> .

Value

A `Heatmap` object

Author(s)

Yichen Wang

plotMASTThresholdGenes

MAST Identify adaptive thresholds

Description

Calculate and produce a list of thresholded counts (on natural scale), thresholds, bins, densities estimated on each bin, and the original data from `thresholdSCRNACountMatrix`

Usage

```
plotMASTThresholdGenes(
  inSCE,
  useAssay = "logcounts",
  isLogged = TRUE,
  check_sanity = TRUE
)
```

Arguments

inSCE	SingleCellExperiment object
useAssay	character, default "logcounts"
isLogged	Logical scalar. Whether the assay used for the analysis is logged. If not, will do a $\log(\text{assay} + 1)$ transformation. Default TRUE.
check_sanity	Logical scalar. Whether to perform MAST's sanity check to see if the counts are logged. Default TRUE

Value

Plot the thresholding onto the plotting region.

Examples

```
data("mouseBrainSubsetSCE")
plotMASTThresholdGenes(mouseBrainSubsetSCE)
```

plotPCA	<i>Plot PCA run data from its components.</i>
---------	---

Description

Plot PCA run data from its components.

Usage

```
plotPCA(
  inSCE,
  colorBy = "No Color",
  shape = "No Shape",
  pcX = "PC1",
  pcY = "PC2",
  reducedDimName = "PCA",
  runPCA = FALSE,
  useAssay = "logcounts"
)
```

Arguments

inSCE	Input SingleCellExperiment object.
colorBy	The variable to color clusters by
shape	Shape of the points
pcX	User choice for the first principal component
pcY	User choice for the second principal component
reducedDimName	a name to store the results of the dimension reduction coordinates obtained from this method. This is stored in the SingleCellExperiment object in the reduced-Dims slot. Required.
runPCA	Run PCA if the reducedDimName does not exist. the Default is FALSE.
useAssay	Indicate which assay to use. The default is "logcounts".

Value

A PCA plot

Examples

```
data("mouseBrainSubsetSCE")
plotPCA(mouseBrainSubsetSCE, colorBy = "level1class",
        reducedDimName = "PCA_counts")
```

plotRunPerCellQCResults

Plots for runPerCellQC outputs.

Description

A wrapper function which visualizes outputs from the runPerCellQC function stored in the colData slot of the SingleCellExperiment object via various plots.

Usage

```
plotRunPerCellQCResults(
  inSCE,
  sample = NULL,
  groupBy = NULL,
  combinePlot = "all",
  violin = TRUE,
  boxplot = FALSE,
  dots = TRUE,
  dotSize = 1,
  axisSize = 15,
  axisLabelSize = 18,
  transparency = 1,
  defaultTheme = TRUE,
  titleSize = 18,
  relHeights = c(1.5, 1.5, 1, 1),
```

```

relWidths = c(1, 1, 1, 1),
plotNcols = NULL,
plotNrows = NULL,
plotLabels = "none",
plotLabelSize = 20,
plotLabelPositionX = NULL,
plotLabelPositionY = NULL,
samplePerColumn = TRUE,
sampleRelHeights = 1,
sampleRelWidths = 1
)

```

Arguments

inSCE	Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results from runPerCellQC. Required.
sample	Character vector. Indicates which sample each cell belongs to. Default NULL.
groupBy	Groupings for each numeric value. A user may input a vector equal length to the number of the samples in the SingleCellExperiment object, or can be retrieved from the colData slot. Default NULL.
combinePlot	Must be either "all", "sample", or "none". "all" will combine all plots into a single .ggplot object, while "sample" will output a list of plots separated by sample. Default "all".
violin	Boolean. If TRUE, will plot the violin plot. Default TRUE.
boxplot	Boolean. If TRUE, will plot boxplots for each violin plot. Default FALSE.
dots	Boolean. If TRUE, will plot dots for each violin plot. Default TRUE.
dotSize	Size of dots. Default 1.
axisSize	Size of x/y-axis ticks. Default 15.
axisLabelSize	Size of x/y-axis labels. Default 18.
transparency	Transparency of the dots, values will be 0-1. Default 1.
defaultTheme	Removes grid in plot and sets axis title size to 10 when TRUE. Default TRUE.
titleSize	Size of title of plot. Default 18.
relHeights	Relative heights of plots when combine is set.
relWidths	Relative widths of plots when combine is set.
plotNcols	Number of columns when plots are combined in a grid.
plotNrows	Number of rows when plots are combined in a grid.
plotLabels	labels to each plot. If set to "default", will use the name of the samples as the labels. If set to "none", no label will be plotted.
plotLabelSize	size of labels
plotLabelPositionX	Numeric vector. The X position of the plot label.
plotLabelPositionY	Numeric vector. The Y position of the plot label.
samplePerColumn	If TRUE, when there are multiple samples and combining by "all", the output .ggplot will have plots from each sample on a single column. Default TRUE.

sampleRelHeights

If there are multiple samples and combining by "all", the relative heights for each plot.

sampleRelWidths

If there are multiple samples and combining by "all", the relative widths for each plot.

Value

list of .ggplot objects

Examples

```
data(scExample, package="singleCellTK")
## Not run:
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
sce <- runPerCellQC(sce)
plotRunPerCellQCResults(inSCE=sce)

## End(Not run)
```

plotScdsHybridResults *Plots for runCxlsBcldsHybrid outputs.*

Description

A wrapper function which visualizes outputs from the runCxlsBcldsHybrid function stored in the colData slot of the SingleCellExperiment object via various plots.

Usage

```
plotScdsHybridResults(
  inSCE,
  sample = NULL,
  shape = NULL,
  groupBy = NULL,
  combinePlot = "all",
  violin = TRUE,
  boxplot = FALSE,
  dots = TRUE,
  reducedDimName = NULL,
  xlab = NULL,
  ylab = NULL,
  dim1 = NULL,
  dim2 = NULL,
  bin = NULL,
  binLabel = NULL,
  defaultTheme = TRUE,
  dotSize = 1,
  transparency = 1,
  titleSize = 18,
  axisLabelSize = 18,
```

```

axisSize = 15,
legendSize = 15,
legendTitleSize = 16,
relHeights = c(1.5, 1, 1),
relWidths = c(1, 1, 1),
plotNCols = NULL,
plotNRows = NULL,
plotLabels = "default",
plotLabelSize = 20,
plotLabelPositionX = NULL,
plotLabelPositionY = NULL,
samplePerColumn = TRUE,
sampleRelHeights = 1,
sampleRelWidths = 1
)

```

Arguments

inSCE	Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results from runCxdscdsHybrid . Required.
sample	Character vector. Indicates which sample each cell belongs to. Default NULL.
shape	If provided, add shapes based on the value.
groupBy	Groupings for each numeric value. A user may input a vector equal length to the number of the samples in the SingleCellExperiment object, or can be retrieved from the colData slot. Default NULL.
combinePlot	Must be either "all", "sample", or "none". "all" will combine all plots into a single .ggplot object, while "sample" will output a list of plots separated by sample. Default "all".
violin	Boolean. If TRUE, will plot the violin plot. Default TRUE.
boxplot	Boolean. If TRUE, will plot boxplots for each violin plot. Default TRUE.
dots	Boolean. If TRUE, will plot dots for each violin plot. Default TRUE.
reducedDimName	Saved dimension reduction name in the SingleCellExperiment object. Required.
xlab	Character vector. Label for x-axis. Default NULL.
ylab	Character vector. Label for y-axis. Default NULL.
dim1	1st dimension to be used for plotting. Can either be a string which specifies the name of the dimension to be plotted from reducedDims, or a numeric value which specifies the index of the dimension to be plotted. Default is NULL.
dim2	2nd dimension to be used for plotting. Can either be a string which specifies the name of the dimension to be plotted from reducedDims, or a numeric value which specifies the index of the dimension to be plotted. Default is NULL.
bin	Numeric vector. If single value, will divide the numeric values into the 'bin' groups. If more than one value, will bin numeric values using values as a cut point.
binLabel	Character vector. Labels for the bins created by the 'bin' parameter. Default NULL.
defaultTheme	Removes grid in plot and sets axis title size to 10 when TRUE. Default TRUE.
dotSize	Size of dots. Default 1.
transparency	Transparency of the dots, values will be 0-1. Default 1.

titleSize	Size of title of plot. Default 18.
axisLabelSize	Size of x/y-axis labels. Default 18.
axisSize	Size of x/y-axis ticks. Default 15.
legendSize	size of legend. Default 15.
legendTitleSize	size of legend title. Default 16.
relHeights	Relative heights of plots when combine is set.
relWidths	Relative widths of plots when combine is set.
plotNCols	Number of columns when plots are combined in a grid.
plotNRows	Number of rows when plots are combined in a grid.
plotLabels	labels to each plot. If set to "default", will use the name of the samples as the labels. If set to "none", no label will be plotted.
plotLabelSize	size of labels
plotLabelPositionX	Numeric vector. The X position of the plot label.
plotLabelPositionY	Numeric vector. The Y position of the plot label.
samplePerColumn	If TRUE, when there are multiple samples and combining by "all", the output .ggplot will have plots from each sample on a single column. Default TRUE.
sampleRelHeights	If there are multiple samples and combining by "all", the relative heights for each plot.
sampleRelWidths	If there are multiple samples and combining by "all", the relative widths for each plot.

Value

list of .ggplot objects

Examples

```
data(scExample, package="singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
sce <- getUMAP(inSCE=sce, useAssay="counts", reducedDimName="UMAP")
sce <- runCxdSBcdsHybrid(sce)
plotScdsHybridResults(inSCE=sce, reducedDimName="UMAP")
```

plotSCEBarAssayData *Bar plot of assay data.*

Description

Visualizes values stored in the assay slot of a SingleCellExperiment object via a bar plot.

Usage

```
plotSCEBarAssayData(
  inSCE,
  sample = NULL,
  useAssay = "counts",
  feature,
  groupBy = NULL,
  xlab = NULL,
  ylab = NULL,
  axisSize = 10,
  axisLabelSize = 10,
  dotSize = 1,
  transparency = 1,
  defaultTheme = TRUE,
  gridLine = FALSE,
  summary = NULL,
  title = NULL,
  titleSize = NULL,
  combinePlot = TRUE
)
```

Arguments

inSCE	Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results. Required.
sample	Character vector. Indicates which sample each cell belongs to.
useAssay	Indicate which assay to use. Default "counts".
feature	Name of feature stored in assay of SingleCellExperiment object.
groupBy	Groupings for each numeric value. A user may input a vector equal length to the number of the samples in the SingleCellExperiment object, or can be retrieved from the colData slot. Default NULL.
xlab	Character vector. Label for x-axis. Default NULL.
ylab	Character vector. Label for y-axis. Default NULL.
axisSize	Size of x/y-axis ticks. Default 10.
axisLabelSize	Size of x/y-axis labels. Default 10.
dotSize	Size of dots. Default 1.
transparency	Transparency of the dots, values will be 0-1. Default 1.
defaultTheme	Removes grid in plot and sets axis title size to 10 when TRUE. Default TRUE.
gridLine	Adds a horizontal grid line if TRUE. Will still be drawn even if defaultTheme is TRUE. Default FALSE.
summary	Adds a summary statistic, as well as a crossbar to the violin plot. Options are "mean" or "median". Default NULL.
title	Title of plot. Default NULL.
titleSize	Size of title of plot. Default 15.
combinePlot	Boolean. If multiple plots are generated (multiple samples, etc.), will combined plots using 'cowplot::plot_grid'. Default TRUE.

Value

a ggplot of the barplot of assay data.

Examples

```
plotSCEBarAssayData(
  inSCE = mouseBrainSubsetSCE,
  feature = "Apoe", groupBy = "sex"
)
```

plotSCEBarColData *Bar plot of colData.*

Description

Visualizes values stored in the colData slot of a SingleCellExperiment object via a bar plot.

Usage

```
plotSCEBarColData(
  inSCE,
  sample = NULL,
  coldata,
  groupBy = NULL,
  dots = TRUE,
  xlab = NULL,
  ylab = NULL,
  axisSize = 10,
  axisLabelSize = 10,
  dotSize = 1,
  transparency = 1,
  defaultTheme = TRUE,
  gridLine = FALSE,
  summary = NULL,
  title = NULL,
  titleSize = NULL,
  combinePlot = TRUE
)
```

Arguments

inSCE	Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results. Required.
sample	Character vector. Indicates which sample each cell belongs to.
coldata	colData value that will be plotted.
groupBy	Groupings for each numeric value. A user may input a vector equal length to the number of the samples in the SingleCellExperiment object, or can be retrieved from the colData slot. Default NULL.
dots	Boolean. If TRUE, will plot dots for each violin plot. Default TRUE.
xlab	Character vector. Label for x-axis. Default NULL.

ylab	Character vector. Label for y-axis. Default NULL.
axisSize	Size of x/y-axis ticks. Default 10.
axisLabelSize	Size of x/y-axis labels. Default 10.
dotSize	Size of dots. Default 1.
transparency	Transparency of the dots, values will be 0-1. Default 1.
defaultTheme	Removes grid in plot and sets axis title size to 10 when TRUE. Default TRUE.
gridLine	Adds a horizontal grid line if TRUE. Will still be drawn even if defaultTheme is TRUE. Default FALSE.
summary	Adds a summary statistic, as well as a crossbar to the violin plot. Options are "mean" or "median". Default NULL.
title	Title of plot. Default NULL.
titleSize	Size of title of plot. Default 15.
combinePlot	Boolean. If multiple plots are generated (multiple samples, etc.), will combined plots using 'cowplot::plot_grid'. Default TRUE.

Value

a ggplot of the barplot of coldata.

Examples

```
plotSCEBarColData(
  inSCE = mouseBrainSubsetSCE,
  coldata = "age", groupBy = "sex"
)
```

plotSCEBatchFeatureMean

Plot mean feature value in each batch of a SingleCellExperiment object

Description

Plot mean feature value in each batch of a SingleCellExperiment object

Usage

```
plotSCEBatchFeatureMean(
  inSCE,
  useAssay = NULL,
  useReddim = NULL,
  useAltExp = NULL,
  batch = "batch",
  xlab = "batch",
  ylab = "Feature Mean",
  ...
)
```

Arguments

inSCE	SingleCellExperiment inherited object.
useAssay	A single character. The name of the assay that stores the value to plot. For useReDim and useAltExp also. Default NULL.
useReDim	A single character. The name of the dimension reduced matrix that stores the value to plot. Default NULL.
useAltExp	A single character. The name of the alternative experiment that stores an assay of the value to plot. Default NULL.
batch	A single character. The name of batch annotation column in colData(inSCE). Default "batch".
xlab	label for x-axis. Default "batch".
ylab	label for y-axis. Default "Feature Mean".
...	Additional arguments passed to .ggViolin .

Value

ggplot

Examples

```
data('sceBatches', package = 'singleCellTK')
plotSCEDensity(sceBatches, useAssay = "logcounts")
```

plotSCEDensity

Density plot of any data stored in the SingleCellExperiment object.

Description

Visualizes values stored in any slot of a SingleCellExperiment object via a density plot.

Usage

```
plotSCEDensity(
  inSCE,
  slot,
  annotation,
  sample = NULL,
  useAssay = "counts",
  feature = NULL,
  groupBy = NULL,
  xlab = NULL,
  ylab = NULL,
  axisSize = 10,
  axisLabelSize = 10,
  defaultTheme = TRUE,
  title = NULL,
  titleSize = 18,
  cutoff = NULL,
  combinePlot = "none",
  plotLabels = NULL
)
```

Arguments

inSCE	Input <code>SingleCellExperiment</code> object with saved dimension reduction components or a variable with saved results. Required
slot	Desired slot of <code>SingleCellExperiment</code> used for plotting. Possible options: "assays", "colData", "metadata"
annotation	Desired vector within the slot used for plotting.
sample	Character vector. Indicates which sample each cell belongs to.
useAssay	Indicate which assay to use. Default "counts".
feature	name of feature stored in assay of <code>SingleCellExperiment</code> object. Will be used only if "assays" slot is chosen. Default NULL.
groupBy	Groupings for each numeric value. A user may input a vector equal length to the number of the samples in the <code>SingleCellExperiment</code> object, or can be retrieved from the colData slot. Default NULL.
xlab	Character vector. Label for x-axis. Default NULL.
ylab	Character vector. Label for y-axis. Default NULL.
axisSize	Size of x/y-axis ticks. Default 10.
axisLabelSize	Size of x/y-axis labels. Default 10.
defaultTheme	Removes grid in plot and sets axis title size to 10 when TRUE. Default TRUE.
title	Title of plot. Default NULL.
titleSize	Size of title of plot. Default 15.
cutoff	Numeric value. The plot will be annotated with a vertical line if set. Default NULL.
combinePlot	Must be either "all", "sample", or "none". "all" will combine all plots into a single <code>.ggplot</code> object, while "sample" will output a list of plots separated by sample. Default "none".
plotLabels	labels to each plot. If set to "default", will use the name of the samples as the labels. If set to "none", no label will be plotted.

Value

a `ggplot` object of the density plot.

Examples

```
plotSCEDensity(
  inSCE = mouseBrainSubsetSCE, slot = "assays",
  annotation = "counts", feature = "ApoE", groupBy = "sex"
)
```

 plotSCEDensityAssayData

Density plot of assay data.

Description

Visualizes values stored in the assay slot of a `SingleCellExperiment` object via a density plot.

Usage

```
plotSCEDensityAssayData(
  inSCE,
  sample = NULL,
  useAssay = "counts",
  feature,
  groupBy = NULL,
  xlab = NULL,
  ylab = NULL,
  axisSize = 10,
  axisLabelSize = 10,
  defaultTheme = TRUE,
  cutoff = NULL,
  title = NULL,
  titleSize = 18,
  combinePlot = "none",
  plotLabels = NULL
)
```

Arguments

<code>inSCE</code>	Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results. Required.
<code>sample</code>	Character vector. Indicates which sample each cell belongs to.
<code>useAssay</code>	Indicate which assay to use. Default "counts".
<code>feature</code>	Name of feature stored in assay of <code>SingleCellExperiment</code> object.
<code>groupBy</code>	Groupings for each numeric value. A user may input a vector equal length to the number of the samples in the <code>SingleCellExperiment</code> object, or can be retrieved from the <code>colData</code> slot. Default NULL.
<code>xlab</code>	Character vector. Label for x-axis. Default NULL.
<code>ylab</code>	Character vector. Label for y-axis. Default NULL.
<code>axisSize</code>	Size of x/y-axis ticks. Default 10.
<code>axisLabelSize</code>	Size of x/y-axis labels. Default 10.
<code>defaultTheme</code>	Removes grid in plot and sets axis title size to 10 when TRUE. Default TRUE.
<code>cutoff</code>	Numeric value. The plot will be annotated with a vertical line if set. Default NULL.
<code>title</code>	Title of plot. Default NULL.
<code>titleSize</code>	Size of title of plot. Default 15.

combinePlot	Must be either "all", "sample", or "none". "all" will combine all plots into a single .ggplot object, while "sample" will output a list of plots separated by sample. Default "none".
plotLabels	labels to each plot. If set to "default", will use the name of the samples as the labels. If set to "none", no label will be plotted.

Value

a ggplot of the density plot of assay data.

Examples

```
plotSCEDensityAssayData(
  inSCE = mouseBrainSubsetSCE,
  feature = "Apoe"
)
```

plotSCEDensityColData *Density plot of colData.*

Description

Visualizes values stored in the colData slot of a SingleCellExperiment object via a density plot.

Usage

```
plotSCEDensityColData(
  inSCE,
  sample = NULL,
  coldata,
  groupBy = NULL,
  xlab = NULL,
  ylab = NULL,
  axisSize = 10,
  axisLabelSize = 10,
  defaultTheme = TRUE,
  title = NULL,
  titleSize = 18,
  cutoff = NULL,
  combinePlot = "none",
  plotLabels = NULL
)
```

Arguments

inSCE	Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results. Required.
sample	Character vector. Indicates which sample each cell belongs to.
coldata	colData value that will be plotted.

groupBy	Groupings for each numeric value. A user may input a vector equal length to the number of the samples in the SingleCellExperiment object, or can be retrieved from the colData slot. Default NULL.
xlab	Character vector. Label for x-axis. Default NULL.
ylab	Character vector. Label for y-axis. Default NULL.
axisSize	Size of x/y-axis ticks. Default 10.
axisLabelSize	Size of x/y-axis labels. Default 10.
defaultTheme	Removes grid in plot and sets axis title size to 10 when TRUE. Default TRUE.
title	Title of plot. Default NULL.
titleSize	Size of title of plot. Default 15.
cutoff	Numeric value. The plot will be annotated with a vertical line if set. Default NULL.
combinePlot	Must be either "all", "sample", or "none". "all" will combine all plots into a single .ggplot object, while "sample" will output a list of plots separated by sample. Default "none".
plotLabels	labels to each plot. If set to "default", will use the name of the samples as the labels. If set to "none", no label will be plotted.

Value

a ggplot of the density plot of colData.

Examples

```
plotSCEDensityColData(
  inSCE = mouseBrainSubsetSCE,
  coldata = "age", groupBy = "sex"
)
```

plotSCEDimReduceColData

Dimension reduction plot tool for colData

Description

Plot results of reduced dimensions data and colors by annotation data stored in the colData slot.

Usage

```
plotSCEDimReduceColData(
  inSCE,
  sample = NULL,
  colorBy,
  groupBy = NULL,
  conditionClass = NULL,
  shape = NULL,
  reducedDimName = NULL,
  xlab = NULL,
  ylab = NULL,
```



```

axisSize = 10,
axisLabelSize = 10,
dim1 = NULL,
dim2 = NULL,
bin = NULL,
binLabel = NULL,
dotSize = 2,
transparency = 1,
colorScale = NULL,
colorLow = "white",
colorMid = "gray",
colorHigh = "blue",
defaultTheme = TRUE,
title = NULL,
titleSize = 15,
labelClusters = TRUE,
legendTitle = NULL,
legendTitleSize = 12,
legendSize = 10,
combinePlot = "none",
plotLabels = NULL
)

```

Arguments

inSCE	Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results. Required
sample	Character vector. Indicates which sample each cell belongs to.
colorBy	Color by a condition(any column of the annotation data). Required.
groupBy	Group by a condition(any column of the annotation data). Default NULL.
conditionClass	Class of the annotation data used in colorBy. Options are NULL, "factor" or "numeric". If NULL, class will default to the original class. Default NULL.
shape	Add shapes to each condition.
reducedDimName	Saved dimension reduction matrix name in the SingleCellExperiment object. Required.
xlab	Character vector. Label for x-axis. Default NULL.
ylab	Character vector. Label for y-axis. Default NULL.
axisSize	Size of x/y-axis ticks. Default 10.
axisLabelSize	Size of x/y-axis labels. Default 10.
dim1	1st dimension to be used for plotting. Can either be a string which specifies the name of the dimension to be plotted from reducedDims, or a numeric value which specifies the index of the dimension to be plotted. Default is NULL.
dim2	2nd dimension to be used for plotting. Can either be a string which specifies the name of the dimension to be plotted from reducedDims, or a numeric value which specifies the index of the dimension to be plotted. Default is NULL.
bin	Numeric vector. If single value, will divide the numeric values into the 'bin' groups. If more than one value, will bin numeric values using values as a cut point.

binLabel	Character vector. Labels for the bins created by the 'bin' parameter. Default NULL.
dotSize	Size of dots. Default 2.
transparency	Transparency of the dots, values will be 0-1. Default 1.
colorScale	Vector. Needs to be same length as the number of unique levels of colorBy. Will be used only if conditionClass = "factor" or "character". Default NULL.
colorLow	Character. A color available from 'colors()'. The color will be used to signify the lowest values on the scale. Default 'white'.
colorMid	Character. A color available from 'colors()'. The color will be used to signify the midpoint on the scale. Default 'gray'.
colorHigh	Character. A color available from 'colors()'. The color will be used to signify the highest values on the scale. Default 'blue'.
defaultTheme	adds grid to plot when TRUE. Default TRUE.
title	Title of plot. Default NULL.
titleSize	Size of title of plot. Default 15.
labelClusters	Logical. Whether the cluster labels are plotted.
legendTitle	title of legend. Default NULL.
legendTitleSize	size of legend title. Default 12.
legendSize	size of legend. Default 10. Default FALSE.
combinePlot	Must be either "all", "sample", or "none". "all" will combine all plots into a single .ggplot object, while "sample" will output a list of plots separated by sample. Default "none".
plotLabels	labels to each plot. If set to "default", will use the name of the samples as the labels. If set to "none", no label will be plotted.

Value

a ggplot of the reduced dimension plot of coldata.

Examples

```
plotSCEDimReduceColData(
  inSCE = mouseBrainSubsetSCE, colorBy = "tissue",
  shape = NULL, conditionClass = "factor",
  reducedDimName = "TSNE_counts",
  xlab = "tSNE1", ylab = "tSNE2", labelClusters = TRUE
)

plotSCEDimReduceColData(
  inSCE = mouseBrainSubsetSCE, colorBy = "age",
  shape = NULL, conditionClass = "numeric",
  reducedDimName = "TSNE_counts", bin = c(-Inf, 20, 25, +Inf),
  xlab = "tSNE1", ylab = "tSNE2", labelClusters = FALSE
)
```

`plotSCEDimReduceFeatures`*Dimension reduction plot tool for assay data*

Description

Plot results of reduced dimensions data and colors by feature data stored in the assays slot.

Usage

```
plotSCEDimReduceFeatures(  
  inSCE,  
  sample = NULL,  
  feature,  
  shape = NULL,  
  reducedDimName,  
  useAssay = "logcounts",  
  xlab = NULL,  
  ylab = NULL,  
  axisSize = 10,  
  axisLabelSize = 10,  
  dim1 = NULL,  
  dim2 = NULL,  
  bin = NULL,  
  binLabel = NULL,  
  dotSize = 2,  
  transparency = 1,  
  colorLow = "white",  
  colorMid = "gray",  
  colorHigh = "blue",  
  defaultTheme = TRUE,  
  title = NULL,  
  titleSize = 15,  
  legendTitle = NULL,  
  legendSize = 10,  
  legendTitleSize = 12,  
  groupBy = NULL,  
  combinePlot = "none",  
  plotLabels = NULL  
)
```

Arguments

<code>inSCE</code>	Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results. Required
<code>sample</code>	Character vector. Indicates which sample each cell belongs to.
<code>feature</code>	name of feature stored in assay of singleCellExperiment object. Plot will be colored based on feature value.
<code>shape</code>	add shapes to each condition. Default NULL.
<code>reducedDimName</code>	saved dimension reduction name in the SingleCellExperiment object. Required.

<code>useAssay</code>	Indicate which assay to use. The default is "logcounts"
<code>xlab</code>	Character vector. Label for x-axis. Default NULL.
<code>ylab</code>	Character vector. Label for y-axis. Default NULL.
<code>axisSize</code>	Size of x/y-axis ticks. Default 10.
<code>axisLabelSize</code>	Size of x/y-axis labels. Default 10.
<code>dim1</code>	1st dimension to be used for plotting. Can either be a string which specifies the name of the dimension to be plotted from <code>reducedDims</code> , or a numeric value which specifies the index of the dimension to be plotted. Default is NULL.
<code>dim2</code>	2nd dimension to be used for plotting. Can either be a string which specifies the name of the dimension to be plotted from <code>reducedDims</code> , or a numeric value which specifies the index of the dimension to be plotted. Default is NULL. Default is second PCA component for PCA data and NULL otherwise.
<code>bin</code>	Numeric vector. If single value, will divide the numeric values into the 'bin' groups. If more than one value, will bin numeric values using values as a cut point.
<code>binLabel</code>	Character vector. Labels for the bins created by the 'bin' parameter. Default NULL.
<code>dotSize</code>	Size of dots. Default 2.
<code>transparency</code>	Transparency of the dots, values will be 0-1. Default 1.
<code>colorLow</code>	Character. A color available from <code>'colors()'</code> . The color will be used to signify the lowest values on the scale. Default 'white'.
<code>colorMid</code>	Character. A color available from <code>'colors()'</code> . The color will be used to signify the midpoint on the scale. Default 'gray'.
<code>colorHigh</code>	Character. A color available from <code>'colors()'</code> . The color will be used to signify the highest values on the scale. Default 'blue'.
<code>defaultTheme</code>	adds grid to plot when TRUE. Default TRUE.
<code>title</code>	Title of plot. Default NULL.
<code>titleSize</code>	Size of title of plot. Default 15.
<code>legendTitle</code>	title of legend. Default NULL.
<code>legendSize</code>	size of legend. Default 10.
<code>legendTitleSize</code>	size of legend title. Default 12.
<code>groupBy</code>	Facet wrap the scatterplot based on value. Default NULL.
<code>combinePlot</code>	Must be either "all", "sample", or "none". "all" will combine all plots into a single <code>.ggplot</code> object, while "sample" will output a list of plots separated by sample. Default "none".
<code>plotLabels</code>	labels to each plot. If set to "default", will use the name of the samples as the labels. If set to "none", no label will be plotted.

Value

a `ggplot` of the reduced dimension plot of feature data.

Examples

```
plotSCEDimReduceFeatures(  
  inSCE = mouseBrainSubsetSCE, feature = "ApoE",  
  shape = NULL, reducedDimName = "TSNE_counts",  
  useAssay = "counts", xlab = "tSNE1", ylab = "tSNE2"  
)
```

plotSCEHeatmap

Plot heatmap of using data stored in SingleCellExperiment Object

Description

Plot heatmap of using data stored in SingleCellExperiment Object

Usage

```
plotSCEHeatmap(  
  inSCE,  
  useAssay = "logcounts",  
  doLog = FALSE,  
  featureIndex = NULL,  
  cellIndex = NULL,  
  featureIndexBy = "rownames",  
  cellIndexBy = "rownames",  
  featureAnnotations = NULL,  
  cellAnnotations = NULL,  
  featureAnnotationColor = NULL,  
  cellAnnotationColor = NULL,  
  rowDataName = NULL,  
  colDataName = NULL,  
  rowSplitBy = NULL,  
  colSplitBy = NULL,  
  rowLabel = FALSE,  
  colLabel = FALSE,  
  rowDend = TRUE,  
  colDend = TRUE,  
  scale = TRUE,  
  trim = c(-2, 2),  
  title = "SCE Heatmap",  
  rowTitle = "Genes",  
  colTitle = "Cells",  
  rowGap = grid::unit(0, "mm"),  
  colGap = grid::unit(0, "mm"),  
  border = FALSE,  
  colorScheme = NULL,  
  ...  
)
```

Arguments

<code>inSCE</code>	<code>SingleCellExperiment</code> inherited object.
<code>useAssay</code>	character. A string indicating the assay name that provides the expression level to plot.
<code>doLog</code>	Logical scalar. Whether to do $\log(\text{assay} + 1)$ transformation on the assay indicated by <code>useAssay</code> . Default FALSE.
<code>featureIndex</code>	A vector that can subset the input SCE object by rows (features). Alternatively, it can be a vector identifying features in another feature list indicated by <code>featureIndexBy</code> . Default NULL.
<code>cellIndex</code>	A vector that can subset the input SCE object by columns (cells). Alternatively, it can be a vector identifying cells in another cell list indicated by <code>featureIndexBy</code> . Default NULL.
<code>featureIndexBy</code>	A single character specifying a column name of <code>rowData(inSCE)</code> , or a vector of the same length as <code>nrow(inSCE)</code> , where we search for the non-rowname feature indices. Default "rownames".
<code>cellIndexBy</code>	A single character specifying a column name of <code>colData(inSCE)</code> , or a vector of the same length as <code>ncol(inSCE)</code> , where we search for the non-rowname cell indices. Default "rownames".
<code>featureAnnotations</code>	<code>data.frame</code> , with rownames containing all the features going to be plotted. Character columns should be factors. Default NULL.
<code>cellAnnotations</code>	<code>data.frame</code> , with rownames containing all the cells going to be plotted. Character columns should be factors. Default NULL.
<code>featureAnnotationColor</code>	A named list. Customized color settings for feature labeling. Should match the entries in the <code>featureAnnotations</code> or <code>rowDataName</code> . For each entry, there should be a list/vector of colors named with categories. Default NULL.
<code>cellAnnotationColor</code>	A named list. Customized color settings for cell labeling. Should match the entries in the <code>cellAnnotations</code> or <code>colDataName</code> . For each entry, there should be a list/vector of colors named with categories. Default NULL.
<code>rowDataName</code>	character. The column name(s) in <code>rowData</code> that need to be added to the annotation. Default NULL.
<code>colDataName</code>	character. The column name(s) in <code>colData</code> that need to be added to the annotation. Default NULL.
<code>rowSplitBy</code>	character. Do semi-heatmap based on the grouping of this(these) annotation(s). Should exist in either <code>rowDataName</code> or <code>names(featureAnnotations)</code> . Default NULL.
<code>colSplitBy</code>	character. Do semi-heatmap based on the grouping of this(these) annotation(s). Should exist in either <code>colDataName</code> or <code>names(cellAnnotations)</code> . Default NULL.
<code>rowLabel</code>	Use a logical for whether to display all the feature names, a single character to display a column of <code>rowData(inSCE)</code> annotation, a vector of the same length as full/subset <code>nrow(inSCE)</code> to display customized info. Default FALSE.
<code>colLabel</code>	Use a logical for whether to display all the cell names, a single character to display a column of <code>colData(inSCE)</code> annotation, a vector of the same length as full/subset <code>ncol(inSCE)</code> to display customized info. Default FALSE.

rowDend	Whether to display row dendrogram. Default TRUE.
colDend	Whether to display column dendrogram. Default TRUE.
scale	Whether to perform z-score scaling on each row. Default TRUE.
trim	A 2-element numeric vector. Values outside of this range will be trimmed to their nearest bound. Default <code>c(-2, 2)</code>
title	The main title of the whole plot. Default "SCE Heatmap"
rowTitle	The subtitle for the rows. Default "Genes".
colTitle	The subtitle for the columns. Default "Cells".
rowGap	A numeric value or a <code>unit</code> object. For the gap size between rows of the splitted heatmap. Default <code>grid::unit(0, 'mm')</code> .
colGap	A numeric value or a <code>unit</code> object. For the gap size between columns of the splitted heatmap. Default <code>grid::unit(0, 'mm')</code> .
border	A logical scalar. Whether to show the border of the heatmap or splitted heatmaps. Default TRUE.
colorScheme	function. A function that generates color code by giving a value. Can be generated by <code>colorRamp2</code> . Default NULL.
...	Other arguments passed to <code>Heatmap</code> .

Value

A `Heatmap` object

Author(s)

Yichen Wang

Examples

```
data(scExample, package = "singleCellTK")
plotSCEHeatmap(sce[1:3,1:3], useAssay = "counts")
```

plotSCEScatter

Dimension reduction plot tool for all types of data

Description

Plot results of reduced dimensions data of counts stored in any slot in the `SingleCellExperiment` object.

Usage

```
plotSCEScatter(
  inSCE,
  slot = NULL,
  sample = NULL,
  annotation,
  feature = NULL,
  groupBy = NULL,
```

```

shape = NULL,
reducedDimName = NULL,
conditionClass = NULL,
xlab = NULL,
ylab = NULL,
axisSize = 10,
axisLabelSize = 10,
dim1 = NULL,
dim2 = NULL,
bin = NULL,
binLabel = NULL,
dotSize = 2,
transparency = 1,
colorLow = "white",
colorMid = "gray",
colorHigh = "blue",
defaultTheme = TRUE,
title = NULL,
titleSize = 15,
labelClusters = TRUE,
legendTitle = NULL,
legendTitleSize = 12,
legendSize = 10,
combinePlot = "none",
plotLabels = NULL
)

```

Arguments

inSCE	Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results. Required
slot	Desired slot of SingleCellExperiment used for plotting. Possible options: "assays", "colData", "metadata", "reducedDims". Default NULL.
sample	Character vector. Indicates which sample each cell belongs to.
annotation	Desired vector within the slot used for plotting. Default NULL.
feature	name of feature stored in assay of SingleCellExperiment object. Will be used only if "assays" slot is chosen. Default NULL.
groupBy	Group by a condition(any column of the annotation data). Default NULL.
shape	add shapes to each condition.
reducedDimName	saved dimension reduction name in the SingleCellExperiment object. Required.
conditionClass	class of the annotation data used in colorBy. Options are NULL, "factor" or "numeric". If NULL, class will default to the original class. Default NULL.
xlab	Character vector. Label for x-axis. Default NULL.
ylab	Character vector. Label for y-axis. Default NULL.
axisSize	Size of x/y-axis ticks. Default 10.
axisLabelSize	Size of x/y-axis labels. Default 10.
dim1	1st dimension to be used for plotting. Can either be a string which specifies the name of the dimension to be plotted from reducedDims, or a numeric value which specifies the index of the dimension to be plotted. Default is NULL.

dim2	2nd dimension to be used for plotting. Can either be a string which specifies the name of the dimension to be plotted from reducedDims, or a numeric value which specifies the index of the dimension to be plotted. Default is NULL. Default is second PCA component for PCA data and NULL otherwise.
bin	Numeric vector. If single value, will divide the numeric values into the 'bin' groups. If more than one value, will bin numeric values using values as a cut point.
binLabel	Character vector. Labels for the bins created by the 'bin' parameter. Default NULL.
dotSize	Size of dots. Default 2.
transparency	Transparency of the dots, values will be 0-1. Default 1.
colorLow	Character. A color available from 'colors()'. The color will be used to signify the lowest values on the scale. Default 'white'.
colorMid	Character. A color available from 'colors()'. The color will be used to signify the midpoint on the scale. Default 'gray'.
colorHigh	Character. A color available from 'colors()'. The color will be used to signify the highest values on the scale. Default 'blue'.
defaultTheme	adds grid to plot when TRUE. Default TRUE.
title	Title of plot. Default NULL.
titleSize	Size of title of plot. Default 15.
labelClusters	Logical. Whether the cluster labels are plotted.
legendTitle	title of legend. Default NULL.
legendTitleSize	size of legend title. Default 12.
legendSize	size of legend. Default 10.
combinePlot	Must be either "all", "sample", or "none". "all" will combine all plots into a single .ggplot object, while "sample" will output a list of plots separated by sample. Default "none".
plotLabels	labels to each plot. If set to "default", will use the name of the samples as the labels. If set to "none", no label will be plotted.

Value

a ggplot of the reduced dimensions.

Examples

```
plotSCEScatter(
  inSCE = mouseBrainSubsetSCE, legendTitle = NULL,
  slot = "assays", annotation = "counts", feature = "ApoE",
  reducedDimName = "TSNE_counts", labelClusters = FALSE
)
```

plotSCEViolin *Violin plot of any data stored in the SingleCellExperiment object.*

Description

Visualizes values stored in any slot of a SingleCellExperiment object via a violin plot.

Usage

```
plotSCEViolin(
  inSCE,
  sample = NULL,
  slot,
  annotation,
  feature,
  groupBy = NULL,
  violin = TRUE,
  boxplot = TRUE,
  dots = TRUE,
  xlab = NULL,
  ylab = NULL,
  axisSize = 10,
  axisLabelSize = 10,
  dotSize = 1,
  transparency = 1,
  defaultTheme = TRUE,
  gridLine = FALSE,
  summary = NULL,
  title = NULL,
  titleSize = NULL,
  combinePlot = "none",
  plotLabels = NULL
)
```

Arguments

inSCE	Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results. Required
sample	Character vector. Indicates which sample each cell belongs to.
slot	Desired slot of SingleCellExperiment used for plotting. Possible options: "assays", "colData", "metadata"
annotation	Desired vector within the slot used for plotting.
feature	name of feature stored in assay of SingleCellExperiment object. Will be used only if "assays" slot is chosen. Default NULL.
groupBy	Groupings for each numeric value. A user may input a vector equal length to the number of the samples in the SingleCellExperiment object, or can be retrieved from the colData slot. Default NULL.
violin	Boolean. If TRUE, will plot the violin plot. Default TRUE.
boxplot	Boolean. If TRUE, will plot boxplots for each violin plot. Default TRUE.

dots	Boolean. If TRUE, will plot dots for each violin plot. Default TRUE.
xlab	Character vector. Label for x-axis. Default NULL.
ylab	Character vector. Label for y-axis. Default NULL.
axisSize	Size of x/y-axis ticks. Default 10.
axisLabelSize	Size of x/y-axis labels. Default 10.
dotSize	Size of dots. Default 1.
transparency	Transparency of the dots, values will be 0-1. Default 1.
defaultTheme	Removes grid in plot and sets axis title size to 10 when TRUE. Default TRUE.
gridLine	Adds a horizontal grid line if TRUE. Will still be drawn even if defaultTheme is TRUE. Default FALSE.
summary	Adds a summary statistic, as well as a crossbar to the violin plot. Options are "mean" or "median". Default NULL.
title	Title of plot. Default NULL.
titleSize	Size of title of plot. Default 15.
combinePlot	Must be either "all", "sample", or "none". "all" will combine all plots into a single .ggplot object, while "sample" will output a list of plots separated by sample. Default "none".
plotLabels	labels to each plot. If set to "default", will use the name of the samples as the labels. If set to "none", no label will be plotted.

Value

a ggplot of the violin plot.

Examples

```
plotSCEViolin(
  inSCE = mouseBrainSubsetSCE, slot = "assays",
  annotation = "counts", feature = "ApoE", groupBy = "sex"
)
```

plotSCEViolinAssayData

Violin plot of assay data.

Description

Visualizes values stored in the assay slot of a SingleCellExperiment object via a violin plot.

Usage

```
plotSCEViolinAssayData(
  inSCE,
  sample = NULL,
  useAssay = "counts",
  feature,
  groupBy = NULL,
  violin = TRUE,
```

```

    boxplot = TRUE,
    dots = TRUE,
    xlab = NULL,
    ylab = NULL,
    axisSize = 10,
    axisLabelSize = 10,
    dotSize = 1,
    transparency = 1,
    defaultTheme = TRUE,
    gridLine = FALSE,
    summary = NULL,
    title = NULL,
    titleSize = NULL,
    combinePlot = "none",
    plotLabels = NULL
  )

```

Arguments

inSCE	Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results. Required.
sample	Character vector. Indicates which sample each cell belongs to.
useAssay	Indicate which assay to use. Default "counts".
feature	Name of feature stored in assay of SingleCellExperiment object.
groupBy	Groupings for each numeric value. A user may input a vector equal length to the number of the samples in the SingleCellExperiment object, or can be retrieved from the colData slot. Default NULL.
violin	Boolean. If TRUE, will plot the violin plot. Default TRUE.
boxplot	Boolean. If TRUE, will plot boxplots for each violin plot. Default TRUE.
dots	Boolean. If TRUE, will plot dots for each violin plot. Default TRUE.
xlab	Character vector. Label for x-axis. Default NULL.
ylab	Character vector. Label for y-axis. Default NULL.
axisSize	Size of x/y-axis ticks. Default 10.
axisLabelSize	Size of x/y-axis labels. Default 10.
dotSize	Size of dots. Default 1.
transparency	Transparency of the dots, values will be 0-1. Default 1.
defaultTheme	Removes grid in plot and sets axis title size to 10 when TRUE. Default TRUE.
gridLine	Adds a horizontal grid line if TRUE. Will still be drawn even if defaultTheme is TRUE. Default FALSE.
summary	Adds a summary statistic, as well as a crossbar to the violin plot. Options are "mean" or "median". Default NULL.
title	Title of plot. Default NULL.
titleSize	Size of title of plot. Default 15.
combinePlot	Must be either "all", "sample", or "none". "all" will combine all plots into a single .ggplot object, while "sample" will output a list of plots separated by sample. Default "none".
plotLabels	labels to each plot. If set to "default", will use the name of the samples as the labels. If set to "none", no label will be plotted.

Value

a ggplot of the violin plot of assay data.

Examples

```
plotSCEViolinAssayData(  
  inSCE = mouseBrainSubsetSCE,  
  feature = "Apoe", groupBy = "sex"  
)
```

```
plotSCEViolinColData  Violin plot of colData.
```

Description

Visualizes values stored in the colData slot of a SingleCellExperiment object via a violin plot.

Usage

```
plotSCEViolinColData(  
  inSCE,  
  sample = NULL,  
  coldata,  
  groupBy = NULL,  
  violin = TRUE,  
  boxplot = TRUE,  
  dots = TRUE,  
  xlab = NULL,  
  ylab = NULL,  
  axisSize = 10,  
  axisLabelSize = 10,  
  dotSize = 1,  
  transparency = 1,  
  defaultTheme = TRUE,  
  gridLine = FALSE,  
  summary = NULL,  
  title = NULL,  
  titleSize = NULL,  
  combinePlot = "none",  
  plotLabels = NULL  
)
```

Arguments

inSCE	Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results. Required.
sample	Character vector. Indicates which sample each cell belongs to.
coldata	colData value that will be plotted.
groupBy	Groupings for each numeric value. A user may input a vector equal length to the number of the samples in the SingleCellExperiment object, or can be retrieved from the colData slot. Default NULL.

violin	Boolean. If TRUE, will plot the violin plot. Default TRUE.
boxplot	Boolean. If TRUE, will plot boxplots for each violin plot. Default TRUE.
dots	Boolean. If TRUE, will plot dots for each violin plot. Default TRUE.
xlab	Character vector. Label for x-axis. Default NULL.
ylab	Character vector. Label for y-axis. Default NULL.
axisSize	Size of x/y-axis ticks. Default 10.
axisLabelSize	Size of x/y-axis labels. Default 10.
dotSize	Size of dots. Default 1.
transparency	Transparency of the dots, values will be 0-1. Default 1.
defaultTheme	Removes grid in plot and sets axis title size to 10 when TRUE. Default TRUE.
gridLine	Adds a horizontal grid line if TRUE. Will still be drawn even if defaultTheme is TRUE. Default FALSE.
summary	Adds a summary statistic, as well as a crossbar to the violin plot. Options are "mean" or "median". Default NULL.
title	Title of plot. Default NULL.
titleSize	Size of title of plot. Default 15.
combinePlot	Must be either "all", "sample", or "none". "all" will combine all plots into a single .ggplot object, while "sample" will output a list of plots separated by sample. Default "none".
plotLabels	labels to each plot. If set to "default", will use the name of the samples as the labels. If set to "none", no label will be plotted.

Value

a ggplot of the violin plot of coldata.

Examples

```
plotSCEViolinColdData(
  inSCE = mouseBrainSubsetSCE,
  coldata = "age", groupBy = "sex"
)
```

plotScrubletResults *Plots for runScrublet outputs.*

Description

A wrapper function which visualizes outputs from the runScrublet function stored in the colData slot of the SingleCellExperiment object via various plots.

Usage

```

plotScrubletResults(
  inSCE,
  sample = NULL,
  shape = NULL,
  groupBy = NULL,
  combinePlot = "all",
  violin = TRUE,
  boxplot = FALSE,
  dots = TRUE,
  reducedDimName,
  xlab = NULL,
  ylab = NULL,
  dim1 = NULL,
  dim2 = NULL,
  bin = NULL,
  binLabel = NULL,
  defaultTheme = TRUE,
  dotSize = 1,
  transparency = 1,
  titleSize = 18,
  axisLabelSize = 18,
  axisSize = 15,
  legendSize = 15,
  legendTitleSize = 16,
  relHeights = c(1.5, 1, 1),
  relWidths = c(1, 1, 1),
  plotNCols = NULL,
  plotNRows = NULL,
  plotLabels = "default",
  plotLabelSize = 20,
  plotLabelPositionX = NULL,
  plotLabelPositionY = NULL,
  samplePerColumn = TRUE,
  sampleRelHeights = 1,
  sampleRelWidths = 1
)

```

Arguments

inSCE	Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results from runScrublet . Required.
sample	Character vector. Indicates which sample each cell belongs to. Default NULL.
shape	If provided, add shapes based on the value.
groupBy	Groupings for each numeric value. A user may input a vector equal length to the number of the samples in the SingleCellExperiment object, or can be retrieved from the colData slot. Default NULL.
combinePlot	Must be either "all", "sample", or "none". "all" will combine all plots into a single .ggplot object, while "sample" will output a list of plots separated by sample. Default "all".
violin	Boolean. If TRUE, will plot the violin plot. Default TRUE.

boxplot	Boolean. If TRUE, will plot boxplots for each violin plot. Default TRUE.
dots	Boolean. If TRUE, will plot dots for each violin plot. Default TRUE.
reducedDimName	Saved dimension reduction name in the SingleCellExperiment object. Required.
xlab	Character vector. Label for x-axis. Default NULL.
ylab	Character vector. Label for y-axis. Default NULL.
dim1	1st dimension to be used for plotting. Can either be a string which specifies the name of the dimension to be plotted from reducedDims, or a numeric value which specifies the index of the dimension to be plotted. Default is NULL.
dim2	2nd dimension to be used for plotting. Can either be a string which specifies the name of the dimension to be plotted from reducedDims, or a numeric value which specifies the index of the dimension to be plotted. Default is NULL.
bin	Numeric vector. If single value, will divide the numeric values into the 'bin' groups. If more than one value, will bin numeric values using values as a cut point.
binLabel	Character vector. Labels for the bins created by the 'bin' parameter. Default NULL.
defaultTheme	Removes grid in plot and sets axis title size to 10 when TRUE. Default TRUE.
dotSize	Size of dots. Default 1.
transparency	Transparency of the dots, values will be 0-1. Default 1.
titleSize	Size of title of plot. Default 18.
axisLabelSize	Size of x/y-axis labels. Default 18.
axisSize	Size of x/y-axis ticks. Default 15.
legendSize	size of legend. Default 15.
legendTitleSize	size of legend title. Default 16.
relHeights	Relative heights of plots when combine is set.
relWidths	Relative widths of plots when combine is set.
plotNcols	Number of columns when plots are combined in a grid.
plotNrows	Number of rows when plots are combined in a grid.
plotLabels	labels to each plot. If set to "default", will use the name of the samples as the labels. If set to "none", no label will be plotted.
plotLabelSize	size of labels
plotLabelPositionX	Numeric vector. The X position of the plot label.
plotLabelPositionY	Numeric vector. The Y position of the plot label.
samplePerColumn	If TRUE, when there are multiple samples and combining by "all", the output .ggplot will have plots from each sample on a single column. Default TRUE.
sampleRelHeights	If there are multiple samples and combining by "all", the relative heights for each plot.
sampleRelWidths	If there are multiple samples and combining by "all", the relative widths for each plot.

Value

list of .ggplot objects

Examples

```
data(scExample, package="singleCellTK")
## Not run:
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
sce <- getUMAP(inSCE=sce, useAssay="counts", reducedDimName="UMAP")
sce <- runScrublet(sce)
plotScrubletResults(inSCE=sce, reducedDimName="UMAP")

## End(Not run)
```

plotTSNE	<i>Plot t-SNE plot on dimensionality reduction data run from t-SNE method.</i>
----------	--

Description

Plot t-SNE plot on dimensionality reduction data run from t-SNE method.

Usage

```
plotTSNE(
  inSCE,
  colorBy = "No Color",
  shape = "No Shape",
  reducedDimName = "TSNE",
  runTSNE = FALSE,
  useAssay = "logcounts"
)
```

Arguments

inSCE	Input SingleCellExperiment object.
colorBy	color by condition.
shape	add shape to each distinct label.
reducedDimName	a name to store the results of the dimension reduction coordinates obtained from this method. This is stored in the SingleCellExperiment object in the reduced-Dims slot. Required.
runTSNE	Run t-SNE if the reducedDimName does not exist. the Default is FALSE.
useAssay	Indicate which assay to use. The default is "logcounts".

Value

A t-SNE plot

Examples

```
data("mouseBrainSubsetSCE")
plotTSNE(mouseBrainSubsetSCE, colorBy = "level1class",
         reducedDimName = "TSNE_counts")
```

plotUMAP

Plot UMAP results either on already run results or run first and then plot.

Description

Plot UMAP results either on already run results or run first and then plot.

Usage

```
plotUMAP(
  inSCE,
  colorBy = "No Color",
  shape = "No Shape",
  reducedDimName = "UMAP",
  runUMAP = FALSE,
  useAssay = "logcounts"
)
```

Arguments

inSCE	Input SingleCellExperiment object with saved dimension reduction components. Required
colorBy	color by a condition(any column of the annotation data).
shape	add shapes to each condition.
reducedDimName	saved dimension reduction name in the SingleCellExperiment object. Required.
runUMAP	If the dimension reduction components are already available set this to FALSE, otherwise set to TRUE. Default is False.
useAssay	Indicate which assay to use. The default is "logcounts"

Value

a UMAP plot of the reduced dimensions.

Examples

```
data(scExample, package = "singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
sce <- getUMAP(inSCE = sce, useAssay = "counts", reducedDimName = "UMAP")
plotUMAP(sce, shape = "No Shape", reducedDimName = "UMAP",
         runUMAP = TRUE, useAssay = "counts")
```

qcInputProcess	<i>Create SingleCellExperiment object from command line input arguments</i>
----------------	---

Description

Create SingleCellExperiment object from command line input arguments

Usage

```
qcInputProcess(
  preproc,
  samplename,
  path,
  raw,
  fil,
  ref,
  rawFile,
  filFile,
  dataType
)
```

Arguments

preproc	Method used to preprocess the data. It's one of the path provided in <code>-preproc</code> argument.
samplename	The sample name of the data. It's one of the path provided in <code>-sample</code> argument.
path	Base path of the dataset. It's one of the path provided in <code>-bash_path</code> argument.
raw	The directory contains droplet matrix, gene and cell barcodes information. It's one of the path provided in <code>-raw_data_path</code> argument.
fil	The directory contains cell matrix, gene and cell barcodes information. It's one of the path provided in <code>-cell_data_path</code> argument.
ref	The name of reference used by cellranger. Only need for CellrangerV2 data.
rawFile	The full path of the RDS file or Matrix file of the raw gene count matrix. It's one of the path provided in <code>-raw_data</code> argument.
filFile	The full path of the RDS file or Matrix file of the cell count matrix. It's one of the path provided in <code>-cell_data</code> argument.
dataType	Type of the input. It can be "Both", "Droplet" or "Cell". It's one of the path provided in <code>-genome</code> argument.

Value

A list of [SingleCellExperiment](#) object containing the droplet or cell data or both, depending on the dataType that users provided.

readSingleCellMatrix *Read single cell expression matrix*

Description

Automatically detect the format of the input file and read the file.

Usage

```
readSingleCellMatrix(  
  file,  
  class = c("Matrix", "matrix"),  
  delayedArray = TRUE,  
  colIndexLocation = NULL,  
  rowIndexLocation = NULL  
)
```

Arguments

file	Path to input file. Supported file endings include .mtx, .txt, .csv, .tab, .tsv, .npz, and their corresponding gzip, bzip2, or xz compressed extensions (*.gz, *.bz2, or *.xz).
class	Character. Class of matrix. One of "Matrix" or "matrix". Specifying "Matrix" will convert to a sparse format which should be used for datasets with large numbers of cells. Default "Matrix".
delayedArray	Boolean. Whether to read the expression matrix as DelayedArray object or not. Default TRUE.
colIndexLocation	Character. For Optimus output, the path to the barcode index .npy file. Used only if file has .npz extension. Default NULL.
rowIndexLocation	Character. For Optimus output, The path to the feature (gene) index .npy file. Used only if file has .npz extension. Default NULL.

Value

A [DelayedArray](#) object or matrix.

Examples

```
mat <- readSingleCellMatrix(system.file("extdata/hgmm_1k_v3_20x20/outs/",  
  "filtered_feature_bc_matrix/matrix.mtx.gz", package = "singleCellTK"))
```

reportCellQC	<i>Get runCellQC .html report</i>
--------------	-----------------------------------

Description

A function to generate .html Rmarkdown report containing the visualizations of the runCellQC function output

Usage

```
reportCellQC(  
  inSCE,  
  output_file = NULL,  
  output_dir = NULL,  
  subTitle = NULL,  
  studyDesign = NULL  
)
```

Arguments

inSCE	A SingleCellExperiment object containing the filtered count matrix with the output from runCellQC function
output_file	name of the generated file. If NULL/default then the output file name will be based on the name of the Rmarkdown template.
output_dir	name of the output directory to save the rendered file. If NULL/default the file is stored to the current working directory
subTitle	subtitle of the QC HTML report. Default is NULL.
studyDesign	description of the data set and experiment design. It would be shown at the top of QC HTML report. Default is NULL.

Value

.html file

Examples

```
data(scExample, package = "singleCellTK")  
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")  
## Not run:  
sce <- runCellQC(sce)  
reportCellQC(inSCE = sce)  
  
## End(Not run)
```

reportDiffExp	<i>Get runDEAnalysis .html report</i>
---------------	---------------------------------------

Description

A function to generate .html Rmarkdown report containing the visualizations of the `runDEAnalysis` function output

Usage

```
reportDiffExp(inSCE, study, output_file = NULL, output_dir = NULL)
```

Arguments

<code>inSCE</code>	A <code>SingleCellExperiment</code> object containing the output from <code>runDEAnalysis</code> function
<code>study</code>	The specific analysis to visualize, used as <code>analysisName</code> argument when running differential expression.
<code>output_file</code>	name of the generated file. If NULL then the output file name will be based on the name of the Rmarkdown template. Default NULL.
<code>output_dir</code>	name of the output directory to save the rendered file. If NULL the file is stored to the current working directory. Default NULL.

Value

.html file

reportDropletQC	<i>Get runDropletQC .html report</i>
-----------------	--------------------------------------

Description

A function to generate .html Rmarkdown report containing the visualizations of the `runDropletQC` function output

Usage

```
reportDropletQC(
  inSCE,
  output_file = NULL,
  output_dir = NULL,
  subTitle = NULL,
  studyDesign = NULL
)
```

Arguments

inSCE	A SingleCellExperiment object containing the full droplet count matrix with the output from runDropletQC function
output_file	name of the generated file. If NULL/default then the output file name will be based on the name of the Rmarkdown template
output_dir	name of the output directory to save the rendered file. If NULL/default the file is stored to the current working directory
subTitle	subtitle of the QC HTML report. Default is NULL.
studyDesign	description of the data set and experiment design. It would be shown at the top of QC HTML report. Default is NULL.

Value

.html file

Examples

```
data(scExample, package = "singleCellTK")
## Not run:
sce <- runDropletQC(sce)
reportDropletQC(inSCE = sce)

## End(Not run)
```

reportQCTool

Get .html report of the output of the selected QC algorithm

Description

A function to generate .html Rmarkdown report for the specified QC algorithm output

Usage

```
reportQCTool(
  inSCE,
  algorithm = c("BarcodeRankDrops", "EmptyDrops", "QCMetrics", "Scrublet",
    "DoubletCells", "Cxls", "Bcxs", "CxlsBcxsHybrid", "DoubletFinder", "DecontX"),
  output_file = NULL,
  output_dir = NULL
)
```

Arguments

inSCE	A SingleCellExperiment object containing the count matrix (full droplets or filtered matrix, depends on the selected QC algorithm) with the output from at least one of these functions: runQCMetrics, runScrublet, runDoubletCells, runCxls, runBcxs, runCxlsBcxsHybrid, runDecontX, runBarcodeRankDrops, runEmptyDrops
algorithm	Character. Specifies which QC algorithm report to generate. Available options are "BarcodeRankDrops", "EmptyDrops", "QCMetrics", "Scrublet", "DoubletCells", "Cxls", "Bcxs", "CxlsBcxsHybrid", "DoubletFinder" and "DecontX".

output_file name of the generated file. If NULL/default then the output file name will be based on the name of the selected QC algorithm name .

output_dir name of the output directory to save the rendered file. If NULL/default the file is stored to the current working directory

Value

.html file

Examples

```
data(scExample, package = "singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
## Not run:
sce <- runDecontX(sce)
sce <- getUMAP(sce)
reportQCTool(inSCE = sce, algorithm = "DecontX")

## End(Not run)
```

retrieveSCEIndex	<i>Retrieve cell/feature index by giving identifiers saved in col/rowData</i>
------------------	---

Description

Originally written in [retrieveFeatureIndex](#). Modified for also retrieving cell indices and only working for [SingleCellExperiment](#) object. This will return indices of features among the rowData/colData. Partial matching (i.e. grepping) can be used.

Usage

```
retrieveSCEIndex(
  inSCE,
  IDs,
  axis,
  by = NULL,
  exactMatch = TRUE,
  firstMatch = TRUE
)
```

Arguments

inSCE	Input SingleCellExperiment object. Required
IDs	Character vector of identifiers for features or cells to find in rowData or colData of inSCE
axis	A character scalar to specify whether to search for features or cells. Use "row", "feature" or "gene" for features; "col" or "cell" for cells.
by	Character. In which column to search for features/cells in rowData/colData. Default NULL for search the rownames/colnames
exactMatch	A logical scalar. Whether to only identify exact matches or to identify partial matches using grep . Default TRUE
firstMatch	A logical scalar. Whether to only identify the first matches or to return all plausible matches. Default TRUE

Value

A unique, non-NA numeric vector of indices for the matching features/cells in `inSCE`.

Author(s)

Yusuke Koga, Joshua Campbell

Examples

```
data(scExample, package = "singleCellTK")
retrieveSCEIndex(inSCE = sce, IDs = "ENSG00000205542",
  axis = "row")
```

runANOVA

Perform differential expression analysis on SCE with ANOVA

Description

Condition specification allows two methods: 1. Index level selection. Arguments `index1` and `index2` will be used. 2. Annotation level selection. Arguments `class`, `classGroup1` and `classGroup2` will be used.

Usage

```
runANOVA(
  inSCE,
  useAssay = "logcounts",
  index1 = NULL,
  index2 = NULL,
  class = NULL,
  classGroup1 = NULL,
  classGroup2 = NULL,
  analysisName,
  groupName1,
  groupName2,
  covariates = NULL,
  onlyPos = FALSE,
  log2fcThreshold = 0.25,
  fdrThreshold = 0.05,
  overwrite = FALSE
)
```

Arguments

<code>inSCE</code>	SingleCellExperiment inherited object.
<code>useAssay</code>	character. A string specifying which assay to use for ANOVA. Default "logcounts".
<code>index1</code>	Any type of indices that can subset a SingleCellExperiment inherited object by cells. Specifies which cells are of interests. Default NULL.

index2	Any type of indices that can subset a SingleCellExperiment inherited object by cells. specifies the control group against those specified by index1. If NULL when using index specification, index1 cells will be compared with all other cells. Default NULL.
class	A vector/factor with <code>ncol(inSCE)</code> elements, or a character scalar that specifies a column name of <code>colData(inSCE)</code> . Default NULL.
classGroup1	a vector specifying which "levels" given in <code>class</code> are of interests. Default NULL.
classGroup2	a vector specifying which "levels" given in <code>class</code> is the control group against those specified by <code>classGroup1</code> . If NULL when using annotation specification, <code>classGroup1</code> cells will be compared with all other cells.
analysisName	A character scalar naming the DEG analysis. Required
groupName1	A character scalar naming the group of interests. Required.
groupName2	A character scalar naming the control group. Required.
covariates	A character vector of additional covariates to use when building the model. All covariates must exist in <code>names(colData(inSCE))</code> . Default NULL.
onlyPos	Whether to only output DEG with positive <code>log2_FC</code> value. Default FALSE.
log2fcThreshold	Only out put DEGs with the absolute values of <code>log2FC</code> greater than this value. Default 0.25
fdrThreshold	Only out put DEGs with FDR value less than this value. Default 0.05
overwrite	A logical scalar. Whether to overwrite result if exists. Default FALSE.

Details

NOTE that ANOVA method does not produce Log2FC value, but P-value and FDR only.

Value

The input [SingleCellExperiment](#) object with `metadata(inSCE)$diffExp` updated with the results: a list named by `analysisName`, with `$groupNames` containing the naming of the two conditions, `$useAssay` storing the assay name that was used for calculation, `$select` storing the cell selection indices (logical) for each condition, `$result` storing a [data.frame](#) of the DEGs summary, and `$method` storing "ANOVA".

Examples

```
data(scExample, package = "singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
library(scater)
sce <- logNormCounts(sce)
sce <- runANOVA(inSCE = sce, groupName1 = "Sample1",
  groupName2 = "Sample2", index1 = 1:20, index2 = 21:40,
  analysisName = "ANOVA", fdrThreshold = NULL)
```

runBarcodeRankDrops *Identify empty droplets using [barcodeRanks](#).*

Description

Run [barcodeRanks](#) on a count matrix provided in a [SingleCellExperiment](#) object. Distinguish between droplets containing cells and ambient RNA in a droplet-based single-cell RNA sequencing experiment.

Usage

```
runBarcodeRankDrops(
  inSCE,
  sample = NULL,
  useAssay = "counts",
  lower = 100,
  fitBounds = NULL,
  df = 20
)
```

Arguments

inSCE	A SingleCellExperiment object. Must contain a raw counts matrix before empty droplets have been removed.
sample	Character vector. Indicates which sample each cell belongs to emptyDrops will be run on cells from each sample separately. If NULL, then all cells will be processed together. Default NULL.
useAssay	A string specifying which assay in the SCE to use.
lower	See emptyDrops for more information. Default 100.
fitBounds	See emptyDrops for more information. Default NULL.
df	See emptyDrops for more information. Default 20.

Value

A [SingleCellExperiment](#) object with the [barcodeRanks](#) output table appended to the [colData](#) slot. The columns include `dropletUtils_BarcodeRank_Knee` and `dropletUtils_BarcodeRank_Knee`. Please refer to the documentation of [barcodeRanks](#) for details.

Examples

```
# The following unfiltered PBMC_1k_v3 data were downloaded from
# https://support.10xgenomics.com/single-cell-gene-expression/datasets/3.0.0
# /pbmc_1k_v3
# Only the top 10 cells with most counts and the last 10 cells with non-zero
# counts are included in this example.
# This example only serves as an proof of concept and a tutorial on how to
# run the function. The results should not be
# used for drawing scientific conclusions.
data(scExample, package = "singleCellTK")
sce <- runBarcodeRankDrops(inSCE = sce)
```

runBBKNN	<i>Apply BBKNN batch effect correction method to SingleCellExperiment object</i>
----------	--

Description

BBKNN, an extremely fast graph-based data integration algorithm. It modifies the neighbourhood construction step to produce a graph that is balanced across all batches of the data.

Usage

```
runBBKNN(
  inSCE,
  useAssay = "logcounts",
  batch = "batch",
  reducedDimName = "BBKNN",
  nComponents = 50L
)
```

Arguments

inSCE	SingleCellExperiment inherited object. Required.
useAssay	A single character indicating the name of the assay requiring batch correction. Default "logcounts".
batch	A single character indicating a field in <code>colData</code> that annotates the batches. Default "batch".
reducedDimName	A single character. The name for the corrected low-dimensional representation. Will be saved to <code>reducedDim(inSCE)</code> . Default "BBKNN".
nComponents	An integer. Number of principle components or the dimensionality, adopted in the pre-PCA-computation step, the BBKNN step (for how many PCs the algorithm takes into account), and the final UMAP combination step where the value represent the dimensionality of the updated reducedDim. Default 50L.

Value

The input [SingleCellExperiment](#) object with `reducedDim(inSCE, reducedDimName)` updated.

References

Krzysztof Polanski et al., 2020

Examples

```
## Not run:
data('sceBatches', package = 'singleCellTK')
sceCorr <- runBBKNN(sceBatches)

## End(Not run)
```

runBcds	<i>Find doublets/multiplets using bcds.</i>
---------	---

Description

A wrapper function for [bcds](#). Annotate doublets/multiplets using a binary classification approach to discriminate artificial doublets from original data. Generate a doublet score for each cell. Infer doublets if `estNdbl` is TRUE.

Usage

```
runBcds(
  inSCE,
  sample = NULL,
  seed = 12345,
  ntop = 500,
  srat = 1,
  verb = FALSE,
  retRes = FALSE,
  nmax = "tune",
  varImp = FALSE,
  estNdbl = FALSE,
  useAssay = "counts"
)
```

Arguments

<code>inSCE</code>	A SingleCellExperiment object. Needs counts in assays slot.
<code>sample</code>	Character vector. Indicates which sample each cell belongs to. bcds will be run on cells from each sample separately. If NULL, then all cells will be processed together. Default NULL.
<code>seed</code>	Seed for the random number generator. Default 12345.
<code>ntop</code>	See bcds for more information. Default 500.
<code>srat</code>	See bcds for more information. Default 1.
<code>verb</code>	See bcds for more information. Default FALSE.
<code>retRes</code>	See bcds for more information. Default FALSE.
<code>nmax</code>	See bcds for more information. Default "tune".
<code>varImp</code>	See bcds for more information. Default FALSE.
<code>estNdbl</code>	See bcds for more information. Default FALSE.
<code>useAssay</code>	A string specifying which assay in the SCE to use.

Value

A [SingleCellExperiment](#) object with [bcds](#) output appended to the `colData` slot. The columns include `bcds_score` and optionally `bcds_call`. Please refer to the documentation of [bcds](#) for details.

Examples

```
data(scExample, package = "singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
sce <- runBcDs(sce)
```

runCellQC

*Perform comprehensive single cell QC***Description**

A wrapper function to run several QC algorithms on a `SingleCellExperiment` object containing cells after empty droplets have been removed.

Usage

```
runCellQC(
  inSCE,
  algorithms = c("QCMetrics", "doubletCells", "cxds", "bcds", "cxds_bcds_hybrid",
    "scrublet", "doubletFinder", "decontX"),
  sample = NULL,
  collectionName = NULL,
  geneSetList = NULL,
  geneSetListLocation = "rownames",
  geneSetCollection = NULL,
  useAssay = "counts",
  seed = 12345,
  paramsList = NULL
)
```

Arguments

<code>inSCE</code>	A SingleCellExperiment object.
<code>algorithms</code>	Character vector. Specify which QC algorithms to run. Available options are "QCMetrics", "scrublet", "doubletCells", "cxds", "bcds", "cxds_bcds_hybrid", and "decontX".
<code>sample</code>	Character vector. Indicates which sample each cell belongs to. Algorithms will be run on cells from each sample separately.
<code>collectionName</code>	Character. Name of a <code>GeneSetCollection</code> obtained by using one of the <code>importGeneSet*</code> functions. Default <code>NULL</code> .
<code>geneSetList</code>	See <code>runPerCellQC</code> . Default <code>NULL</code> .
<code>geneSetListLocation</code>	See <code>runPerCellQC</code> . Default <code>NULL</code> .
<code>geneSetCollection</code>	See <code>runPerCellQC</code> . Default <code>NULL</code> .
<code>useAssay</code>	A string specifying which assay contains the count matrix for cells.
<code>seed</code>	Seed for the random number generator. Default 12345.
<code>paramsList</code>	A list containing parameters for QC functions. Default <code>NULL</code> .

Value

SingleCellExperiment object containing the outputs of the specified algorithms in the `colData` of `inSCE`.

Examples

```
data(scExample, package = "singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
## Not run:
sce <- runCellQC(sce)

## End(Not run)
```

runComBat	<i>Apply ComBat batch effect correction method to SingleCellExperiment object</i>
-----------	---

Description

The ComBat batch adjustment approach assumes that batch effects represent non-biological but systematic shifts in the mean or variability of genomic features for all samples within a processing batch. It uses either parametric or non-parametric empirical Bayes frameworks for adjusting data for batch effects.

Usage

```
runComBat(
  inSCE,
  useAssay = "logcounts",
  batch = "batch",
  par.prior = TRUE,
  covariates = NULL,
  mean.only = FALSE,
  ref.batch = NULL,
  assayName = "ComBat"
)
```

Arguments

<code>inSCE</code>	<code>SingleCellExperiment</code> inherited object. Required.
<code>useAssay</code>	A single character indicating the name of the assay requiring batch correction. Default "logcounts".
<code>batch</code>	A single character indicating a field in <code>colData</code> that annotates the batches. Default "batch".
<code>par.prior</code>	A logical scalar. TRUE indicates parametric adjustments will be used, FALSE indicates non-parametric adjustments will be used. Default TRUE.
<code>covariates</code>	List of other column names in <code>colData</code> to be added to the ComBat model as covariates. Default NULL.
<code>mean.only</code>	If TRUE ComBat only corrects the mean of the batch effect. Default FALSE.

ref.batch	If given, will use the selected batch as a reference for batch adjustment. Default NULL.
assayName	A single character. The name for the corrected assay. Will be saved to assay . Default "ComBat".

Value

The input [SingleCellExperiment](#) object with `assay(inSCE, assayName)` updated.

Examples

```
## Not run:
data('sceBatches', package = 'singleCellTK')
# parametric adjustment
sceCorr <- runComBat(sceBatches)
# non-parametric adjustment, mean-only version
sceCorr <- runComBat(sceBatches, par.prior=FALSE, mean.only=TRUE)
# reference-batch version, with covariates
sceCorr <- runComBat(sceBatches, covariates = "cell_type", ref.batch = 'w')

## End(Not run)
```

runCxls

Find doublets/multiplets using [cxds](#).

Description

A wrapper function for [cxds](#). Annotate doublets/multiplets using co-expression based approach. Generate a doublet score for each cell. Infer doublets if `estNdbl` is TRUE.

Usage

```
runCxls(
  inSCE,
  sample = NULL,
  seed = 12345,
  ntop = 500,
  binThresh = 0,
  verb = FALSE,
  retRes = FALSE,
  estNdbl = FALSE,
  useAssay = "counts"
)
```

Arguments

inSCE	A SingleCellExperiment object. Needs counts in assays slot.
sample	Character vector. Indicates which sample each cell belongs to. cxds will be run on cells from each sample separately. If NULL, then all cells will be processed together. Default NULL.
seed	Seed for the random number generator. Default 12345.

ntop	See cxds for more information. Default 500.
binThresh	See cxds for more information. Default 0.
verb	See cxds for more information. Default FALSE.
retRes	See cxds for more information. Default FALSE.
estNdbl	See cxds for more information. Default FALSE.
useAssay	A string specifying which assay in the SCE to use.

Value

A [SingleCellExperiment](#) object with [cxds](#) output appended to the [colData](#) slot. The columns include [cxds_score](#) and optionally [cxds_call](#). Please refer to the documentation of [cxds](#) for details.

Examples

```
data(scExample, package = "singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
sce <- runCxds(sce)
```

runCxdsBcdsHybrid	<i>Find doublets/multiplets using cxds_bcds_hybrid.</i>
-------------------	---

Description

A wrapper function for [cxds_bcds_hybrid](#). Annotate doublets/multiplets using a binary classification approach to discriminate artificial doublets from original data. Generate a doublet score for each cell. Infer doublets if [estNdbl](#) is TRUE.

Usage

```
runCxdsBcdsHybrid(
  inSCE,
  sample = NULL,
  seed = 12345,
  nTop = 500,
  cxdsArgs = list(),
  bcdsArgs = list(),
  verb = FALSE,
  estNdbl = FALSE,
  force = FALSE,
  useAssay = "counts"
)
```

Arguments

inSCE	A SingleCellExperiment object. Needs counts in assays slot.
sample	Character vector. Indicates which sample each cell belongs to. cxds_bcds_hybrid will be run on cells from each sample separately. If NULL, then all cells will be processed together. Default NULL.
seed	Seed for the random number generator. Default 12345.

nTop	The number of top variable genes to consider. Used in both cnds and bcnds. Default 500.
cxdsArgs	See cxds_bcnds_hybrid for more information. Default NULL.
bcndsArgs	See cxds_bcnds_hybrid for more information. Default NULL.
verb	See cxds_bcnds_hybrid for more information. Default FALSE.
estNdbl	See cxds_bcnds_hybrid for more information. Default FALSE.
force	See cxds_bcnds_hybrid for more information. Default FALSE.
useAssay	A string specifying which assay in the SCE to use.

Value

A [SingleCellExperiment](#) object with [cxds_bcnds_hybrid](#) output appended to the `colData` slot. The columns include `hybrid_score` and optionally `hybrid_call`. Please refer to the documentation of [cxds_bcnds_hybrid](#) for details.

Examples

```
data(scExample, package = "singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
sce <- runCxdsBcndsHybrid(sce)
```

runDEAnalysis	<i>Perform differential expression analysis on SCE with specified method</i> <i>Method supported: 'MAST', 'DESeq2', 'Limma', 'ANOVA'</i>
---------------	---

Description

Perform differential expression analysis on SCE with specified method Method supported: 'MAST', 'DESeq2', 'Limma', 'ANOVA'

Usage

```
runDEAnalysis(method = c("MAST", "DESeq2", "Limma", "ANOVA"), ...)
```

Arguments

method	A single character for specific method. Choose from "MAST", "DESeq2", "Limma", "ANOVA". Default "MAST".
...	Other arguments passed to specific functions. Refer to runMAST , runDESeq2 , runLimmaDE , runANOVA

Value

Input SCE object with `metadata(inSCE)` updated with name "diffExp" as a list object. Detail refers to the four child functions.

Examples

```
data(scExample, package = "singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
sce <- runDEAnalysis(inSCE = sce, groupName1 = "Sample1", method = "DESeq2",
  groupName2 = "Sample2", index1 = 1:20, index2 = 21:40,
  analysisName = "DESeq2")
```

runDecontX

*Detecting contamination with DecontX.***Description**

A wrapper function for [decontX](#). Identify potential contamination from experimental factors such as ambient RNA.

Usage

```
runDecontX(
  inSCE,
  sample = NULL,
  useAssay = "counts",
  z = NULL,
  maxIter = 500,
  delta = c(10, 10),
  estimateDelta = TRUE,
  convergence = 0.001,
  iterLogLik = 10,
  varGenes = 5000,
  dbscanEps = 1,
  seed = 12345,
  logfile = NULL,
  verbose = TRUE
)
```

Arguments

inSCE	A SingleCellExperiment object.
sample	Character vector. Indicates which sample each cell belongs to. Default NULL. decontX will be run on cells from each sample separately.
useAssay	A string specifying which assay in the SCE to use. Default 'counts'.
z	Numeric or character vector. Cell cluster labels. If NULL, PCA will be used to reduce the dimensionality of the dataset initially, 'umap' from the 'uwot' package will be used to further reduce the dataset to 2 dimensions and the 'dbscan' function from the 'dbscan' package will be used to identify clusters of broad cell types. Default NULL.
maxIter	Integer. Maximum iterations of the EM algorithm. Default 500.
delta	Numeric Vector of length 2. Concentration parameters for the Dirichlet prior for the contamination in each cell. The first element is the prior for the native counts while the second element is the prior for the contamination counts. These essentially act as pseudocounts for the native and contamination in each cell. If <code>estimateDelta = TRUE</code> , this is only used to produce a random sample of proportions for an initial value of contamination in each cell. Then fit_dirichlet is used to update <code>delta</code> in each iteration. If <code>estimateDelta = FALSE</code> , then <code>delta</code> is fixed with these values for the entire inference procedure. Fixing <code>delta</code> and setting a high number in the second element will force decontX to be more aggressive and estimate higher levels of contamination at the expense of potentially removing native expression. Default <code>c(10, 10)</code> .

estimateDelta	Boolean. Whether to update delta at each iteration.
convergence	Numeric. The EM algorithm will be stopped if the maximum difference in the contamination estimates between the previous and current iterations is less than this. Default 0.001.
iterLogLik	Integer. Calculate log likelihood every iterLogLik iteration. Default 10.
varGenes	Integer. The number of variable genes to use in dimensionality reduction before clustering. Variability is calculated using <code>modelGeneVar</code> function from the 'scran' package. Used only when z is not provided. Default 5000.
dbscanEps	Numeric. The clustering resolution parameter used in 'dbscan' to estimate broad cell clusters. Used only when z is not provided. Default 1.
seed	Integer. Passed to <code>with_seed</code> . For reproducibility, a default value of 12345 is used. If NULL, no calls to <code>with_seed</code> are made.
logfile	Character. Messages will be redirected to a file named 'logfile'. If NULL, messages will be printed to stdout. Default NULL.
verbose	Logical. Whether to print log messages. Default TRUE.

Value

A `SingleCellExperiment` object with 'decontX_Contamination' and 'decontX_Clusters' added to the `colData` slot. Additionally, the decontaminated counts will be added as an assay called 'decontXCounts'.

Examples

```
data(scExample, package = "singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
sce <- runDecontX(sce)
```

runDESeq2

Perform differential expression analysis on SCE with DESeq2.

Description

Condition specification allows two methods: 1. Index level selection. Arguments `index1` and `index2` will be used. 2. Annotation level selection. Arguments `class`, `classGroup1` and `classGroup2` will be used.

Usage

```
runDESeq2(
  inSCE,
  useAssay = "counts",
  index1 = NULL,
  index2 = NULL,
  class = NULL,
  classGroup1 = NULL,
  classGroup2 = NULL,
  analysisName,
  groupName1,
```

```

    groupName2,
    covariates = NULL,
    fullReduced = TRUE,
    onlyPos = FALSE,
    log2fcThreshold = NULL,
    fdrThreshold = 1,
    overwrite = FALSE
  )

```

Arguments

<code>inSCE</code>	SingleCellExperiment inherited object.
<code>useAssay</code>	character. A string specifying which assay to use for the DESeq2 regression. Default "logcounts".
<code>index1</code>	Any type of indices that can subset a SingleCellExperiment inherited object by cells. Specifies which cells are of interests. Default NULL.
<code>index2</code>	Any type of indices that can subset a SingleCellExperiment inherited object by cells. specifies the control group against those specified by <code>index1</code> . If NULL when using index specification, <code>index1</code> cells will be compared with all other cells. Default NULL.
<code>class</code>	A vector/factor with <code>ncol(inSCE)</code> elements, or a character scalar that specifies a column name of <code>colData(inSCE)</code> . Default NULL.
<code>classGroup1</code>	a vector specifying which "levels" given in <code>class</code> are of interests. Default NULL.
<code>classGroup2</code>	a vector specifying which "levels" given in <code>class</code> is the control group against those specified by <code>classGroup1</code> . If NULL when using annotation specification, <code>classGroup1</code> cells will be compared with all other cells.
<code>analysisName</code>	A character scalar naming the DEG analysis. Required
<code>groupName1</code>	A character scalar naming the group of interests. Required.
<code>groupName2</code>	A character scalar naming the control group. Required.
<code>covariates</code>	A character vector of additional covariates to use when building the model. All covariates must exist in <code>names(colData(inSCE))</code> . Default NULL.
<code>fullReduced</code>	Whether to apply LRT (Likelihood ratio test) with a 'full' model. Default TRUE.
<code>onlyPos</code>	Whether to only output DEG with positive <code>log2_FC</code> value. Default FALSE.
<code>log2fcThreshold</code>	Only out put DEGs with the absolute values of <code>log2FC</code> greater than this value. Default 0.25
<code>fdrThreshold</code>	Only out put DEGs with FDR value less than this value. Default 0.05
<code>overwrite</code>	A logical scalar. Whether to overwrite result if exists. Default FALSE.

Value

The input [SingleCellExperiment](#) object with `metadata(inSCE)$DESeq2` updated with the results: a list named by `analysisName`, with `$groupNames` containing the naming of the two conditions, `$useAssay` storing the assay name that was used for calculation, `$select` storing the cell selection indices (logical) for each condition, `$result` storing a [data.frame](#) of the DEGs summary, and `$method` storing "DESeq2".

Examples

```
data(scExample, package = "singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
sce <- runDESeq2(inSCE = sce, groupName1 = "Sample1",
  groupName2 = "Sample2", index1 = 1:20, index2 = 21:40,
  analysisName = "DESeq2")
```

runDoubletCells *Detect doublet cells using [scDbfFinder](#).*

Description

A wrapper function for [scDbfFinder](#). Identify potential doublet cells based on simulations of putative doublet expression profiles. Generate a doublet score for each cell.

Usage

```
runDoubletCells(
  inSCE,
  sample = NULL,
  useAssay = "counts",
  nNeighbors = 50,
  simDoublets = max(10000, ncol(inSCE)),
  seed = 12345,
  BPPARAM = BiocParallel::SerialParam()
)
```

Arguments

inSCE	A SingleCellExperiment object.
sample	Character vector. Indicates which sample each cell belongs to. scDbfFinder will be run on cells from each sample separately.
useAssay	A string specifying which assay in the SCE to use.
nNeighbors	Number of nearest neighbors used to calculate density for doublet detection. Default 50.
simDoublets	Number of simulated doublets created for doublet detection. Default 10000.
seed	Seed for the random number generator. Default 12345.
BPPARAM	A BiocParallelParam object specifying whether the neighbour searches should be parallelized.

Details

This function is a wrapper function for [scDbfFinder](#). `runDoubletCells` runs [scDbfFinder](#) for each sample within `inSCE` iteratively. The resulting doublet scores for all cells will be appended to the `colData` of `inSCE`.

Value

A [SingleCellExperiment](#) object with the 'scran_doubletCells_score' column added to the `colData` slot.

References

Lun ATL (2018). Detecting doublet cells with scran. https://l1la.github.io/SingleCellThoughts/software/doublet_detection/bycell.html

See Also

[doubletCells](#)

Examples

```
data(scExample, package = "singleCellTK")
sce <- subsetSCEcols(sce, colData = "type != 'EmptyDroplet'")
sce <- runDoubletCells(sce)
```

runDoubletFinder	<i>Generates a doublet score for each cell via doubletFinder</i>
------------------	--

Description

Uses doubletFinder to determine cells within the dataset suspected to be doublets.

Usage

```
runDoubletFinder(
  inSCE,
  useAssay = "counts",
  sample = NULL,
  seed = 12345,
  seuratNfeatures = 2000,
  seuratPcs = 1:15,
  seuratRes = 1.5,
  formationRate = 0.075,
  nCores = NULL,
  verbose = FALSE
)
```

Arguments

inSCE	Input SingleCellExperiment object. Must contain a counts matrix
useAssay	Indicate which assay to use. Default "counts".
sample	Numeric vector. Each cell will be assigned a sample number.
seed	Seed for the random number generator. Default 12345.
seuratNfeatures	Integer. Number of highly variable genes to use. Default 2000.
seuratPcs	Numeric vector. The PCs used in seurat function to determine number of clusters. Default 1:15.
seuratRes	Numeric vector. The resolution parameter used in seurat, which adjusts the number of clusters determined via the algorithm. Default 1.5.
formationRate	Doublet formation rate used within algorithm. Default 0.075.

nCores	Number of cores used for running the function.
verbose	Boolean. Whether to print messages from Seurat and DoubletFinder. Default FALSE.

Value

SingleCellExperiment object containing the 'doublet_finder_doublet_score'.

Examples

```
data(scExample, package = "singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
sce <- runDoubletFinder(sce)
```

runDropletQC	<i>Perform comprehensive droplet QC</i>
--------------	---

Description

A wrapper function to run several QC algorithms for determining empty droplets in single cell RNA-seq data

Usage

```
runDropletQC(
  inSCE,
  algorithms = c("QCMetrics", "emptyDrops", "barcodeRanks"),
  sample = NULL,
  useAssay = "counts",
  paramsList = NULL
)
```

Arguments

inSCE	A SingleCellExperiment object containing the full droplet count matrix
algorithms	Character vector. Specify which QC algorithms to run. Available options are "emptyDrops" and "barcodeRanks".
sample	Character vector. Indicates which sample each cell belongs to. Algorithms will be run on cells from each sample separately.
useAssay	A string specifying which assay contains the count matrix for droplets.
paramsList	A list containing parameters for QC functions. Default NULL.

Value

SingleCellExperiment object containing the outputs of the specified algorithms in the `colData` of `inSCE`.

Examples

```
data(scExample, package = "singleCellTK")
## Not run:
sce <- runDropletQC(sce)

## End(Not run)
```

runEmptyDrops	<i>Identify empty droplets using emptyDrops.</i>
---------------	--

Description

Run [emptyDrops](#) on the count matrix in the provided [SingleCellExperiment](#) object. Distinguish between droplets containing cells and ambient RNA in a droplet-based single-cell RNA sequencing experiment.

Usage

```
runEmptyDrops(
  inSCE,
  sample = NULL,
  useAssay = "counts",
  lower = 100,
  niters = 10000,
  testAmbient = FALSE,
  ignore = NULL,
  alpha = NULL,
  retain = NULL,
  barcodeArgs = list(),
  BPPARAM = BiocParallel::SerialParam()
)
```

Arguments

inSCE	Input SingleCellExperiment object. Must contain a raw counts matrix before empty droplets have been removed.
sample	Character vector. Indicates which sample each cell belongs to. emptyDrops will be run on cells from each sample separately. If NULL, then all cells will be processed together. Default NULL.
useAssay	A string specifying which assay in the SCE to use.
lower	See emptyDrops for more information.
niters	See emptyDrops for more information.
testAmbient	See emptyDrops for more information.
ignore	See emptyDrops for more information.
alpha	See emptyDrops for more information.
retain	See emptyDrops for more information.
barcodeArgs	See emptyDrops for more information.
BPPARAM	See emptyDrops for more information.

Value

A [SingleCellExperiment](#) object with the [emptyDrops](#) output table appended to the [colData](#) slot. The columns include *emptyDrops_total*, *emptyDrops_logprob*, *emptyDrops_pvalue*, *emptyDrops_limited*, *emptyDrops_fdr*. Please refer to the documentation of [emptyDrops](#) for details.

Examples

```
# The following unfiltered PBMC_1k_v3 data were downloaded from
# https://support.10xgenomics.com/single-cell-gene-expression/datasets/3.0.0
# /pbmc_1k_v3
# Only the top 10 cells with most counts and the last 10 cells with non-zero
# counts are included in this example.
# This example only serves as an proof of concept and a tutorial on how to
# run the function. The results should not be
# used for drawing scientific conclusions.
data(scExample, package = "singleCellTK")
sce <- runEmptyDrops(inSCE = sce)
```

runFastMNN

Apply a fast version of the mutual nearest neighbors (MNN) batch effect correction method to SingleCellExperiment object

Description

fastMNN is a variant of the classic MNN method, modified for speed and more robust performance. For introduction of MNN, see [runMNNCorrect](#).

Usage

```
runFastMNN(
  inSCE,
  useAssay = "logcounts",
  reducedDimName = "fastMNN",
  batch = "batch",
  pcInput = FALSE
)
```

Arguments

inSCE	inherited object. Required.
useAssay	A single character indicating the name of the assay requiring batch correction. Default "logcounts". Alternatively, see pcInput parameter.
reducedDimName	A single character. The name for the corrected low-dimensional representation. Will be saved to reducedDim(inSCE). Default "fastMNN".
batch	A single character indicating a field in colData that annotates the batches. Default "batch".
pcInput	A logical scalar. Whether to use a low-dimension matrix for batch effect correction. If TRUE, useAssay will be searched from reducedDimNames(inSCE). Default FALSE.

Value

The input [SingleCellExperiment](#) object with `reducedDim(inSCE, reducedDimName)` updated.

References

Lun ATL, et al., 2016

Examples

```
## Not run:
data('sceBatches', package = 'singleCellTK')
sceCorr <- runFastMNN(sceBatches, useAssay = 'logcounts', pcInput = FALSE)

## End(Not run)
```

runKMeans

Get clustering with KMeans

Description

Perform KMeans clustering on a [SingleCellExperiment](#) object, with `kmeans`.

Usage

```
runKMeans(
  inSCE,
  useReducedDim = "PCA",
  clusterName = "KMeans_cluster",
  nCenters,
  nIter = 10,
  nStart = 1,
  seed = 12345,
  algorithm = c("Hartigan-Wong", "Lloyd", "MacQueen")
)
```

Arguments

<code>inSCE</code>	A SingleCellExperiment object.
<code>useReducedDim</code>	A single character, specifying which low-dimension representation to perform the clustering algorithm on. Default "PCA".
<code>clusterName</code>	A single character, specifying the name to store the cluster label in <code>colData</code> . Default "scransNN_cluster".
<code>nCenters</code>	An integer, the number of centroids (clusters).
<code>nIter</code>	An integer, the maximum number of iterations allowed. Default 10.
<code>nStart</code>	An integer, the number of random sets to choose. Default 1.
<code>seed</code>	An integer. The seed for the random number generator. Default 12345.
<code>algorithm</code>	A single character. Choose from "Hartigan-Wong", "Lloyd", "MacQueen". May be abbreviated. Default "Hartigan-Wong".

Value

The input `SingleCellExperiment` object with factor cluster labeling updated in `colData(inSCE)[[clusterName]]`.

Examples

```
data("mouseBrainSubsetSCE")
mouseBrainSubsetSCE <- runKMeans(mouseBrainSubsetSCE,
                                useReducedDim = "PCA_logcounts",
                                nCenters = 2)
```

runLimmaBC	<i>Apply Limma's batch effect correction method to SingleCellExperiment object</i>
------------	--

Description

Limma's batch effect removal function fits a linear model to the data, then removes the component due to the batch effects.

Usage

```
runLimmaBC(inSCE, useAssay = "logcounts", assayName = "LIMMA", batch = "batch")
```

Arguments

inSCE	<code>SingleCellExperiment</code> inherited object. Required.
useAssay	A single character indicating the name of the assay requiring batch correction. Default "logcounts".
assayName	A single character. The name for the corrected assay. Will be saved to <code>assay</code> . Default "LIMMA".
batch	A single character indicating a field in <code>colData</code> that annotates the batches. Default "batch".

Value

The input `SingleCellExperiment` object with `assay(inSCE, assayName)` updated.

References

Gordon K Smyth, et al., 2003

Examples

```
data('sceBatches', package = 'singleCellTK')
## Not run:
sceCorr <- runLimmaBC(sceBatches)

## End(Not run)
```

runLimmaDE

*Perform differential expression analysis on SCE with Limma.***Description**

Condition specification allows two methods: 1. Index level selection. Arguments `index1` and `index2` will be used. 2. Annotation level selection. Arguments `class`, `classGroup1` and `classGroup2` will be used.

Usage

```
runLimmaDE(
  inSCE,
  useAssay = "logcounts",
  index1 = NULL,
  index2 = NULL,
  class = NULL,
  classGroup1 = NULL,
  classGroup2 = NULL,
  analysisName,
  groupName1,
  groupName2,
  covariates = NULL,
  onlyPos = FALSE,
  log2fcThreshold = 0.25,
  fdrThreshold = 0.05,
  overwrite = FALSE
)
```

Arguments

<code>inSCE</code>	SingleCellExperiment inherited object.
<code>useAssay</code>	character. A string specifying which assay to use for the Limma regression. Default "logcounts".
<code>index1</code>	Any type of indices that can subset a SingleCellExperiment inherited object by cells. Specifies which cells are of interests. Default NULL.
<code>index2</code>	Any type of indices that can subset a SingleCellExperiment inherited object by cells. specifies the control group against those specified by <code>index1</code> . If NULL when using index specification, <code>index1</code> cells will be compared with all other cells. Default NULL.
<code>class</code>	A vector/factor with <code>ncol(inSCE)</code> elements, or a character scalar that specifies a column name of <code>colData(inSCE)</code> . Default NULL.
<code>classGroup1</code>	a vector specifying which "levels" given in <code>class</code> are of interests. Default NULL.
<code>classGroup2</code>	a vector specifying which "levels" given in <code>class</code> is the control group against those specified by <code>classGroup1</code> . If NULL when using annotation specification, <code>classGroup1</code> cells will be compared with all other cells.
<code>analysisName</code>	A character scalar naming the DEG analysis. Required
<code>groupName1</code>	A character scalar naming the group of interests. Required.

groupName2	A character scalar naming the control group. Required.
covariates	A character vector of additional covariates to use when building the model. All covariates must exist in <code>names(colData(inSCE))</code> . Default NULL.
onlyPos	Whether to only output DEG with positive <code>log2_FC</code> value. Default FALSE.
log2fcThreshold	Only out put DEGs with the absolute values of <code>log2FC</code> greater than this value. Default 0.25
fdrThreshold	Only out put DEGs with FDR value less than this value. Default 0.05
overwrite	A logical scalar. Whether to overwrite result if exists. Default FALSE.

Value

The input `SingleCellExperiment` object with `metadata(inSCE)$diffExp` updated with the results: a list named by `analysisName`, with `$groupNames` containing the naming of the two conditions, `$useAssay` storing the assay name that was used for calculation, `$select` storing the cell selection indices (logical) for each condition, `$result` storing a `data.frame` of the DEGs summary, and `$method` storing "Limma".

Examples

```
data(scExample, package = "singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
library(scater)
sce <- logNormCounts(sce)
sce <- runLimmaDE(inSCE = sce, groupName1 = "Sample1",
  groupName2 = "Sample2", index1 = 1:20, index2 = 21:40,
  analysisName = "Limma")
```

runMAST

Perform differential expression analysis on SCE with MAST

Description

Condition specification allows two methods: 1. Index level selection. Arguments `index1` and `index2` will be used. 2. Annotation level selection. Arguments `class`, `classGroup1` and `classGroup2` will be used.

Usage

```
runMAST(
  inSCE,
  useAssay = "logcounts",
  index1 = NULL,
  index2 = NULL,
  class = NULL,
  classGroup1 = NULL,
  classGroup2 = NULL,
  analysisName,
  groupName1,
  groupName2,
```

```

covariates = NULL,
onlyPos = FALSE,
log2fcThreshold = NULL,
fdrThreshold = 0.05,
overwrite = FALSE,
check_sanity = TRUE
)

```

Arguments

inSCE	SingleCellExperiment inherited object.
useAssay	character. A string specifying which assay to use for MAST Default "logcounts".
index1	Any type of indices that can subset a SingleCellExperiment inherited object by cells. Specifies which cells are of interests. Default NULL.
index2	Any type of indices that can subset a SingleCellExperiment inherited object by cells. specifies the control group against those specified by index1. If NULL when using index specification, index1 cells will be compared with all other cells. Default NULL.
class	A vector/factor with <code>ncol(inSCE)</code> elements, or a character scalar that specifies a column name of <code>colData(inSCE)</code> . Default NULL.
classGroup1	a vector specifying which "levels" given in <code>class</code> are of interests. Default NULL.
classGroup2	a vector specifying which "levels" given in <code>class</code> is the control group against those specified by <code>classGroup1</code> . If NULL when using annotation specification, <code>classGroup1</code> cells will be compared with all other cells.
analysisName	A character scalar naming the DEG analysis. Required
groupName1	A character scalar naming the group of interests. Required.
groupName2	A character scalar naming the control group. Required.
covariates	A character vector of additional covariates to use when building the model. All covariates must exist in <code>names(colData(inSCE))</code> . Default NULL.
onlyPos	Whether to only output DEG with positive <code>log2_FC</code> value. Default FALSE.
log2fcThreshold	Only out put DEGs with the absolute values of <code>log2FC</code> greater than this value. Default 0.25
fdrThreshold	Only out put DEGs with FDR value less than this value. Default 0.05
overwrite	A logical scalar. Whether to overwrite result if exists. Default FALSE.
check_sanity	Logical scalar. Whether to perform MAST's sanity check to see if the counts are logged. Default TRUE.

Value

The input [SingleCellExperiment](#) object with `metadata(inSCE)$diffExp` updated with the results: a list named by `analysisName`, with `$groupNames` containing the naming of the two conditions, `$useAssay` storing the assay name that was used for calculation, `$select` storing the cell selection indices (logical) for each condition, `$result` storing a [data.frame](#) of the DEGs summary, and `$method` storing "MAST".

Examples

```
data(scExample, package = "singleCellTK")
sce <- subsetSCEcols(sce, colData = "type != 'EmptyDroplet'")
logcounts(sce) <- log(counts(sce) + 1)
sce <- runMAST(inSCE = sce, groupName1 = "Sample1",
              groupName2 = "Sample2", index1 = 1:20, index2 = 21:40,
              analysisName = "MAST")
```

runMNNCorrect	<i>Apply the mutual nearest neighbors (MNN) batch effect correction method to SingleCellExperiment object</i>
---------------	---

Description

MNN is designed for batch correction of single-cell RNA-seq data where the batches are partially confounded with biological conditions of interest. It does so by identifying pairs of MNN in the high-dimensional log-expression space. For each MNN pair, a pairwise correction vector is computed by applying a Gaussian smoothing kernel with bandwidth 'sigma'.

Usage

```
runMNNCorrect(
  inSCE,
  useAssay = "logcounts",
  batch = "batch",
  assayName = "MNN",
  k = 20L,
  sigma = 0.1
)
```

Arguments

inSCE	SingleCellExperiment inherited object. Required.
useAssay	A single character indicating the name of the assay requiring batch correction. Default "logcounts".
batch	A single character indicating a field in colData that annotates the batches. Default "batch".
assayName	A single character. The name for the corrected assay. Will be saved to assay . Default "MNN".
k	An integer. Specifies the number of nearest neighbours to consider when defining MNN pairs. This should be interpreted as the minimum frequency of each cell type or state in each batch. Larger values will improve the precision of the correction by increasing the number of MNN pairs, at the cost of reducing accuracy by allowing MNN pairs to form between cells of different type. Default 20L.
sigma	A Numeric scalar. Specifies how much information is shared between MNN pairs when computing the batch effect. Larger values will share more information, approaching a global correction for all cells in the same batch. Smaller values allow the correction to vary across cell types, which may be more accurate but comes at the cost of precision. Default 0.1.

Value

The input [SingleCellExperiment](#) object with assay(inSCE, assayName) updated.

References

Lun ATL, et al., 2016 & 2018

Examples

```
data('sceBatches', package = 'singleCellTK')
## Not run:
sceCorr <- runMNNCorrect(sceBatches)

## End(Not run)
```

runPerCellQC

Wrapper for calculating QC metrics with scater.

Description

A wrapper function for [decontX](#). Identify potential contamination from experimental factors such as ambient RNA.

Usage

```
runPerCellQC(
  inSCE,
  useAssay = "counts",
  collectionName = NULL,
  geneSetList = NULL,
  geneSetListLocation = "rownames",
  geneSetCollection = NULL,
  percent_top = c(50, 100, 200, 500),
  use_altexps = FALSE,
  flatten = TRUE,
  detectionLimit = 0,
  BPPARAM = BiocParallel::SerialParam()
)
```

Arguments

inSCE	Input SingleCellExperiment object.
useAssay	A string specifying which assay in the SCE to use. Default "counts".
collectionName	Character. Name of a GeneSetCollection obtained by using one of the import-GeneSet* functions. Default NULL.
geneSetList	List of gene sets to be quantified. The genes in the assays will be matched to the genes in the list based on geneSetListLocation. Default NULL.

<code>geneSetListLocation</code>	Character or numeric vector. If set to 'rownames', then the genes in 'geneSetList' will be looked up in <code>rownames(inSCE)</code> . If another character is supplied, then genes will be looked up in the column names of <code>rowData(inSCE)</code> . A character vector with the same length as <code>geneSetList</code> can be supplied if the IDs for different gene sets are found in different places, including a mixture of 'rownames' and <code>rowData(inSCE)</code> . An integer or integer vector can be supplied to denote the column index in <code>rowData(inSCE)</code> . Default 'rownames'.
<code>geneSetCollection</code>	Class of <code>GeneSetCollection</code> from package <code>GSEAbase</code> . The location of the gene IDs in <code>inSCE</code> should be in the description slot of each gene set and should follow the same notation as <code>geneSetListLocation</code> . The function <code>getGmt</code> can be used to read in gene sets from a GMT file. If reading a GMT file, the second column for each gene set should be the description denoting the location of the gene IDs in <code>inSCE</code> . These gene sets will be included with those from <code>geneSetList</code> if both parameters are provided.
<code>percent_top</code>	An integer vector. Each element is treated as a number of top genes to compute the percentage of library size occupied by the most highly expressed genes in each cell.
<code>use_altexprs</code>	Logical scalar indicating whether QC statistics should be computed for alternative Experiments in <code>x</code> . If <code>TRUE</code> , statistics are computed for all alternative experiments. Alternatively, an integer or character vector specifying the alternative Experiments to use to compute QC statistics. Alternatively <code>NULL</code> , in which case alternative experiments are not used.
<code>flatten</code>	Logical scalar indicating whether the nested <code>DataFrame-class</code> in the output should be flattened.
<code>detectionLimit</code>	A numeric scalar specifying the lower detection limit for expression.
<code>BPPARAM</code>	A <code>BiocParallelParam</code> object specifying whether the QC calculations should be parallelized.

Value

A `SingleCellExperiment` object with cell QC metrics added to the `colData` slot. If `geneSetList` or `geneSetCollection` are provided, then the rownames for each gene set will be saved in `metadata(inSCE)$scatter$addl`.

Examples

```
data(scExample, package = "singleCellTK")
mito.ix = grep("^MT-", rowData(sce)$feature_name)
geneSet <- list("Mito"=rownames(sce)[mito.ix])
sce <- runPerCellQC(sce, geneSetList = geneSet)
```

<code>runSCANORAMA</code>	<i>Apply the mutual nearest neighbors (MNN) batch effect correction method to <code>SingleCellExperiment</code> object</i>
---------------------------	--

Description

SCANORAMA is analogous to computer vision algorithms for panorama stitching that identify images with overlapping content and merge these into a larger panorama.

Usage

```
runSCANORAMA(  
  inSCE,  
  useAssay = "logcounts",  
  batch = "batch",  
  SIGMA = 15,  
  ALPHA = 0.1,  
  KNN = 20L,  
  assayName = "SCANORAMA"  
)
```

Arguments

inSCE	SingleCellExperiment inherited object. Required.
useAssay	A single character indicating the name of the assay requiring batch correction. Default "logcounts".
batch	A single character indicating a field in colData that annotates the batches. Default "batch".
SIGMA	A numeric scalar. Algorithmic parameter, correction smoothing parameter on Gaussian kernel. Default 15.
ALPHA	A numeric scalar. Algorithmic parameter, alignment score minimum cutoff. Default 0.1.
KNN	An integer. Algorithmic parameter, number of nearest neighbors to use for matching. Default 20L.
assayName	A single character. The name for the corrected assay. Will be saved to assay . Default "SCANORAMA".

Value

The input [SingleCellExperiment](#) object with `assay(inSCE, assayName)` updated.

References

Brian Hie et al, 2019

Examples

```
## Not run:  
data('sceBatches', package = 'singleCellTK')  
sceCorr <- runSCANORAMA(sceBatches)  
  
## End(Not run)
```

runSCMerge	<i>Apply scMerge batch effect correction method to SingleCellExperiment object</i>
------------	--

Description

The scMerge method leverages factor analysis, stably expressed genes (SEGs) and (pseudo-) replicates to remove unwanted variations and merge multiple scRNA-Seq data.

Usage

```
runSCMerge(
  inSCE,
  useAssay = "logcounts",
  batch = "batch",
  assayName = "scMerge",
  seg = NULL,
  kmeansK = NULL,
  cellType = "cell_type",
  nCores = 1L
)
```

Arguments

inSCE	SingleCellExperiment inherited object. Required.
useAssay	A single character indicating the name of the assay requiring batch correction. Default "logcounts".
batch	A single character indicating a field in colData that annotates the batches. Default "batch".
assayName	A single character. The name for the corrected assay. Will be saved to assay . Default "scMerge".
seg	A vector of gene names or indices that specifies SEG (Stably Expressed Genes) set as negative control. Pre-defined dataset with human and mouse SEG lists is available to user by running <code>data('SEG')</code> . Default NULL, and this value will be auto-detected by default with scSEGIindex .
kmeansK	An integer vector. Indicating the kmeans' K-value for each batch (i.e. how many subclusters in each batch should exist), in order to construct pseudo-replicates. The length of <code>kmeansK</code> needs to be the same as the number of batches. Default NULL, and this value will be auto-detected by default, depending on <code>cellType</code> .
cellType	A single character. A string indicating a field in <code>colData(inSCE)</code> that defines different cell types. Default 'cell_type'.
nCores	An integer. The number of cores of processors to allocate for the task. Default 1L.

Value

The input [SingleCellExperiment](#) object with `assay(inSCE, assayName)` updated.

References

Hoa, et al., 2020

Examples

```
data('sceBatches', package = 'singleCellTK')
## Not run:
sceCorr <- runSCMerge(sceBatches)

## End(Not run)
```

runScranSNN

Get clustering with SNN graph

Description

Perform SNN graph clustering on a [SingleCellExperiment](#) object, with graph construction by [buildSNNGraph](#) and graph clustering by "igraph" package.

Usage

```
runScranSNN(
  inSCE,
  useAssay = NULL,
  useReducedDim = NULL,
  useAltExp = NULL,
  altExpAssay = "counts",
  clusterName = "scranSNN_cluster",
  k = 10,
  nComp = 50,
  weightType = c("rank", "number", "jaccard"),
  algorithm = c("walktrap", "louvain", "infomap", "fastGreedy", "labelProp",
    "leadingEigen")
)
```

Arguments

inSCE	A SingleCellExperiment object.
useAssay	A single character, specifying which assay to perform the clustering algorithm on. Default NULL.
useReducedDim	A single character, specifying which low-dimension representation (reducedDim) to perform the clustering algorithm on. Default NULL.
useAltExp	A single character, specifying the assay which altExp to perform the clustering algorithm on. Default NULL.
altExpAssay	A single character, specifying which assay in the chosen altExp to work on. Only used when useAltExp is set. Default "counts".
clusterName	A single character, specifying the name to store the cluster label in colData . Default "scranSNN_cluster".
k	An integer, the number of nearest neighbors used to construct the graph. Smaller value indicates higher resolution and larger number of clusters. Default 10.

nComp	An integer, the number of components to use when useAssay or useAltExp is specified. WON'T work with useReducedDim. Default 50.
weightType	A single character, that specifies the edge weighing scheme when constructing the Shared Nearest-Neighbor (SNN) graph. Choose from "rank", "number", "jaccard". Default "rank".
algorithm	A single character, that specifies the community detection algorithm to work on the SNN graph. Choose from "walktrap", "louvain", "infomap", "fastGreedy", "labelProp", "leadingEigen". Default "walktrap".

Value

The input `SingleCellExperiment` object with factor cluster labeling updated in `colData(inSCE)[[clusterName]]`.

References

Aaron Lun and et. al., 2016

Examples

```
data("mouseBrainSubsetSCE")
mouseBrainSubsetSCE <- runScranSNN(mouseBrainSubsetSCE,
                                   useReducedDim = "PCA_logcounts")
```

runScrublet	<i>Find doublets using scrublet.</i>
-------------	--------------------------------------

Description

A wrapper function that calls `scrub_doublets` from python module `scrublet`. Simulates doublets from the observed data and uses a k-nearest-neighbor classifier to calculate a continuous `scrublet_score` (between 0 and 1) for each transcriptome. The score is automatically thresholded to generate `scrublet_call`, a boolean array that is TRUE for predicted doublets and FALSE otherwise.

Usage

```
runScrublet(
  inSCE,
  sample = NULL,
  useAssay = "counts",
  simDoubletRatio = 2,
  nNeighbors = NULL,
  minDist = NULL,
  expectedDoubletRate = 0.1,
  stdevDoubletRate = 0.02,
  syntheticDoubletUmiSubsampling = 1,
  useApproxNeighbors = TRUE,
  distanceMetric = "euclidean",
  getDoubletNeighborParents = FALSE,
  minCounts = 3,
  minCells = 3L,
```

```

    minGeneVariabilityPctl = 85,
    logTransform = FALSE,
    meanCenter = TRUE,
    normalizeVariance = TRUE,
    nPrinComps = 30L,
    tsneAngle = NULL,
    tsnePerplexity = NULL,
    verbose = TRUE,
    seed = 12345
)

```

Arguments

<code>inSCE</code>	A SingleCellExperiment object. Needs counts in assays slot.
<code>sample</code>	Character vector. Indicates which sample each cell belongs to. Scrublet will be run on cells from each sample separately. If NULL, then all cells will be processed together. Default NULL.
<code>useAssay</code>	A string specifying which assay in the SCE to use. Default 'counts'.
<code>simDoubletRatio</code>	Numeric. Number of doublets to simulate relative to the number of observed transcriptomes. Default 2.0.
<code>nNeighbors</code>	Integer. Number of neighbors used to construct the KNN graph of observed transcriptomes and simulated doublets. If NULL, this is set to $\text{round}(0.5 * \sqrt{n_cells})$. Default NULL.
<code>minDist</code>	Float. Determines how tightly UMAP packs points together. If NULL, this is set to 0.1. Default NULL.
<code>expectedDoubletRate</code>	The estimated doublet rate for the experiment. Default 0.1.
<code>stdevDoubletRate</code>	Uncertainty in the expected doublet rate. Default 0.02.
<code>syntheticDoubletUmiSubsampling</code>	Numeric. Rate for sampling UMIs when creating synthetic doublets. If 1.0, each doublet is created by simply adding the UMIs from two randomly sampled observed transcriptomes. For values less than 1, the UMI counts are added and then randomly sampled at the specified rate. Default: 1.0.
<code>useApproxNeighbors</code>	Boolean. Use approximate nearest neighbor method (annoy) for the KNN classifier. Default TRUE.
<code>distanceMetric</code>	Character. Distance metric used when finding nearest neighbors. For list of valid values, see the documentation for <code>annoy</code> (if <code>useApproxNeighbors</code> is TRUE) or <code>sklearn.neighbors.NearestNeighbors</code> (if <code>useApproxNeighbors</code> is FALSE). Default "euclidean".
<code>getDoubletNeighborParents</code>	Boolean. If TRUE, return the parent transcriptomes that generated the doublet neighbors of each observed transcriptome. This information can be used to infer the cell states that generated a given doublet state. Default FALSE.
<code>minCounts</code>	Numeric. Used for gene filtering prior to PCA. Genes expressed at fewer than <code>minCounts</code> in fewer than <code>minCells</code> (see below) are excluded. Default 3.
<code>minCells</code>	Integer. Used for gene filtering prior to PCA. Genes expressed at fewer than <code>minCounts</code> (see above) in fewer than <code>minCells</code> are excluded. Default 3.

minGeneVariabilityPctl	Numeric. Used for gene filtering prior to PCA. Keep the most highly variable genes (in the top minGeneVariabilityPctl percentile), as measured by the v-statistic (<i>Klein et al., Cell 2015</i>). Default 85.
logTransform	Boolean. If TRUE, log-transform the counts matrix ($\log_{10}(1+TPM)$). <code>sklearn.decomposition.TruncatedSVD</code> will be used for dimensionality reduction, unless meanCenter is TRUE. Default FALSE.
meanCenter	If TRUE, center the data such that each gene has a mean of 0. <code>sklearn.decomposition.PCA</code> will be used for dimensionality reduction. Default TRUE.
normalizeVariance	Boolean. If TRUE, normalize the data such that each gene has a variance of 1. <code>sklearn.decomposition.TruncatedSVD</code> will be used for dimensionality reduction, unless meanCenter is TRUE. Default TRUE.
nPrinComps	Integer. Number of principal components used to embed the transcriptomes prior to k-nearest-neighbor graph construction. Default 30.
tsneAngle	Float. Determines angular size of a distant node as measured from a point in the t-SNE plot. If default, it is set to 0.5 Default NULL.
tsnePerplexity	Integer. The number of nearest neighbors that is used in other manifold learning algorithms. If default, it is set to 30. Default NULL.
verbose	Boolean. If TRUE, print progress updates. Default TRUE.
seed	Seed for the random number generator. Default 12345.

Value

A [SingleCellExperiment](#) object with `scrub_doublets` output appended to the `colData` slot. The columns include `scrublet_score` and `scrublet_call`.

Examples

```
data(scExample, package = "singleCellTK")
## Not run:
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
sce <- runScrublet(sce)

## End(Not run)
```

runZINBWAVE

Apply ZINBWAVE Batch effect correction method to SingleCellExperiment object

Description

A general and flexible zero-inflated negative binomial model that can be used to provide a low-dimensional representations of scRNAseq data. The model accounts for zero inflation (dropouts), over-dispersion, and the count nature of the data. The model also accounts for the difference in library sizes and optionally for batch effects and/or other covariates.

Usage

```
runZINBWaVE(
  inSCE,
  useAssay = "counts",
  batch = "batch",
  nHVG = 1000L,
  nComponents = 50L,
  epsilon = 1000,
  nIter = 10L,
  reducedDimName = "zinbwave"
)
```

Arguments

inSCE	SingleCellExperiment inherited object. Required.
useAssay	A single character indicating the name of the assay requiring batch correction. Note that ZINBWaVE works for counts (integer) input rather than logcounts that other methods prefer. Default "counts".
batch	A single character indicating a field in colData that annotates the batches. Default "batch".
nHVG	An integer. Number of highly variable genes to use when fitting the model. Default 1000L.
nComponents	An integer. The number of principle components or dimensionality to generate in the resulting matrix. Default 50L.
epsilon	An integer. Algorithmic parameter. Empirically, a high epsilon is often required to obtain a good low-level representation. Default 1000L.
nIter	An integer, The max number of iterations to perform. Default 10L.
reducedDimName	A single character. The name for the corrected low-dimensional representation. Will be saved to <code>reducedDim(inSCE)</code> . Default "zinbwave".

Value

The input [SingleCellExperiment](#) object with `reducedDim(inSCE, reducedDimName)` updated.

References

Pollen, Alex A et al., 2014

Examples

```
data('sceBatches', package = 'singleCellTK')
## Not run:
  sceCorr <- runZINBWaVE(sceBatches, nIter = 5)

## End(Not run)
```

sampleSummaryStats *Generate table of SCTK QC outputs.*

Description

Creates a table of QC metrics generated from QC algorithms via either kable or csv file.

Usage

```
sampleSummaryStats(inSCE, sample = NULL, useAssay = "counts", simple = TRUE)
```

Arguments

inSCE	Input SingleCellExperiment object with saved assay data and/or colData data. Required.
sample	Character vector. Indicates which sample each cell belongs to.
useAssay	A string specifying which assay in the SCE to use. Default 'counts'.
simple	Boolean. Indicates whether to generate a table of only basic QC stats (ex. library size), or to generate a summary table of all QC stats stored in the inSCE.

Value

A matrix/array object.

Examples

```
data(scExample, package = "singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
sampleSummaryStats(sce, simple = TRUE)
```

scater_logNormCounts *scater_logNormCounts Uses [logNormCounts](#) to log normalize input data*

Description

scater_logNormCounts Uses [logNormCounts](#) to log normalize input data

Usage

```
scater_logNormCounts(
  inSCE,
  logAssayName = "ScaterLogNormCounts",
  useAssay = "counts"
)
```

Arguments

inSCE	Input SingleCellExperiment object
logAssayName	New assay name for log normalized data
useAssay	Input assay

Value

inSCE Updated SingleCellExperiment object that contains the new log normalized data

Author(s)

Irzam Sarfraz

Examples

```
data(sce_chcl, package = "scds")
sce_chcl <- scater_logNormCounts(sce_chcl, "logcounts", "counts")
```

sce

Example Single Cell RNA-Seq data in SingleCellExperiment Object, subset of 10x public dataset <https://support.10xgenomics.com/single-cell-gene-expression/datasets/2.1.0/pbmc4k> A subset of 390 barcodes and top 200 genes were included in this example. Within 390 barcodes, 195 barcodes are empty droplet, 150 barcodes are cell barcode and 45 barcodes are doublets predicted by scrublet and doubletFinder package. This example only serves as a proof of concept and a tutorial on how to run the functions in this package. The results should not be used for drawing scientific conclusions.

Description

Example Single Cell RNA-Seq data in SingleCellExperiment Object, subset of 10x public dataset <https://support.10xgenomics.com/single-cell-gene-expression/datasets/2.1.0/pbmc4k> A subset of 390 barcodes and top 200 genes were included in this example. Within 390 barcodes, 195 barcodes are empty droplet, 150 barcodes are cell barcode and 45 barcodes are doublets predicted by scrublet and doubletFinder package. This example only serves as a proof of concept and a tutorial on how to run the functions in this package. The results should not be used for drawing scientific conclusions.

Usage

```
sce
```

Format

A [SingleCellExperiment](#) object.

Examples

```
data("scExample")
```

sceBatches	<i>Example Single Cell RNA-Seq data in SingleCellExperiment object, with different batches annotated</i>
------------	--

Description

Two batches of pancreas scRNAseq dataset are combined with their original counts. Cell types and batches are annotated in 'colData(sceBatches)'. Two batches came from Wang, et al., 2016, annotated as 'w'; and Xin, et al., 2016, annotated as 'x'. Two common cell types, 'alpha' and 'beta', that could be found in both original studies with relatively large population were kept for cleaner demonstration. `data('sceBatches')`

Usage

```
sceBatches
```

Format

An object of class `SingleCellExperiment` with 100 rows and 250 columns.

scran_modelGeneVar	<i>scran_modelGeneVar Generates and stores variability data from scran::modelGeneVar in the input singleCellExperiment object</i>
--------------------	---

Description

scran_modelGeneVar Generates and stores variability data from scran::modelGeneVar in the input singleCellExperiment object

Usage

```
scran_modelGeneVar(inSCE, assayName)
```

Arguments

inSCE	a singleCellExperiment object
assayName	selected assay to compute variable features from

Value

inSCE updated singleCellExperiment object that contains variable feature metrics in rowData

Author(s)

Irzam Sarfraz

Examples

```
data(sce_chc1, package = "scds")
sce_chc1 <- scran_modelGeneVar(sce_chc1, "counts")
```

`sctkListGeneSetCollections`*Lists imported GeneSetCollections*

Description

Returns a vector of GeneSetCollections that have been imported and stored in `metadata(inSCE)$sctk$genesets`.

Usage

```
sctkListGeneSetCollections(inSCE)
```

Arguments

`inSCE` A [SingleCellExperiment](#) object.

Value

Character vector.

Author(s)

Joshua D. Campbell

See Also

[importGeneSetsFromList](#) for importing from lists, [importGeneSetsFromGMT](#) for importing from GMT files, [GeneSetCollection](#) objects, and [importGeneSetsFromMSigDB](#) for importing MSigDB gene sets.

Examples

```
data(scExample)
library(GSEABase)
gs1 <- GeneSet(setName = "geneset1", geneIds = rownames(sce)[1:10])
gs2 <- GeneSet(setName = "geneset2", geneIds = rownames(sce)[11:20])
gsc1 <- GeneSetCollection(gs1)
gsc2 <- GeneSetCollection(gs2)
sce <- importGeneSetsFromCollection(inSCE = sce,
                                  geneSetCollection = gsc1,
                                  by = "rownames",
                                  collectionName = "Collection1")
sce <- importGeneSetsFromCollection(inSCE = sce,
                                  geneSetCollection = gsc2,
                                  by = "rownames",
                                  collectionName = "Collection2")
collections <- sctkListGeneSetCollections(sce)
```

sctkPythonInstallConda

Installs Python packages into a Conda environment

Description

Install all Python packages used in the [singleCellTK](#) package using [conda_install](#) from package [reticulate](#). This will create a new Conda environment with the name `envname` if not already present. Note that Anaconda or Miniconda already need to be installed on the local system.

Usage

```
sctkPythonInstallConda(
  envname = "sctk-reticulate",
  conda = "auto",
  packages = c("scipy", "numpy", "astroid", "six"),
  pipPackages = c("scrublet", "scanpy", "bbknn", "scanorama", "anndata"),
  selectConda = TRUE,
  forge = FALSE,
  pipIgnoreInstalled = TRUE,
  pythonVersion = NULL,
  ...
)
```

Arguments

<code>envname</code>	Character. Name of the conda environment to create.
<code>conda</code>	Character. Path to conda executable. Use "auto" to find conda using the PATH and other conventional install locations. Default 'auto'.
<code>packages</code>	Character Vector. List of packages to install from Conda.
<code>pipPackages</code>	Character Vector. List of packages to install into the Conda environment using 'pip'.
<code>selectConda</code>	Boolean. Run selectSCTKConda after installing all packages to select the Conda environment. Default TRUE.
<code>forge</code>	Boolean. Include the Conda Forge repository.
<code>pipIgnoreInstalled</code>	Boolean. Ignore installed versions when using pip. This is TRUE by default so that specific package versions can be installed even if they are downgrades. The FALSE option is useful for situations where you don't want a pip install to attempt an overwrite of a conda binary package (e.g. SciPy on Windows which is very difficult to install via pip due to compilation requirements).
<code>pythonVersion</code>	Passed to <code>python_version</code> variable in conda_install . Default NULL.
<code>...</code>	Other parameters to pass to conda_install .

Value

None. Installation of Conda environment.

See Also

See [conda_create](#) for more information on creating a Conda environment. See [conda_install](#) for more description of the installation parameters. See <https://rstudio.github.io/reticulate/> for more information on package [reticulate](#). See [selectSCTKConda](#) for reloading the Conda environment if R is restarted without going through the whole installation process again. See <https://docs.conda.io/en/latest/> for more information on Conda environments.

Examples

```
## Not run:
sctkPythonInstallConda(envname = "sctk-reticulate")

## End(Not run)
```

```
sctkPythonInstallVirtualEnv
```

Installs Python packages into a virtual environment

Description

Install all Python packages used in the [singleCellTK](#) package using [virtualenv_install](#) from package [reticulate](#). This will create a new virtual environment with the name `envname` if not already present.

Usage

```
sctkPythonInstallVirtualEnv(
  envname = "sctk-reticulate",
  packages = c("scipy", "numpy", "astroid", "six", "scrublet", "scanpy", "scanorama",
              "bbknn", "anndata"),
  selectEnvironment = TRUE,
  python = NULL
)
```

Arguments

<code>envname</code>	Character. Name of the virtual environment to create.
<code>packages</code>	Character Vector. List of packages to install.
<code>selectEnvironment</code>	Boolean. Run selectSCTKVirtualEnvironment after installing all packages to select the virtual environment. Default TRUE.
<code>python</code>	The path to a Python interpreter, to be used with the created virtual environment. When NULL, the Python interpreter associated with the current session will be used. Default NULL.

Value

None. Installation of virtual environment.

See Also

See [virtualenv_create](#) for more information on creating a Conda environment. See [virtualenv_install](#) for more description of the installation parameters. See <https://rstudio.github.io/reticulate/> for more information on package [reticulate](#). See [selectSCTKVirtualEnvironment](#) for reloading the virtual environment if R is restarted without going through the whole installation process again.

Examples

```
## Not run:
sctkPythonInstallVirtualEnv(envname = "sctk-reticulate")

## End(Not run)
```

SEG	<i>Stably Expressed Gene (SEG) list object, with SEG sets for human and mouse.</i>
-----	--

Description

The two gene sets came from dataset called 'segList' of package 'scMerge'.

Usage

```
SEG
```

Format

list, with two entries "human" and "mouse", each is a character vector.

Source

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

Examples

```
data('SEG')
humanSEG <- SEG$human
```

selectSCTKConda	<i>Selects a Conda environment</i>
-----------------	------------------------------------

Description

Selects a Conda environment with Python packages used in [singleCellTK](#).

Usage

```
selectSCTKConda(envname = "sctk-reticulate")
```


Arguments

envname Character. Name of the conda environment to activate.

Value

None. Selects Conda environment.

See Also

[conda-tools](#) for more information on using Conda environments with package [reticulate](#). See <https://rstudio.github.io/reticulate/> for more information on package [reticulate](#).

See [sctkPythonInstallConda](#) for installation of Python modules into a Conda environment. See [conda-tools](#) for more information on using Conda environments with package [reticulate](#). See <https://rstudio.github.io/reticulate/> for more information on package [reticulate](#). See <https://docs.conda.io/en/latest/> for more information on Conda environments.

Examples

```
## Not run:
sctkPythonInstallConda(envname = "sctk-reticulate", selectConda = FALSE)
selectSCTKConda(envname = "sctk-reticulate")

## End(Not run)
```

```
selectSCTKVirtualEnvironment
                              Selects a virtual environment
```

Description

Selects a virtual environment with Python packages used in [singleCellTK](#)

Usage

```
selectSCTKVirtualEnvironment(envname = "sctk-reticulate")
```

Arguments

envname Character. Name of the virtual environment to activate.

Value

None. Selects virtual environment.

See Also

See [sctkPythonInstallVirtualEnv](#) for installation of Python modules into a virtual environment. See [virtualenv-tools](#) for more information on using virtual environments with package [reticulate](#). See <https://rstudio.github.io/reticulate/> for more information on package [reticulate](#).

Examples

```
## Not run:
sctkPythonInstallVirtualEnv(envname = "sctk-reticulate", selectEnvironment = FALSE)
selectSCTKVirtualEnvironment(envname = "sctk-reticulate")

## End(Not run)
```

```
seuratComputeHeatmap seuratComputeHeatmap Computes the heatmap plot object from the
pca slot in the input sce object
```

Description

seuratComputeHeatmap Computes the heatmap plot object from the pca slot in the input sce object

Usage

```
seuratComputeHeatmap(
  inSCE,
  useAssay,
  useReduction = c("pca", "ica"),
  dims = NULL,
  nfeatures = 30,
  fast = TRUE,
  combine = TRUE,
  raster = TRUE
)
```

Arguments

inSCE	(sce) object from which to compute heatmap (pca should be computed)
useAssay	Assay containing scaled counts to use in heatmap.
useReduction	Reduction method to use for computing clusters. One of "pca" or "ica". Default "pca".
dims	Number of components to generate heatmap plot objects. If NULL, a heatmap will be generated for all components. Default NULL.
nfeatures	Numer of features to include in the heatmap. Default 30.
fast	See DimHeatmap for more information. Default TRUE.
combine	See DimHeatmap for more information. Default TRUE.
raster	See DimHeatmap for more information. Default TRUE.

Value

plot object

Examples

```
data(scExample, package = "singleCellTK")
## Not run:
sce <- seuratNormalizeData(sce, useAssay = "counts")
sce <- seuratFindHVG(sce, useAssay = "counts")
sce <- seuratScaleData(sce, useAssay = "counts")
sce <- seuratPCA(sce, useAssay = "counts")
heatmap <- seuratComputeHeatmap(sce, useAssay = "counts")
seuratHeatmapPlot(heatmap)

## End(Not run)
```

seuratComputeJackStraw

seuratComputeJackStraw Compute jackstraw plot and store the computations in the input sce object

Description

seuratComputeJackStraw Compute jackstraw plot and store the computations in the input sce object

Usage

```
seuratComputeJackStraw(inSCE, useAssay, dims = NULL)
```

Arguments

inSCE	(sce) object on which to compute and store jackstraw plot
useAssay	Assay containing scaled counts to use in JackStraw calculation.
dims	Number of components to test in Jackstraw. If NULL, then all components are used. Default NULL.

Value

Updated SingleCellExperiment object with jackstraw computations stored in it

Examples

```
data(scExample, package = "singleCellTK")
## Not run:
sce <- seuratNormalizeData(sce, useAssay = "counts")
sce <- seuratFindHVG(sce, useAssay = "counts")
sce <- seuratScaleData(sce, useAssay = "counts")
sce <- seuratPCA(sce, useAssay = "counts")
sce <- seuratComputeJackStraw(sce, useAssay = "counts")

## End(Not run)
```

seuratElbowPlot	<i>seuratElbowPlot Computes the plot object for elbow plot from the pca slot in the input sce object</i>
-----------------	--

Description

seuratElbowPlot Computes the plot object for elbow plot from the pca slot in the input sce object

Usage

```
seuratElbowPlot(inSCE, significantPC = NULL, reduction = "pca")
```

Arguments

inSCE	(sce) object from which to compute the elbow plot (pca should be computed)
significantPC	Number of significant principal components to plot. This is used to alter the color of the points for the corresponding PCs. If NULL, all points will be the same color. Default NULL.
reduction	Reduction to use for elbow plot generation. Either "pca" or "ica". Default "pca".

Value

plot object

Examples

```
data(scExample, package = "singleCellTK")
## Not run:
sce <- seuratNormalizeData(sce, useAssay = "counts")
sce <- seuratFindHVG(sce, useAssay = "counts")
sce <- seuratScaleData(sce, useAssay = "counts")
sce <- seuratPCA(sce, useAssay = "counts")
seuratElbowPlot(sce)

## End(Not run)
```

seuratFindClusters	<i>seuratFindClusters Computes the clusters from the input sce object and stores them back in sce object</i>
--------------------	--

Description

seuratFindClusters Computes the clusters from the input sce object and stores them back in sce object

Usage

```
seuratFindClusters(
  inSCE,
  useAssay,
  useReduction = c("pca", "ica"),
  dims = 10,
  algorithm = c("louvain", "multilevel", "SLM"),
  groupSingletons = TRUE,
  resolution = 0.8,
  externalReduction = NULL
)
```

Arguments

<code>inSCE</code>	(sce) object from which clusters should be computed and stored in
<code>useAssay</code>	Assay containing scaled counts to use for clustering.
<code>useReduction</code>	Reduction method to use for computing clusters. One of "pca" or "ica". Default "pca".
<code>dims</code>	numeric value of how many components to use for computing clusters. Default 10.
<code>algorithm</code>	selected algorithm to compute clusters. One of "louvain", "multilevel", or "SLM". Use <code>louvain</code> for "original Louvain algorithm" and <code>multilevel</code> for "Louvain algorithm with multilevel refinement". Default <code>louvain</code> .
<code>groupSingletons</code>	boolean if singletons should be grouped together or not. Default TRUE.
<code>resolution</code>	Set the resolution parameter to find larger (value above 1) or smaller (value below 1) number of communities. Default 0.8.
<code>externalReduction</code>	Pass <code>DimReduc</code> object if PCA/ICA computed through other libraries. Default NULL.

Value

Updated sce object which now contains the computed clusters

Examples

```
data(scExample, package = "singleCellTK")
## Not run:
sce <- seuratNormalizeData(sce, useAssay = "counts")
sce <- seuratFindHVG(sce, useAssay = "counts")
sce <- seuratScaleData(sce, useAssay = "counts")
sce <- seuratPCA(sce, useAssay = "counts")
sce <- seuratFindClusters(sce, useAssay = "counts")

## End(Not run)
```

seuratFindHVG	<i>seuratFindHVG Find highly variable genes and store in the input sce object</i>
---------------	---

Description

seuratFindHVG Find highly variable genes and store in the input sce object

Usage

```
seuratFindHVG(inSCE, useAssay, hvgMethod = "vst", hvgNumber = 2000)
```

Arguments

inSCE	(sce) object to compute highly variable genes from and to store back to it
useAssay	Normalized assay inside the SCE object to use for hvg computation.
hvgMethod	selected method to use for computation of highly variable genes. One of 'vst', 'dispersion', or 'mean.var.plot'. Default "vst".
hvgNumber	numeric value of how many genes to select as highly variable. Default 2000.

Value

Updated SingleCellExperiment object with highly variable genes computation stored

Examples

```
data(scExample, package = "singleCellTK")
## Not run:
sce <- seuratNormalizeData(sce, useAssay = "counts")
sce <- seuratFindHVG(sce, useAssay = "counts")

## End(Not run)
```

seuratHeatmapPlot	<i>seuratHeatmapPlot Modifies the heatmap plot object so it contains specified number of heatmaps in a single plot</i>
-------------------	--

Description

seuratHeatmapPlot Modifies the heatmap plot object so it contains specified number of heatmaps in a single plot

Usage

```
seuratHeatmapPlot(plotObject, dims, ncol, labels)
```

Arguments

plotObject	plot object computed from seuratComputeHeatmap() function
dims	numerical value of how many heatmaps to draw (default is 0)
ncol	numerical value indicating that in how many columns should the heatmaps be distributed (default is 2)
labels	list() of labels to draw on heatmaps

Value

modified plot object

seuratICA	<i>seuratICA Computes ICA on the input sce object and stores the calculated independent components within the sce object</i>
-----------	--

Description

seuratICA Computes ICA on the input sce object and stores the calculated independent components within the sce object

Usage

```
seuratICA(inSCE, useAssay, reducedDimName = "seuratICA", nics = 20)
```

Arguments

inSCE	(sce) object on which to compute ICA
useAssay	Assay containing scaled counts to use in ICA.
reducedDimName	Name of new reducedDims object containing Seurat ICA Default seuratICA.
nics	Number of independent components to compute. Default 20.

Value

Updated SingleCellExperiment object which now contains the computed independent components

Examples

```
data(scExample, package = "singleCellTK")
## Not run:
sce <- seuratNormalizeData(sce, useAssay = "counts")
sce <- seuratFindHVG(sce, useAssay = "counts")
sce <- seuratScaleData(sce, useAssay = "counts")
sce <- seuratICA(sce, useAssay = "counts")

## End(Not run)
```

seuratIntegration	<i>seuratIntegration</i> A wrapper function to Seurat Batch-Correction/Integration workflow.
-------------------	--

Description

seuratIntegration A wrapper function to Seurat Batch-Correction/Integration workflow.

Usage

```
seuratIntegration(
  inSCE,
  useAssay = "counts",
  batch,
  newAssayName = "SeuratIntegratedAssay",
  kAnchor,
  kFilter,
  kWeight,
  ndims = 10
)
```

Arguments

inSCE	Input SingleCellExperiment object that contains the assay to batch-correct.
useAssay	Assay to batch-correct.
batch	Batch variable from colData slot of SingleCellExperiment object.
newAssayName	Assay name for the batch-corrected output assay.
kAnchor	Number of neighbours to use for finding the anchors in the FindIntegrationAnchors function.
kFilter	Number of neighbours to use for filtering the anchors in the FindIntegrationAnchors function.
kWeight	Number of neighbours to use when weighing the anchors in the IntegrateData function.
ndims	Number of dimensions to use. Default 10.

Value

A SingleCellExperiment object that contains the batch-corrected assay inside the altExp slot of the object

seuratJackStrawPlot	<i>seuratJackStrawPlot Computes the plot object for jackstraw plot from the pca slot in the input sce object</i>
---------------------	--

Description

seuratJackStrawPlot Computes the plot object for jackstraw plot from the pca slot in the input sce object

Usage

```
seuratJackStrawPlot(inSCE, dims = NULL)
```

Arguments

inSCE	(sce) object from which to compute the jackstraw plot (pca should be computed)
dims	Number of components to plot in Jackstraw. If NULL, then all components are plotted Default NULL.

Value

plot object

Examples

```
data(scExample, package = "singleCellTK")
## Not run:
sce <- seuratNormalizeData(sce, useAssay = "counts")
sce <- seuratFindHVG(sce, useAssay = "counts")
sce <- seuratScaleData(sce, useAssay = "counts")
sce <- seuratPCA(sce, useAssay = "counts")
sce <- seuratComputeJackStraw(sce, useAssay = "counts")
seuratJackStrawPlot(sce)

## End(Not run)
```

seuratNormalizeData	<i>seuratNormalizeData Wrapper for NormalizeData() function from seurat library Normalizes the sce object according to the input parameters</i>
---------------------	---

Description

seuratNormalizeData Wrapper for NormalizeData() function from seurat library Normalizes the sce object according to the input parameters

Usage

```
seuratNormalizeData(
  inSCE,
  useAssay,
  normAssayName = "seuratNormData",
  normalizationMethod = "LogNormalize",
  scaleFactor = 10000
)
```

Arguments

`inSCE` (sce) object to normalize

`useAssay` Assay containing raw counts to use for normalization.

`normAssayName` Name of new assay containing normalized data. Default `seuratNormData`.

`normalizationMethod` selected normalization method. Default `"LogNormalize"`.

`scaleFactor` numeric value that represents the scaling factor. Default `10000`.

Value

Normalized `SingleCellExperiment` object

Examples

```
data(scExample, package = "singleCellTK")
## Not run:
sce <- seuratNormalizeData(sce, useAssay = "counts")

## End(Not run)
```

seuratPCA	<i>seuratPCA Computes PCA on the input sce object and stores the calculated principal components within the sce object</i>
-----------	--

Description

`seuratPCA` Computes PCA on the input `sce` object and stores the calculated principal components within the `sce` object

Usage

```
seuratPCA(inSCE, useAssay, reducedDimName = "seuratPCA", nPCs = 20)
```

Arguments

`inSCE` (sce) object on which to compute PCA

`useAssay` Assay containing scaled counts to use in PCA.

`reducedDimName` Name of new `reducedDims` object containing Seurat PCA. Default `seuratPCA`.

`nPCs` numeric value of how many components to compute. Default `20`.

Value

Updated SingleCellExperiment object which now contains the computed principal components

Examples

```
data(scExample, package = "singleCellTK")
## Not run:
sce <- seuratNormalizeData(sce, useAssay = "counts")
sce <- seuratFindHVG(sce, useAssay = "counts")
sce <- seuratScaleData(sce, useAssay = "counts")
sce <- seuratPCA(sce, useAssay = "counts")

## End(Not run)
```

seuratPlotHVG	<i>seuratPlotHVG Plot highly variable genes from input sce object (must have highly variable genes computations stored)</i>
---------------	---

Description

seuratPlotHVG Plot highly variable genes from input sce object (must have highly variable genes computations stored)

Usage

```
seuratPlotHVG(inSCE)
```

Arguments

inSCE (sce) object that contains the highly variable genes computations

Value

plot object

Examples

```
data(scExample, package = "singleCellTK")
## Not run:
sce <- seuratNormalizeData(sce, useAssay = "counts")
sce <- seuratFindHVG(sce, useAssay = "counts")
seuratPlotHVG(sce)

## End(Not run)
```

seuratReductionPlot	<i>seuratReductionPlot Plots the selected dimensionality reduction method</i>
---------------------	---

Description

seuratReductionPlot Plots the selected dimensionality reduction method

Usage

```
seuratReductionPlot(
  inSCE,
  useReduction = c("pca", "ica", "tsne", "umap"),
  showLegend = FALSE
)
```

Arguments

inSCE	(sce) object which has the selected dimensionality reduction algorithm already computed and stored
useReduction	Dimensionality reduction to plot. One of "pca", "ica", "tsne", or "umap". Default "umap".
showLegend	Select if legends should be shown on the output plot or not. Either "TRUE" or "FALSE". Default FALSE.

Value

plot object

Examples

```
data(scExample, package = "singleCellTK")
## Not run:
sce <- seuratNormalizeData(sce, useAssay = "counts")
sce <- seuratFindHVG(sce, useAssay = "counts")
sce <- seuratScaleData(sce, useAssay = "counts")
sce <- seuratPCA(sce, useAssay = "counts")
seuratReductionPlot(sce, useReductionPlot = "pca")
## End(Not run)
```

seuratRunTSNE	<i>seuratRunTSNE Computes tSNE from the given sce object and stores the tSNE computations back into the sce object</i>
---------------	--

Description

seuratRunTSNE Computes tSNE from the given sce object and stores the tSNE computations back into the sce object

Usage

```
seuratRunTSNE(
  inSCE,
  useReduction = c("pca", "ica"),
  reducedDimName = "seuratTSNE",
  dims = 10,
  perplexity = 30
)
```

Arguments

<code>inSCE</code>	(sce) object on which to compute the tSNE
<code>useReduction</code>	selected reduction algorithm to use for computing tSNE. One of "pca" or "ica". Default "pca".
<code>reducedDimName</code>	Name of new reducedDims object containing Seurat tSNE Default seuratTSNE.
<code>dims</code>	Number of reduction components to use for tSNE computation. Default 10.
<code>perplexity</code>	Adjust the perplexity tuneable parameter for the underlying tSNE call. Default 30.

Value

Updated sce object with tSNE computations stored

<code>seuratRunUMAP</code>	<i>seuratRunUMAP Computes UMAP from the given sce object and stores the UMAP computations back into the sce object</i>
----------------------------	--

Description

`seuratRunUMAP` Computes UMAP from the given sce object and stores the UMAP computations back into the sce object

Usage

```
seuratRunUMAP(
  inSCE,
  useReduction = c("pca", "ica"),
  reducedDimName = "seuratUMAP",
  dims = 10,
  minDist = 0.3,
  nNeighbors = 30L,
  spread = 1
)
```

Arguments

<code>inSCE</code>	(sce) object on which to compute the UMAP
<code>useReduction</code>	Reduction to use for computing UMAP. One of "pca" or "ica". Default is "pca".
<code>reducedDimName</code>	Name of new reducedDims object containing Seurat UMAP Default seuratUMAP.
<code>dims</code>	Numerical value of how many reduction components to use for UMAP computation. Default 10.
<code>minDist</code>	Sets the "min.dist" parameter to the underlying UMAP call. See RunUMAP for more information. Default 0.3.
<code>nNeighbors</code>	Sets the "n.neighbors" parameter to the underlying UMAP call. See RunUMAP for more information. Default 30L.
<code>spread</code>	Sets the "spread" parameter to the underlying UMAP call. See RunUMAP for more information. Default 1.

Value

Updated sce object with UMAP computations stored

Examples

```
data(scExample, package = "singleCellTK")
## Not run:
sce <- seuratNormalizeData(sce, useAssay = "counts")
sce <- seuratFindHVG(sce, useAssay = "counts")
sce <- seuratScaleData(sce, useAssay = "counts")
sce <- seuratPCA(sce, useAssay = "counts")
sce <- seuratFindClusters(sce, useAssay = "counts")
sce <- seuratRunUMAP(sce, useReduction = "pca")

## End(Not run)
```

<code>seuratScaleData</code>	<i>seuratScaleData Scales the input sce object according to the input parameters</i>
------------------------------	--

Description

`seuratScaleData` Scales the input sce object according to the input parameters

Usage

```
seuratScaleData(
  inSCE,
  useAssay,
  scaledAssayName = "seuratScaledData",
  model = "linear",
  scale = TRUE,
  center = TRUE,
  scaleMax = 10
)
```

Arguments

inSCE	(sce) object to scale
useAssay	Assay containing normalized counts to scale.
scaledAssayName	Name of new assay containing scaled data. Default seuratScaledData.
model	selected model to use for scaling data. Default "linear".
scale	boolean if data should be scaled or not. Default TRUE.
center	boolean if data should be centered or not. Default TRUE
scaleMax	maximum numeric value to return for scaled data. Default 10.

Value

Scaled SingleCellExperiment object

Examples

```
data(scExample, package = "singleCellTK")
## Not run:
sce <- seuratNormalizeData(sce, useAssay = "counts")
sce <- seuratFindHVG(sce, useAssay = "counts")
sce <- seuratScaleData(sce, useAssay = "counts")

## End(Not run)
```

seuratSCTransform *seuratSCTransform* Runs the *SCTransform* function to transform/normalize the input data

Description

seuratSCTransform Runs the *SCTransform* function to transform/normalize the input data

Usage

```
seuratSCTransform(inSCE, normAssayName = "SCTCounts", useAssay = "counts")
```

Arguments

inSCE	Input SingleCellExperiment object
normAssayName	Name for the output data assay. Default "SCTCounts".
useAssay	Name for the input data assay. Default "counts".

Value

Updated SingleCellExperiment object containing the transformed data

Examples

```
data(sce_chc1, package = "scds")
sce_chc1 <- seuratSCTransform(sce_chc1, "SCTCounts", "counts")
```

simpleLog	<i>A decorator that prints the arguments to the decorated function</i>
-----------	--

Description

A decorator that prints the arguments to the decorated function

Usage

```
simpleLog(f)
```

Arguments

f	A function to decorate
---	------------------------

Value

Prints message

singleCellTK	<i>Run the single cell analysis app</i>
--------------	---

Description

Use this function to run the single cell analysis app.

Usage

```
singleCellTK(inSCE = NULL, includeVersion = TRUE, theme = "yeti")
```

Arguments

inSCE	Input SingleCellExperiment object.
includeVersion	Include the version number in the SCTK header. The default is TRUE.
theme	The bootswatch theme to use for the singleCellTK UI. The default is 'flatly'.

Value

The shiny app will open

Examples

```
#Upload data through the app
if(interactive()){
  singleCellTK()
}

# Load the app with a SingleCellExperiment object
if(interactive()){
  data("mouseBrainSubsetSCE")
  singleCellTK(mouseBrainSubsetSCE)
}
```

subDiffEx	<i>Passes the output of generateSimulatedData() to differential expression tests, picking either t-tests or ANOVA for data with only two conditions or multiple conditions, respectively.</i>
-----------	---

Description

Passes the output of generateSimulatedData() to differential expression tests, picking either t-tests or ANOVA for data with only two conditions or multiple conditions, respectively.

Usage

```
subDiffEx(tempData)

subDiffExttest(countMatrix, class.labels, test.type = "t.equalvar")

subDiffExANOVA(countMatrix, condition)
```

Arguments

tempData	Matrix. The output of generateSimulatedData(), where the first row contains condition labels.
countMatrix	Matrix. A simulated counts matrix, sans labels.
class.labels	Factor. The condition labels for the simulated cells. Will be coerced into 1's and 0's.
test.type	Type of test to perform. The default is t.equalvar.
condition	Factor. The condition labels for the simulated cells.

Value

subDiffEx(): A vector of fdr-adjusted p-values for all genes. Nonviable results (such as for genes with 0 counts in a simulated dataset) are coerced to 1.

subDiffExttest(): A vector of fdr-adjusted p-values for all genes. Nonviable results (such as for genes with 0 counts in a simulated dataset) are coerced to 1.

subDiffExANOVA(): A vector of fdr-adjusted p-values for all genes. Nonviable results (such as for genes with 0 counts in a simulated dataset) are coerced to 1.

Functions

- subDiffEx:
- subDiffExttest: Runs t-tests on all genes in a simulated dataset with 2 conditions, and adjusts for FDR.
- subDiffExANOVA: Runs ANOVA on all genes in a simulated dataset with more than 2 conditions, and adjusts for FDR.

Examples

```

data("mouseBrainSubsetSCE")
res <- generateSimulatedData(
  totalReads = 1000, cells=10,
  originalData = assay(mouseBrainSubsetSCE, "counts"),
  realLabels = colData(mouseBrainSubsetSCE)[, "level1class"])
tempSigDiff <- subDiffEx(res)

data("mouseBrainSubsetSCE")
#sort first 100 expressed genes
ord <- rownames(mouseBrainSubsetSCE)[
  order(rowSums(assay(mouseBrainSubsetSCE, "counts")),
    decreasing = TRUE)][1:100]
#subset to those first 100 genes
subset <- mouseBrainSubsetSCE[ord, ]
res <- generateSimulatedData(totalReads = 1000, cells=10,
  originalData = assay(subset, "counts"),
  realLabels = colData(subset)[, "level1class"])

realLabels <- res[1, ]
output <- res[-1, ]
fdr <- subDiffExttest(output, realLabels)

data("mouseBrainSubsetSCE")
#sort first 100 expressed genes
ord <- rownames(mouseBrainSubsetSCE)[
  order(rowSums(assay(mouseBrainSubsetSCE, "counts")),
    decreasing = TRUE)][1:100]
# subset to those first 100 genes
subset <- mouseBrainSubsetSCE[ord, ]
res <- generateSimulatedData(totalReads = 1000, cells=10,
  originalData = assay(subset, "counts"),
  realLabels = colData(subset)[, "level2class"])

realLabels <- res[1, ]
output <- res[-1, ]
fdr <- subDiffExANOVA(output, realLabels)

```

subsetSCECols

*Subset a SingleCellExperiment object by columns***Description**

Used to perform subsetting of a [SingleCellExperiment](#) object using a variety of methods that indicate the correct columns to keep. The various methods, `index`, `bool`, and `colData`, can be used in conjunction with one another.

Usage

```
subsetSCECols(inSCE, index = NULL, bool = NULL, colData = NULL)
```

Arguments

`inSCE` Input [SingleCellExperiment](#) object.

index	Integer vector. Vector of indices indicating which columns to keep. If NULL, this will not be used for subsetting. Default NULL.
bool	Boolean vector. Vector of TRUE or FALSE indicating which columns should be kept. Needs to be the same length as the number of columns in inSCE. If NULL, this will not be used for subsetting. Default NULL.
colData	Character. An expression that will identify a subset of columns using variables found in the colData of inSCE. For example, if x is a numeric vector in colData, then "x < 5" will return all columns with x less than 5. Single quotes should be used for character strings. For example, "y == 'yes'" will return all columns where y is "yes". Multiple expressions can be evaluated by placing them in a vector. For example c("x < 5", "y == 'yes'") will apply both operations for subsetting. If NULL, this will not be used for subsetting. Default NULL.

Value

A [SingleCellExperiment](#) object that has been subsetted by colData.

Author(s)

Joshua D. Campbell

Examples

```
data(scExample)
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
```

subsetSCERows

Subset a SingleCellExperiment object by rows

Description

Used to perform subsetting of a [SingleCellExperiment](#) object using a variety of methods that indicate the correct rows to keep. The various methods, index, bool, and rowData, can be used in conjunction with one another. If returnAsAltExp is set to TRUE, then the returned object will have the same number of rows as the input inSCE as the subsetted object will be stored in the altExp slot.

Usage

```
subsetSCERows(
  inSCE,
  index = NULL,
  bool = NULL,
  rowData = NULL,
  returnAsAltExp = TRUE,
  altExpName = "subset",
  prependAltExpName = TRUE
)
```

Arguments

<code>inSCE</code>	Input SingleCellExperiment object.
<code>index</code>	Integer vector. Vector of indices indicating which rows to keep. If NULL, this will not be used for subsetting. Default NULL.
<code>bool</code>	Boolean vector. Vector of TRUE or FALSE indicating which rows should be kept. Needs to be the same length as the number of rows in <code>inSCE</code> . If NULL, this will not be used for subsetting. Default NULL.
<code>rowData</code>	Character. An expression that will identify a subset of rows using variables found in the <code>rowData</code> of <code>inSCE</code> . For example, if <code>x</code> is a numeric vector in <code>rowData</code> , then <code>"x < 5"</code> will return all rows with <code>x</code> less than 5. Single quotes should be used for character strings. For example, <code>"y == 'yes'"</code> will return all rows where <code>y</code> is "yes". Multiple expressions can be evaluated by placing them in a vector. For example <code>c("x < 5", "y == 'yes'")</code> will apply both operations for subsetting. If NULL, this will not be used for subsetting. Default NULL.
<code>returnAsAltExp</code>	Boolean. If TRUE, the subsetted SingleCellExperiment object will be returned in the <code>altExp</code> slot of <code>inSCE</code> . If FALSE, the subsetted SingleCellExperiment object will be directly returned.
<code>altExpName</code>	Character. Name of the alternative experiment object to add if <code>returnAsAltExp = TRUE</code> . Default <code>subset</code> .
<code>prependAltExpName</code>	Boolean. If TRUE, <code>altExpName</code> will be added to the beginning of the assay names in the <code>altExp</code> object. This is only utilized if <code>returnAsAltExp = TRUE</code> . Default TRUE.

Value

A [SingleCellExperiment](#) object that has been subsetted by `rowData`.

Author(s)

Joshua D. Campbell

Examples

```
data(scExample)

# Set a variable up in the rowData indicating mitochondrial genes
rowData(sce)$isMito <- ifelse(grepl("^MT-", rowData(sce)$feature_name),
                             "yes", "no")
sce <- subsetSCERows(sce, rowData = "isMito == 'yes'")
```

summarizeSCE

Summarize an assay in a [SingleCellExperiment](#)

Description

Creates a table of summary metrics from an input [SingleCellExperiment](#)

Usage

```
summarizeSCE(inSCE, useAssay = NULL, sampleVariableName = NULL)
```

Arguments

`inSCE` Input SingleCellExperiment object.

`useAssay` Indicate which assay to summarize. If NULL, then the first assay in `inSCE` will be used. Default NULL.

`sampleVariableName` Variable name in `colData` denoting which sample each cell belongs to. If NULL, all cells will be assumed to come from the same sample. Default "sample".

Value

A data.frame object of summary metrics.

Examples

```
data("mouseBrainSubsetSCE")
summarizeSCE(mouseBrainSubsetSCE, sample = NULL)
```

<code>trimCounts</code>	<i>Trim Counts</i>
-------------------------	--------------------

Description

Trims an input count matrix such that each value greater than a threshold value and each value less than a provided lower threshold value is trimmed to the lower threshold value.

Usage

```
trimCounts(counts, trimValue = c(10, -10))
```

Arguments

`counts` matrix

`trimValue` where `trimValue[1]` for upper threshold and `trimValue[2]` as lower threshold (default is `c(10,-10)`)

Value

trimmed counts matrix

Examples

```
data(sce_chc1, package = "scds")
assay(sce_chc1, "countsTrimmed") <- trimCounts(assay(sce_chc1, "counts"), c(10, -10))
```

visPlot

*visPlot***Description**

Given a plotting method with condition and gene list, return the respective visualization plot(s).

Usage

```
visPlot(
  inSCE,
  useAssay,
  method,
  condition = NULL,
  glist,
  facetWrap = TRUE,
  scaleHMap = TRUE,
  convertFactor = FALSE
)
```

Arguments

inSCE	Input SingleCellExperiment object.
useAssay	The assay to use in the visualization plot. Required
method	Visualization method. Available options are boxplot, scatterplot, or heatmap. Required
condition	colData annotation of the experiment. Required
glist	selected genes for visualization. Maximum 25 genes. Required
facetWrap	facet wrap according to genes for boxplot, scatterplot and barplot. Default is FALSE. Optional
scaleHMap	scale heatmap expression values. Default is TRUE. Optional
convertFactor	If the condition is not a factor, convert it to a factor before plotting. The default is FALSE

Value

A visualization plot

Examples

```
visPlot(mouseBrainSubsetSCE, "logcounts", "boxplot", "level1class", "C1qa")
visPlot(mouseBrainSubsetSCE, "counts", "scatterplot", "age", "Cmtm5")
visPlot(mouseBrainSubsetSCE, "counts", "heatmap", "level1class",
        c("Cmtm5", "C1qa"))
```

Index

* datasets

- mouseBrainSubsetSCE, 64
- msigdb_table, 64
- sce, 163
- sceBatches, 164
- SEG, 168
- .addSeuratToMetaDataSCE, 5
- .checkDiffExpResultExists, 6
- .computeSignificantPC, 7
- .extractSCEAnnotation, 7
- .formatDEAList, 8
- .GetComponentNames, 9
- .ggBar, 9
- .ggDensity, 10
- .ggScatter, 11
- .ggViolin, 13, 100
- .sce2adata, 14
- .seuratGetVariableFeatures, 15
- .seuratInvalidate, 15
- .updateAssaySCE, 16
- altExp, 157, 187
- assay, 16, 136, 148, 152, 155–157, 162
- barcodeRanks, 131
- bcds, 133
- BiocParallelParam, 142, 154
- buildSNNGraph, 157
- calcEffectSizes, 17
- colData, 38, 131–133, 135, 137, 138, 140, 142, 144, 146–148, 152, 154–157, 160–162
- colorRamp2, 111
- combineSCE, 17
- computeZScore, 18
- conda_create, 167
- conda_install, 166, 167
- constructSCE, 18
- convertGeneIDs, 19
- convertSCEToSeurat, 20
- convertSeuratToSCE, 20
- cxds, 136, 137
- cxds_bcds_hybrid, 137, 138

- data.frame, 130, 141, 150, 151
- data.table, 18
- dataAnnotationColor, 21
- DataFrame-class, 154
- dbscan, 139, 140
- decontX, 139, 153
- DelayedArray, 40, 44, 46–49, 51, 57, 59–61, 124
- DelayedArray-class, 42
- DimHeatmap, 170
- discreteColorPalette, 22
- distinctColors, 21, 22, 22
- doubletCells, 143
- downSampleCells, 23
- downSampleDepth, 24
- emptyDrops, 58, 131, 145, 146
- enrichRSCE, 25
- exportSCE, 26
- exportSCEtoAnnData, 27
- exportSCEtoFlatFile, 28
- featureIndex, 28, 52, 53, 55, 56
- featureNameDedup, 30
- FindIntegrationAnchors, 176
- findMarkerDiffExp, 30, 89, 90
- fit_dirichlet, 139
- generateMeta, 31
- generateSimulatedData, 32
- GeneSetCollection, 51–57, 165
- getBiomarker, 33
- getGmt, 53, 154
- getMSigDBTable, 33
- getPCA, 34
- getSceParams, 35
- getTopHVG, 35
- getTSNE, 36
- getUMAP, 37
- grep, 29, 128
- GSEABase, 52, 53, 55, 56
- gsvaPlot (gsvaSCE), 39
- gsvaSCE, 39
- Heatmap, 90, 111

- importAnnData, 40
- importBUStools, 41
- importCellRanger, 42
- importCellRangerV2 (importCellRanger), 42
- importCellRangerV2Sample, 46
- importCellRangerV3 (importCellRanger), 42
- importCellRangerV3Sample, 47
- importDropEst, 48
- importExampleData, 49
- importFromFiles, 50
- importGeneSetsFromCollection, 51, 54, 55
- importGeneSetsFromGMT, 52, 53, 55, 57, 165
- importGeneSetsFromList, 52, 54, 54, 57, 165
- importGeneSetsFromMSigDB, 33, 34, 52, 54, 55, 56, 64, 165
- importMultipleSources, 57
- importOptimus, 58
- importSEQC, 59
- importSTARsolo, 61
- IntegrateData, 176
- iterateSimulations, 62

- kmeans, 147

- list.dirs, 43
- logNormCounts, 38, 162

- Matrix, 49
- matrix, 42, 44, 46, 47, 51, 59–61
- mergeSCEColData, 63
- metadata, 52, 53, 55, 56
- modelGeneVar, 140
- mouseBrainSubsetSCE, 64
- msigdb_table, 64
- msigdbr, 56
- msigdbr_show_species, 56

- plotBarcodeRankDropsResults, 65
- plotBarcodeRankScatter, 66
- plotBatchVariance, 67
- plotBcDsResults, 68
- plotBiomarker, 71
- plotCxdsResults, 72
- plotDecontXResults, 74
- plotDEGHeatmap, 77
- plotDEGRegression, 79
- plotDEGViolin, 80
- plotDoubletCellsResults, 81
- plotDoubletFinderResults, 83
- plotEmptyDropsResults, 86

- plotEmptyDropsScatter, 87
- plotMarkerDiffExp, 89
- plotMASTThresholdGenes, 90
- plotPCA, 91
- plotRunPerCellQCResults, 92
- plotScdsHybridResults, 94
- plotSCEBarAssayData, 96
- plotSCEBarColData, 98
- plotSCEBatchFeatureMean, 99
- plotSCEDensity, 100
- plotSCEDensityAssayData, 102
- plotSCEDensityColData, 103
- plotSCEDimReduceColData, 104
- plotSCEDimReduceFeatures, 107
- plotSCEHeatmap, 78, 90, 109
- plotSCEScatter, 111
- plotSCEViolin, 114
- plotSCEViolinAssayData, 115
- plotSCEViolinColData, 117
- plotScrubletResults, 118
- plotTSNE, 121
- plotUMAP, 122

- qcInputProcess, 123

- rainbow, 21
- readMM, 42, 44, 46, 47, 51, 59–61
- readSingleCellMatrix, 124
- reducedDim, 157
- reportCellQC, 125
- reportDiffExp, 126
- reportDropletQC, 126
- reportQCTool, 127
- ReprocessedAllenData, 49
- ReprocessedFluidigmData, 49
- reticulate, 166–169
- retrieveFeatureIndex, 128
- retrieveFeatureInfo, 29
- retrieveSCEIndex, 128
- runANOVA, 129, 138
- runBarcodeRankDrops, 65, 66, 131
- runBBKNN, 132
- runBcDs, 69, 133
- runCellQC, 52, 53, 55, 56, 134
- runComBat, 135
- runCxds, 72, 136
- runCxdsBcDsHybrid, 95, 137
- runDEAnalysis, 126, 138
- runDecontX, 75, 139
- runDESeq2, 138, 140
- runDoubletCells, 82, 142
- runDoubletFinder, 84, 143
- runDropletQC, 144

- runEmptyDrops, [88](#), [145](#)
- runFastMNN, [146](#)
- runKMeans, [147](#)
- runLimmaBC, [148](#)
- runLimmaDE, [138](#), [149](#)
- runMAST, [138](#), [150](#)
- runMNNCorrect, [146](#), [152](#)
- runPCA, [34](#)
- runPerCellQC, [153](#)
- runSCANORAMA, [154](#)
- runSCMerge, [156](#)
- runScranSNN, [157](#)
- runScrublet, [86](#), [119](#), [158](#)
- RunUMAP, [182](#)
- runZINBWave, [160](#)

- sampleSummaryStats, [162](#)
- scater_logNormCounts, [162](#)
- scDblFinder, [142](#)
- sce, [163](#)
- sceBatches, [164](#)
- scran_modelGeneVar, [164](#)
- scRNAseq, [49](#)
- scSEGIndex, [156](#)
- sctkListGeneSetCollections, [165](#)
- sctkPythonInstallConda, [166](#), [169](#)
- sctkPythonInstallVirtualEnv, [167](#), [169](#)
- SCTransform, [183](#)
- SEG, [168](#)
- selectSCTKConda, [166](#), [167](#), [168](#)
- selectSCTKVirtualEnvironment, [167](#), [168](#), [169](#)
- seuratComputeHeatmap, [170](#)
- seuratComputeJackStraw, [171](#)
- seuratElbowPlot, [172](#)
- seuratFindClusters, [172](#)
- seuratFindHVG, [174](#)
- seuratHeatmapPlot, [174](#)
- seuratICA, [175](#)
- seuratIntegration, [176](#)
- seuratJackStrawPlot, [177](#)
- seuratNormalizeData, [177](#)
- seuratPCA, [178](#)
- seuratPlotHVG, [179](#)
- seuratReductionPlot, [180](#)
- seuratRunTSNE, [180](#)
- seuratRunUMAP, [181](#)
- seuratScaleData, [182](#)
- seuratSCTransform, [183](#)
- simpleLog, [184](#)
- SingleCellExperiment, [6–8](#), [12](#), [16–19](#), [21](#), [23](#), [25–29](#), [31–42](#), [49](#), [51–59](#), [61](#), [63](#), [65–67](#), [69](#), [71–73](#), [75](#), [77](#), [79](#), [80](#), [82](#), [84](#), [86](#), [88](#), [89](#), [92](#), [93](#), [95](#), [97](#), [98](#), [100–103](#), [105](#), [107](#), [110](#), [112](#), [114](#), [116](#), [117](#), [119–123](#), [125–142](#), [144–163](#), [165](#), [184](#), [186–188](#), [190](#)
- singleCellTK, [51](#), [53](#), [54](#), [56](#), [166–169](#), [184](#)
- subDiffEx, [185](#)
- subDiffExANOVA (subDiffEx), [185](#)
- subDiffExttest (subDiffEx), [185](#)
- subsetSCECols, [186](#)
- subsetSCERows, [187](#)
- SummarizedExperiment, [28](#), [29](#)
- summarizeSCE, [188](#)

- TENxPBMCData, [49](#)
- thresholdSCRNACountMatrix, [90](#)
- trimCounts, [189](#)

- umap, [139](#)
- unit, [111](#)

- virtualenv_create, [168](#)
- virtualenv_install, [167](#), [168](#)
- visPlot, [190](#)

- with_seed, [140](#)