Package ‘singleCellTK’

March 5, 2024

Type Package

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

Version 2.12.2

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

Description The Single Cell Toolkit (SCTK) in the singleCellTK package provides an interface to popular tools for importing, quality control, analysis, and visualization of single cell RNA-seq data. SCTK allows users to seamlessly integrate tools from various packages at different stages of the analysis workflow. A general "a la carte" workflow gives users the ability access to multiple methods for data importing, calculation of general QC metrics, doublet detection, ambient RNA estimation and removal, filtering, normalization, batch correction or integration, dimensionality reduction, 2-D embedding, clustering, marker detection, differential expression, cell type labeling, pathway analysis, and data exporting. Curated workflows can be used to run Seurat and Celda. Streamlined quality control can be performed on the command line using the SCTK-QC pipeline. Users can analyze their data using commands in the R console or by using an interactive Shiny Graphical User Interface (GUI). Specific analyses or entire workflows can be summarized and shared with comprehensive HTML reports generated by Rmarkdown. Additional documentation and vignettes can be found at camplab.net/sctk.

License MIT + file LICENSE

Encoding UTF-8

biocView SingleCell, GeneExpression, DifferentialExpression, Alignment, Clustering, ImmunoOncology, BatchEffect, Normalization, QualityControl, DataImport, GUI

LazyData FALSE

Imports ape, anndata, AnnotationHub, batchelor, BiocParallel, cellxex, colR, colorspace, cowplot, cluster, ComplexHeatmap, data.table, DelayedMatrixStats, DESeq2, dplyr, DT, ExperimentHub, ensembladb, fields, ggplot2, ggplotify, ggrepel, ggtree, gridExtra, GSVA (>= 1.26.0), GSVAdata, igraph, KernSmooth, limma, MAST, Matrix (>= 1.5-3), matrixStats, methods, msigdb, multtest, plotly, plyr, ROCR, Rtsne,
S4Vectors, scater, scMerge (>= 1.2.0), scran, Seurat (>= 3.1.3), shiny, shinyjs, SingleR, SoupX, sva, reshape2, shinyalert, circalize, enrichR (>= 3.2), celda, shinyssloaders, DropletUtils, scds (>= 1.2.0), reticulate (>= 1.14), tools, tximport, eds, withr, GSEABase, R.utils, zinbwave, scRNAseq (>= 2.0.2), TENxPBMCData, yaml, rmarkdown, magrittr, scDbIFinder, metap, VAM (>= 0.5.3), tibble, rlang, TSCAN, TrajectoryUtils, scuttle, utils, stats, zellkonverter

**RoxygenNote** 7.2.3

**Suggests** testthat, Rsubread, BiocStyle, knitr, lintr, spelling, org.Mm.eg.db, stringr, kableExtra, shinythemes, shinyBS, shinyjqui, shinyWidgets, shinyFiles, BiocGenerics, RColorBrewer, fastmap (>= 1.1.0), harmony, SeuratObject, optparse

**VignetteBuilder** knitr

**URL** https://www.camplab.net/sctk/

**BugReports** https://github.com/compbiomed/singleCellTK/issues

**Language** en-US

**git_url** https://git.bioconductor.org/packages/singleCellTK

**git_branch** RELEASE_3_18

**git_last_commit** 14c92130

**git_last_commit_date** 2024-02-05

**Repository** Bioconductor 3.18

**Date/Publication** 2024-03-04

**Author** Yichen Wang [aut] (<https://orcid.org/0000-0003-4347-5199>), Irzam Sarfraz [aut] (<https://orcid.org/0000-0001-8121-792X>), Rui Hong [aut], Yusuke Koga [aut], Salam Alabdullatif [aut], Nida Pervaiz [aut], David Jenkins [aut] (<https://orcid.org/0000-0002-7451-4288>), Vidya Akavoor [aut], Xinyun Cao [aut], Shruthi Bandyadka [aut], Anastasia Leshchyk [aut], Tyler Faits [aut], Mohammed Muzamil Khan [aut], Zhe Wang [aut], W. Evan Johnson [aut] (<https://orcid.org/0000-0002-6247-6595>), Ming Liu [aut], Joshua David Campbell [aut, cre] (<https://orcid.org/0000-0003-0780-8662>)

**Maintainer** Joshua David Campbell <camp@bu.edu>
### R topics documented:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>calcEffectSizes</td>
<td>8</td>
</tr>
<tr>
<td>combineSCE</td>
<td>8</td>
</tr>
<tr>
<td>computeHeatmap</td>
<td>9</td>
</tr>
<tr>
<td>computeZScore</td>
<td>10</td>
</tr>
<tr>
<td>constructSCE</td>
<td>11</td>
</tr>
<tr>
<td>convertSCEToSeurat</td>
<td>13</td>
</tr>
<tr>
<td>convertSeuratToSCE</td>
<td>13</td>
</tr>
<tr>
<td>dedupRowNames</td>
<td>14</td>
</tr>
<tr>
<td>detectCellOutlier</td>
<td>14</td>
</tr>
<tr>
<td>diffAbundanceFET</td>
<td>16</td>
</tr>
<tr>
<td>discreteColorPalette</td>
<td>17</td>
</tr>
<tr>
<td>distinctColors</td>
<td>17</td>
</tr>
<tr>
<td>downSampleCells</td>
<td>18</td>
</tr>
<tr>
<td>downSampleDepth</td>
<td>20</td>
</tr>
<tr>
<td>expData</td>
<td>21</td>
</tr>
<tr>
<td>expData,ANY,character-method</td>
<td>22</td>
</tr>
<tr>
<td>expData&lt;</td>
<td>22</td>
</tr>
<tr>
<td>expData&lt;-,ANY,character,CharacterOrNullOrMissing,logical-method</td>
<td>23</td>
</tr>
<tr>
<td>expDataNames</td>
<td>24</td>
</tr>
<tr>
<td>expDataNames,ANY-method</td>
<td>24</td>
</tr>
<tr>
<td>expDeleteDataTag</td>
<td>25</td>
</tr>
<tr>
<td>exportSCE</td>
<td>26</td>
</tr>
<tr>
<td>exportSCEtoAnnData</td>
<td>27</td>
</tr>
<tr>
<td>exportSCEtoFlatFile</td>
<td>28</td>
</tr>
<tr>
<td>exportSCEToSeurat</td>
<td>29</td>
</tr>
<tr>
<td>expSetDataTag</td>
<td>30</td>
</tr>
<tr>
<td>expTaggedData</td>
<td>30</td>
</tr>
<tr>
<td>featureIndex</td>
<td>31</td>
</tr>
<tr>
<td>generateHTANMeta</td>
<td>33</td>
</tr>
<tr>
<td>generateMeta</td>
<td>33</td>
</tr>
<tr>
<td>generateSimulatedData</td>
<td>34</td>
</tr>
<tr>
<td>getBiomarker</td>
<td>35</td>
</tr>
<tr>
<td>getDEGTopTable</td>
<td>36</td>
</tr>
<tr>
<td>getDiffAbundanceResults</td>
<td>37</td>
</tr>
<tr>
<td>getEnrichRResult&lt;-</td>
<td>38</td>
</tr>
<tr>
<td>getFindMarkerTopTable</td>
<td>39</td>
</tr>
<tr>
<td>getGenesetNamesFromCollection</td>
<td>40</td>
</tr>
<tr>
<td>getMSigDBTable</td>
<td>41</td>
</tr>
<tr>
<td>getPathwayResultNames</td>
<td>42</td>
</tr>
<tr>
<td>getSampleSummaryStatsTable</td>
<td>42</td>
</tr>
<tr>
<td>getSceParams</td>
<td>43</td>
</tr>
<tr>
<td>getSeuratVariableFeatures</td>
<td>44</td>
</tr>
<tr>
<td>getSoupX&lt;</td>
<td>45</td>
</tr>
<tr>
<td>getTopHVG</td>
<td>46</td>
</tr>
<tr>
<td>getTSCANResults</td>
<td>48</td>
</tr>
<tr>
<td>importAlevin</td>
<td>49</td>
</tr>
</tbody>
</table>
importAnnData .................................................. 50
importBUStools .................................................. 51
importCellRanger ................................................. 53
importCellRangerV2Sample ..................................... 56
importCellRangerV3Sample ..................................... 57
importDropEst ................................................... 58
importExampleData ............................................... 60
importFromFiles .................................................. 61
importGeneSetsFromCollection ................................ 63
importGeneSetsFromGMT ........................................ 64
importGeneSetsFromList ........................................ 66
importGeneSetsFromMSigDB ..................................... 68
importMitoGeneSet ................................................. 69
importMultipleSources .......................................... 71
importOptimus .................................................... 71
importSEQC ........................................................ 73
importSTAR solo .................................................. 74
iterateSimulations ............................................... 76
listSampleSummaryStatsTables .................................. 77
mergeSCEColData .................................................. 78
MitoGenes ......................................................... 79
mouseBrainSubsetSCE ............................................. 79
msigdb_table ...................................................... 80
plotBarcodeRankDropsResults ................................... 80
plotBarcodeRankScatter ......................................... 81
plotBatchCorrCompare .......................................... 83
plotBatchVariance ............................................... 84
plotBcdsResults ................................................. 85
plotBubble ....................................................... 88
plotClusterAbundance ...........................................) 89
plotCxdsResults .................................................. 90
plotDecontXResults .............................................. 93
plotDEGheatmap .................................................. 96
plotDEGRegression ............................................... 98
plotDEGViolin ................................................... 99
plotDEGVolcano .................................................. 101
plotDimRed ........................................................ 102
plotDoubletFinderResults ....................................... 103
plotEmptyDropsResults ......................................... 105
plotEmptyDropsScatter .......................................... 107
plotFindMarkerHeatmap ......................................... 109
plotMASTThresholdGenes ....................................... 112
plotPathway ...................................................... 113
plotPCA .......................................................... 114
plotRunPerCellQCResults ....................................... 115
plotScanpyDotPlot ............................................... 117
plotScanpyEmbedding ............................................ 119
plotScanpyHeatmap .............................................. 120
plotScanpyHVG .................................................. 121
plotScanpyMarkerGenes ........................................ 122
plotScanpyMarkerGenesDotPlot ................................ 123
plotScanpyMarkerGenesHeatmap ................................ 124
plotScanpyMarkerGenesMatrixPlot .............................. 125
plotScanpyMarkerGenesViolin .................................. 127
plotScanpyMatrixPlot ........................................... 128
plotScanpyPCA .................................................. 129
plotScanpyPCAGeneRanking ..................................... 130
plotScanpyPCAVariance ......................................... 131
plotScanpyViolin ................................................ 131
plotScDbfFinderResults .......................................... 132
plotSdsHybridResults .......................................... 135
plotSCEBarAssayData .......................................... 138
plotSCEBarColData ............................................. 140
plotSCEBatchFeatureMean ....................................... 141
plotSCEDensity .................................................. 142
plotSCEDensityAssayData ....................................... 144
plotSCEDensityColData ......................................... 145
plotSCEDimReduceColData ...................................... 147
plotSCEDimReduceFeatures ...................................... 150
plotSCEHeatmap .................................................. 152
plotSCEScatter ................................................... 155
plotSCEViolin ..................................................... 158
plotSCEViolinAssayData ....................................... 160
plotSCEViolinColData .......................................... 163
plotScrubletResults ............................................ 165
plotSeuratElbow ............................................... 168
plotSeuratGenes ................................................. 169
plotSeuratHeatmap .............................................. 170
plotSeuratHVG .................................................... 171
plotSeuratJackStraw ............................................ 171
plotSeuratReduction ............................................. 172
plotSoupXResults ............................................... 173
plotTopHVG ....................................................... 175
plotTSCANClusterDEG .......................................... 177
plotTSCANClusterPseudo ....................................... 178
plotTSCANDimReduceFeatures .................................. 179
plotTSCANPseudotimeGenes .................................... 180
plotTSCANPseudotimeHeatmap ................................ 182
plotTSCANResults .............................................. 183
plotTSNE .......................................................... 184
plotUMAP .......................................................... 185
qcInputProcess .................................................. 186
readSingleCellMatrix .......................................... 187
reportCellIQC .................................................... 188
reportClusterAbundance ........................................ 189
reportDiffAbundanceFET ....................................... 190
R topics documented:

- reportDiffExp .......................................................... 191
- reportDropletQC .......................................................... 192
- reportFindMarker .......................................................... 193
- reportQCTool ................................................................. 193
- reportSeurat ................................................................. 194
- reportSeuratClustering ...................................................... 197
- reportSeuratDimRed ......................................................... 198
- reportSeuratFeatureSelection ............................................ 200
- reportSeuratMarkerSelection ............................................. 201
- reportSeuratNormalization ............................................... 202
- reportSeuratResults ......................................................... 203
- reportSeuratRun ............................................................. 205
- reportSeuratScaling ........................................................ 208
- retrieveSCEIndex ........................................................... 209
- runBarcodeRankDrops ....................................................... 210
- runBBKNN ................................................................. 211
- runBcds ................................................................. 212
- runCellQC ................................................................. 213
- runClusterSummaryMetrics ............................................... 215
- runComBatSeq ............................................................... 216
- runCxds ................................................................. 217
- runCxdsBcdsHybrid ........................................................ 219
- runDEAnalysis .............................................................. 220
- runDecontX ................................................................. 224
- runDimReduce .............................................................. 227
- runDoubletFinder ........................................................... 228
- runDropletQC .............................................................. 229
- runEmptyDrops .............................................................. 230
- runEnrichR ................................................................. 232
- runFastMNN ................................................................. 233
- runFeatureSelection ......................................................... 235
- runFindMarker .............................................................. 236
- runGSVA ................................................................. 238
- runHarmony ................................................................. 239
- runKMeans ................................................................. 241
- runLimmaBC ................................................................. 242
- runMNNCorrect .............................................................. 243
- runModelGeneVar ........................................................... 244
- runNormalization .......................................................... 245
- runPerCellQC .............................................................. 247
- runSCANORAMA ............................................................ 249
- runScanpyFindClusters .................................................... 250
- runScanpyFindHVG ........................................................ 252
- runScanpyFindMarkers ..................................................... 253
- runScanpyNormalizeData .................................................. 254
- runScanpyPCA ............................................................... 255
- runScanpyScaleData ........................................................ 256
- runScanpyTSNE .............................................................. 257
```
<table>
<thead>
<tr>
<th>R topics documented:</th>
</tr>
</thead>
<tbody>
<tr>
<td>runScanpyUMAP</td>
</tr>
<tr>
<td>runScDbiFinder</td>
</tr>
<tr>
<td>runSCMerge</td>
</tr>
<tr>
<td>runScranSNN</td>
</tr>
<tr>
<td>runScrublet</td>
</tr>
<tr>
<td>runSeuratFindClusters</td>
</tr>
<tr>
<td>runSeuratFindHVG</td>
</tr>
<tr>
<td>runSeuratFindMarkers</td>
</tr>
<tr>
<td>runSeuratHeatmap</td>
</tr>
<tr>
<td>runSeuratICA</td>
</tr>
<tr>
<td>runSeuratIntegration</td>
</tr>
<tr>
<td>runSeuratJackStraw</td>
</tr>
<tr>
<td>runSeuratNormalizeData</td>
</tr>
<tr>
<td>runSeuratPCA</td>
</tr>
<tr>
<td>runSeuratScaleData</td>
</tr>
<tr>
<td>runSeuratSCTransform</td>
</tr>
<tr>
<td>runSeuratTSNE</td>
</tr>
<tr>
<td>runSeuratUMAP</td>
</tr>
<tr>
<td>runSingleR</td>
</tr>
<tr>
<td>runSoupX</td>
</tr>
<tr>
<td>runTSCAN</td>
</tr>
<tr>
<td>runTSCANClusterDEAnalysis</td>
</tr>
<tr>
<td>runTSCANDEG</td>
</tr>
<tr>
<td>runTSNE</td>
</tr>
<tr>
<td>runUMAP</td>
</tr>
<tr>
<td>runVAM</td>
</tr>
<tr>
<td>runZINBWaVE</td>
</tr>
<tr>
<td>sampleSummaryStats</td>
</tr>
<tr>
<td>scaterCPM</td>
</tr>
<tr>
<td>scaterlogNormCounts</td>
</tr>
<tr>
<td>scaterPCA</td>
</tr>
<tr>
<td>sce</td>
</tr>
<tr>
<td>sceBatches</td>
</tr>
<tr>
<td>sctkListGeneSetCollections</td>
</tr>
<tr>
<td>sctkPythonInstallConda</td>
</tr>
<tr>
<td>sctkPythonInstallVirtualEnv</td>
</tr>
<tr>
<td>SEG</td>
</tr>
<tr>
<td>selectSCTKConda</td>
</tr>
<tr>
<td>selectSCTKVirtualEnvironment</td>
</tr>
<tr>
<td>setRowNames</td>
</tr>
<tr>
<td>setSCTKDisplayRow</td>
</tr>
<tr>
<td>singleCellTK</td>
</tr>
<tr>
<td>subDiffEx</td>
</tr>
<tr>
<td>subsetSCECols</td>
</tr>
<tr>
<td>subsetSCERows</td>
</tr>
<tr>
<td>summarizeSCE</td>
</tr>
<tr>
<td>trimCounts</td>
</tr>
<tr>
<td>Index</td>
</tr>
</tbody>
</table>
```
calcEffectSizes  

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

Description

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

Usage

\[ \text{calcEffectSizes(countMatrix, condition)} \]

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

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

combineSCE  

Combine a list of SingleCellExperiment objects as one SingleCellExperiment object

Description

Combine a list of SingleCellExperiment objects as one SingleCellExperiment object

Usage

\[ \text{combineSCE(sceList, by.r = NULL, by.c = NULL, combined = TRUE)} \]
computeHeatmap

Arguments

sceList A list contains SingleCellExperiment objects. Currently, combineSCE function only support combining SCE objects with assay in dgCMatrix format. It does not support combining SCE with assay in delayedArray format.

by.r Specifications of the columns used for merging rowData. If set as NULL, the rownames of rowData tables will be used to merging rowData. Default is NULL.

by.c Specifications of the columns used for merging colData. If set as NULL, the rownames of colData tables will be used to merging colData. Default is NULL.

combined logical; if TRUE, it will combine the list of SingleCellExperiment objects and return a SingleCellExperiment. If FALSE, it will return a list of SingleCellExperiment whose rowData, colData, assay and reducedDim data slot are compatible within SCE objects in the list. Default is TRUE.

Value

A SingleCellExperiment object which combines all objects in sceList. The colData is merged.

Examples

data(scExample, package = "singleCellTK")
combinedsce <- combineSCE(list(sce,sce), by.r = NULL, by.c = NULL, combined = TRUE)

computeHeatmap

Computes heatmap for a set of features against dimensionality reduction components

Description

The computeHeatmap method computes the heatmap visualization for a set of features against a set of dimensionality reduction components. This method uses the heatmap computation algorithm code from Seurat but plots the heatmap using ComplexHeatmap and cowplot libraries.

Usage

computeHeatmap(
  inSCE,
  useAssay,
  dims = 10,
  nfeatures = 30,
  cells = NULL,
  reduction = "pca",
  disp.min = -2.5,
  disp.max = 2.5,
  balanced = TRUE,
  nCol = NULL,
  externalReduction = NULL
)
Arguments

inSCE Input SingleCellExperiment object.
useAssay Specify the name of the assay that will be scaled by this function for the features that are used in the heatmap.
dims Specify the number of dimensions to use for heatmap. Default 10.
nfeatures Specify the number of features to use for heatmap. Default is 30.
cells Specify the samples/cells to use for heatmap computation. Default is NULL which will utilize all samples in the assay.
reduction Specify the reduction slot in the input object. Default is "pca".
disp.min Specify the minimum dispersion value to use for floor clipping of assay values. Default is -2.5.
disp.max Specify the maximum dispersion value to use for ceiling clipping of assay values. Default is 2.5.
balanced Specify if the number of up-regulated and down-regulated features should be balanced. Default is TRUE.
nCol Specify the number of columns in the output plot. Default is NULL which will auto-compute the number of columns.
externalReduction Specify an external reduction if not present in the input object. This external reduction should be created using CreateDimReducObject function.

Value

Heatmap plot object.

computeZScore Compute Z-Score

Description

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

Usage

computeZScore(counts)

Arguments

counts matrix (base matrix, dgCMatrix or DelayedMatrix)
constructSCE

Value

z-score computed counts matrix (DelayedMatrix)

Examples

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

---

| constructSCE | Create SingleCellExperiment object from csv or txt input |

Description

Create SingleCellExperiment object from csv or txt input

Usage

```r
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.

---

convertSCEToSeurat

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

Description

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

convertSCEToSeurat(
  inSCE,
  countsAssay = NULL,
  normAssay = NULL,
  scaledAssay = NULL,
  copyColData = FALSE,
  copyReducedDim = FALSE,
  copyDecontX = FALSE,
  pcaReducedDim = NULL,
  icaReducedDim = NULL,
  tsneReducedDim = NULL,
  umapReducedDim = 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.
copyColData Boolean. Whether copy 'colData' of SCE object to the 'meta.data' of Seurat object. Default FALSE.
copyReducedDim Boolean. Whether copy 'reducedDims' of the SCE object to the 'reductions' of Seurat object. Default FALSE.
copyDecontX Boolean. Whether copy 'decontXcounts' assay of the SCE object to the 'assays' of Seurat object. Default TRUE.
pcaReducedDim Specify a character value indicating the name of the reducedDim to store as default pca computation in the output seurat object. Default is NULL which will not store any reducedDim as the default pca. This will only work when copyReducedDim parameter is set to TRUE.
icaReducedDim Specify a character value indicating the name of the reducedDim to store as default ica computation in the output seurat object. Default is NULL which will not store any reducedDim as the default ica. This will only work when copyReducedDim parameter is set to TRUE.
tsneReducedDim Specify a character value indicating the name of the reducedDim to store as default tsne computation in the output seurat object. Default is NULL which will not store any reducedDim as the default tsne. This will only work when copyReducedDim parameter is set to TRUE.
umapReducedDim Specify a character value indicating the name of the reducedDim to store as default umap computation in the output seurat object. Default is NULL which will not store any reducedDim as the default umap. This will only work when copyReducedDim parameter is set to TRUE.

Value

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

Examples

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

convertSeuratToSCE

convertSeuratToSCE Converts the input seurat object to a sce object

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)
**dedupRowNames**

*Deduplicate the rownames of a matrix or SingleCellExperiment object*

**Description**

Adds ‘-1’, ‘-2’, ... ‘-i’ to multiple duplicated rownames, and in place replace the unique rownames, store unique rownames in `rowData`, or return the unique rownames as character vector.

**Usage**

```r
dedupRowNames(x, as.rowData = FALSE, return.list = FALSE)
```

**Arguments**

- `x`: A matrix like or `SingleCellExperiment` object, on which we can apply `rownames()` to and has duplicated rownames.
- `as.rowData`: Only applicable when `x` is a `SingleCellExperiment` object. When set to `TRUE`, will insert a new column called ”rownames.uniq” to `rowData(x)`, with the deduplicated rownames.
- `return.list`: When set to `TRUE`, will return a character vector of the deduplicated rownames.

**Value**

By default, a matrix or `SingleCellExperiment` object with rownames deduplicated. When `x` is a `SingleCellExperiment` and `as.rowData` is set to `TRUE`, will return `x` with `rowData` updated. When `return.list` is set to `TRUE`, will return a character vector with the deduplicated rownames.

**Examples**

```r
data("scExample", package = "singleCellTK")
sce <- dedupRowNames(sce)
```

**detectCellOutlier**

*Detecting outliers within the SingleCellExperiment object.*

**Description**

A wrapper function for `isOutlier`. Identify outliers from numeric vectors stored in the `SingleCellExperiment` object.
detectCellOutlier

Usage

detectCellOutlier(
  inSCE,
  slotName,
  itemName,
  sample = NULL,
  nmads = 3,
  type = "both",
  overwrite = TRUE
)

Arguments

inSCE A SingleCellExperiment object.
slotName Desired slot of SingleCellExperiment used for plotting. Possible options: "assays", "colData", "metadata", "reducedDims". Required.
itemName Desired vector within the slot used for plotting. Required.
sample A single character specifying a name that can be found in colData(inSCE) to directly use the cell annotation; or a character vector with as many elements as cells to indicates which sample each cell belongs to. Default NULL. decontX will be run on cells from each sample separately.
mad Integer. Number of median absolute deviation. Parameter may be adjusted for more lenient or stringent outlier cutoff. Default 3.
type Character. Type/direction of outlier detection; whether the lower/higher outliers should be detected, or both. Options are "both", "lower", "higher".
overwrite Boolean. If TRUE, and this function has previously generated an outlier decision on the same itemName, the outlier decision will be overwritten. Default TRUE.

Value

A SingleCellExperiment object with " 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[,sample(ncol(sce),20)])
sce <- detectCellOutlier(sce, slotName = "colData", sample = sce$sample,
  nmads = 4, itemName = "decontX_contamination", type = "both")
Calculate Differential Abundance with FET

**Description**

Calculate Differential Abundance with FET

**Usage**

```r
diffAbundanceFET(inSCE, cluster, variable, control, case, analysisName)
```

**Arguments**

- `inSCE`: A `SingleCellExperiment` object.
- `cluster`: A single character, specifying the name to store the cluster label in `colData`.
- `variable`: A single character, specifying the name to store the phenotype labels in `colData`.
- `control`: A single character. Specifying one or more categories that can be found in the vector specified by `variable`.
- `case`: A single character. Specifying one or more categories that can be found in the vector specified by `variable`.
- `analysisName`: A single character. Will be used for naming the result table, which will be saved in metadata slot.

**Details**

This function will calculate the cell counting and fraction by dividing all cells to groups specified by the arguments, together with statistical summary by performing Fisher Exact Tests (FET).

**Value**

The original `SingleCellExperiment` object with metadata(`inSCE`) updated with a list `diffAbundanceFET`, containing a new data.frame for the analysis result, named by `analysisName`. The data.frame contains columns for number and fraction of cells that belong to different cases, as well as "Odds_Ratio", "PValue" and "FDR".

**Examples**

```r
data("mouseBrainSubsetSCE", package = "singleCellTK")
mouseBrainSubsetSCE <- diffAbundanceFET(inSCE = mouseBrainSubsetSCE, 
  cluster = "tissue",
  variable = "level1class",
  case = "oligodendrocytes",
  control = "microglia",
  analysisName = "diffAbundFET")```
discreteColorPalette  

Generate given number of color codes

Description

Three different generation methods are wrapped, including distinctColors, [randomcolor](SCTK_PerformingQC_Cell_V3.Rmd) 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

Usage

downSampleCells(  
  originalData,
  useAssay = "counts",
  minCountDetec = 10,
  minCellsDetec = 3,
  minCellnum = 10,
  maxCellnum = 1000,
  realLabels,
  depthResolution = 10,
)
downSampleCells

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[seq(100),]
res <- downSampleCells(subset,
  realLabels = "level1class",
  iterations=2)
downSampleDepth

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

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 mediansignificant effect size in each simulation. If no genes are significantly differentially expressed, the median effect size defaults to infinity.
expData

Examples

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

expData

expData Get data item from an input SingleCellExperiment object. The data item can be an assay, altExp (subset) or a reducedDim, which is retrieved based on the name of the data item.

Description

expData Get data item from an input SingleCellExperiment object. The data item can be an assay, altExp (subset) or a reducedDim, which is retrieved based on the name of the data item.

Usage

expData(inSCE, assayName)

Arguments

inSCE Input SingleCellExperiment object.
assayName Specify the name of the data item to retrieve.

Value

Specified data item.

Examples

data(scExample, package = "singleCellTK")
mat <- expData(sce, "counts")
expData<- expData Get data item from an input SingleCellExperiment object. The data item can be an assay, altExp (subset) or a reducedDim, which is retrieved based on the name of the data item.

Description
expData Get data item from an input SingleCellExperiment object. The data item can be an assay, altExp (subset) or a reducedDim, which is retrieved based on the name of the data item.

Usage
## S4 method for signature 'ANY,character'
expData(inSCE, assayName)

Arguments

inSCE Input SingleCellExperiment object.
assayName Specify the name of the data item to retrieve.

Value
Specified data item.

Examples
data(scExample, package = "singleCellTK")
mat <- expData(sce, "counts")

expData<- expData Store data items using tags to identify the type of data item stored. To be used as a replacement for assay<- setter function but with additional parameter to set a tag to a data item.

Description
expData Store data items using tags to identify the type of data item stored. To be used as a replacement for assay<- setter function but with additional parameter to set a tag to a data item.

Usage
expData(inSCE, assayName, tag = NULL, altExp = FALSE) <- value
Arguments

inSCE Input SingleCellExperiment object.
assayName Specify the name of the input assay.
tag Specify the tag to store against the input assay. Default is NULL, which will set the tag to "uncategorized".
altExp A logical value indicating if the input assay is an altExp or a subset assay.
value An input matrix-like value to store in the SCE object.

Value

A SingleCellExperiment object containing the newly stored data.

Examples

data(scExample, package = "singleCellTK")
mat <- expData(sce, "counts")
expData(sce, "counts", tag = "raw") <- mat

Description

expData Store data items using tags to identify the type of data item stored. To be used as a replacement for assay<- setter function but with additional parameter to set a tag to a data item.

Usage

## S4 replacement method for signature 'ANY,character,CharacterOrNullOrMissing,logical'
expData(inSCE, assayName, tag = NULL, altExp = FALSE) <- value

Arguments

inSCE Input SingleCellExperiment object.
assayName Specify the name of the input assay.
tag Specify the tag to store against the input assay. Default is NULL, which will set the tag to "uncategorized".
altExp A logical value indicating if the input assay is an altExp or a subset assay.
value An input matrix-like value to store in the SCE object.

Value

A SingleCellExperiment object containing the newly stored data.
expDataNames, ANY-method

expDataNames

expDataNames Get names of all the data items in the input
SingleCellExperiment object including assays, altExps and reducedDims.

Description

expDataNames Get names of all the data items in the input SingleCellExperiment object including assays, altExps and reducedDims.

Usage

expDataNames(inSCE)

Arguments

inSCE Input SingleCellExperiment object.

Value

A combined vector of assayNames, altExpNames and reducedDimNames.

Examples

data(scExample, package = "singleCellTK")
expDataNames(sce)

Description

expDataNames Get names of all the data items in the input SingleCellExperiment object including assays, altExps and reducedDims.

Usage

## S4 method for signature 'ANY'
expDataNames(inSCE)
expDeleteDataTag

Arguments

inSCE Input SingleCellExperiment object.

Value

A combined vector of assayNames, altExpNames and reducedDimNames.

Examples

data(scExample, package = "singleCellTK")
expDataNames(sce)

Description

expDeleteDataTag Remove tag against an input data from the stored tag information in the metadata of the input object.

Usage

expDeleteDataTag(inSCE, assay)

Arguments

inSCE Input SingleCellExperiment object.
assay Name of the assay or the data item against which a tag should be removed.

Value

The input SingleCellExperiment object with tag information removed from the metadata slot.

Examples

data(scExample, package = "singleCellTK")
sce <- expSetDataTag(sce, "raw", "counts")
sce <- expDeleteDataTag(sce, "counts")
**Description**

Export data in SingleCellExperiment object

**Usage**

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

**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**

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

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

Arguments

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

Value

Generates a Python anndata object containing data from inSCE.
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,
  prefix = "SCE"
)

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.
prefix Prefix of file names.

Value

Generates text files containing data from inSCE.

Examples

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

## End(Not run)
Export data in Seurat object

Description

Export data in Seurat object

Usage

```r
exportSCEToSeurat(
  inSCE,  
  prefix = "sample",  
  outputDir = "./",  
  overwrite = TRUE,  
  copyColData = TRUE,  
  copyReducedDim = TRUE,  
  copyDecontX = TRUE
)
```

Arguments

- **inSCE**: A `SingleCellExperiment` object that contains the data. QC metrics are stored in `colData` of the `singleCellExperiment` object.
- **prefix**: Prefix to use for the name of the output file. Default "sample".
- **outputDir**: Path to the directory where outputs will be written. Default is the current working directory.
- **overwrite**: Boolean. Whether overwrite the output if it already exists in the `outputDir`. Default `TRUE`.
- **copyColData**: Boolean. Whether copy `colData` of SCE object to the `meta.data` of Seurat object. Default `TRUE`.
- **copyReducedDim**: Boolean. Whether copy `reducedDims` of the SCE object to the `reductions` of Seurat object. Default `TRUE`.
- **copyDecontX**: Boolean. Whether copy `decontXcounts` assay of the SCE object to the `assays` of Seurat object. Default `TRUE`.

Value

Generates a Seurat object containing data from `inSCE`. 
expSetDataTag

Set tag to an assay or a data item in the input SCE object.

Description

expSetDataTag Set tag to an assay or a data item in the input SCE object.

Usage

expSetDataTag(inSCE, assayType, assays)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inSCE</td>
<td>Input SingleCellExperiment object.</td>
</tr>
<tr>
<td>assayType</td>
<td>Specify a character(1) value as a tag that should be set against a data item.</td>
</tr>
<tr>
<td>assays</td>
<td>Specify name(s) character() of data item(s) against which the tag should be set.</td>
</tr>
</tbody>
</table>

Value

The input SingleCellExperiment object with tag information stored in the metadata slot.

Examples

data(scExample, package = "singleCellTK")
sce <- expSetDataTag(sce, "raw", "counts")

expTaggedData

Returns a list of names of data items from the input SingleCellExperiment object based upon the input parameters.

Description

expTaggedData Returns a list of names of data items from the input SingleCellExperiment object based upon the input parameters.

Usage

expTaggedData(
inSCE,
tags = NULL,
redDims = FALSE,
recommended = NULL,
showTags = TRUE
)

**Arguments**

- `inSCE` Input SingleCellExperiment object.
- `tags` A character() value indicating if the data items should be returned separated by the specified tags. Default is NULL indicating that returned names of the data items are simply returned as a list with default tag as "uncategorized".
- `redDims` A logical value indicating if reducedDims should be returned as well separated with 'redDims' tag.
- `recommended` A character() vector indicating the tags that should be displayed as recommended. Default is NULL.
- `showTags` A logical value indicating if the tags should be shown. If FALSE, output is just a simple list, not separated by tags.

**Value**

A list of names of data items specified by the other parameters.

**Examples**

```r
data(scExample, package = "singleCellTK")
sce <- expSetDataTag(sce, "raw", "counts")
tags <- expTaggedData(sce)
```

---

**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**

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

*Generate HTAN manifest file for droplet and cell count data*

**Description**

Generate HTAN manifest file for droplet and cell count data

**Usage**

```r
generateHTANMeta(
  dropletSCE = NULL,
  cellSCE = NULL,
  samplename, 
  htan_biospecimen_id, 
  dir, 
  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
- `htan_biospecimen_id`: The HTAN biospecimen id of the sample in `SingleCellExperiment` object
- `dir`: The output directory of the SCTK QC pipeline.
- `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.

---

**generateMeta**

*Generate HTAN manifest file for droplet and cell count data*

**Description**

Generate HTAN 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 generates manifest file including HTAN specific ID (HTAN Biospecimen ID, HTAN parent file ID and HTAN patient ID). 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.

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.
getBiomarker

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

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

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",  
  featureLocation = NULL, 
  featureDisplay = NULL 
)

Arguments

inSCE Input SingleCellExperiment object.
gene gene list
binary "Binary" for binary expression or "Continuous" for a gradient. Default: "Binary"
useAssay Indicates which assay to use. The default is "counts".
featureLocation Indicates which column name of rowData to query gene.
featureDisplay Indicates which column name of rowData to use to display feature for visualization.

Value

getBiomarker(): A data.frame of expression values
Examples

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

getDEGTopTable

Get Top Table of a DEG analysis

Description

Users have to run runDEAnalysis() first, any of the wrapped functions of this generic function. Users can set further filters on the result. A data.frame object, with variables of Gene, Log2_FC, Pvalue, and FDR, will be returned.

Usage

getDEGTopTable(
  inSCE, 
  useResult, 
  labelBy = S4Vectors::metadata(inSCE)$featureDisplay, 
  onlyPos = FALSE, 
  log2fcThreshold = 0.25, 
  fdrThreshold = 0.05, 
  minGroup1MeanExp = NULL, 
  maxGroup2MeanExp = NULL, 
  minGroup1ExprPerc = NULL, 
  maxGroup2ExprPerc = NULL 
)

Arguments

inSCE

SingleCellExperiment inherited object, with of the singleCellITK DEG method performed in advance.

useResult

character. A string specifying the analysisName used when running a differential expression analysis function.

labelBy

A single character for a column of rowData(inSCE) as where to search for the labeling text. Leave NULL for rownames. Default metadata(inSCE)$featureDisplay (see setSCTKDisplayRow).

onlyPos

logical. Whether to only fetch DEG with positive log2_FC value. Default FALSE.

log2fcThreshold

numeric. Only fetch DEGs with the absolute values of log2FC larger than this value. Default 0.25.

fdrThreshold

numeric. Only fetch DEGs with FDR value smaller than this value. Default 0.05.
**getDiffAbundanceResults**

**Description**

Get/Set diffAbundanceFET result table

**Usage**

```r
getDiffAbundanceResults(x, analysisName)

# S4 method for signature 'SingleCellExperiment'
getDiffAbundanceResults(x, analysisName)

getDiffAbundanceResults(x, analysisName) <- value

# S4 replacement method for signature 'SingleCellExperiment'
getDiffAbundanceResults(x, analysisName) <- value
```

**Value**

A data.frame object of the top DEGs, with variables of Gene, Log2.FC, Pvalue, and FDR.

**Examples**

```r
data("sceBatches")
sceBatches <- scaterlogNormCounts(sceBatches, "logcounts")
sce.w <- subsetSCECols(sceBatches, colData = "batch == 'w'")
sce.w <- runWilcox(sce.w, class = "cell_type", classGroup1 = "alpha",
                   groupName1 = "w.alpha", groupName2 = "w.beta",
                   analysisName = "w.aVSb")
getDEGTopTable(sce.w, "w.aVSb")
```
getEnrichRResult<-  

Arguments

x  A SingleCellExperiment object.
analysisName  A single character string specifying an analysis performed with diffAbundanceFET
value  The output table of diffAbundanceFET

Value

The differential abundance table for getter method, or update the SCE object with new result for setter method.

Examples

data("mouseBrainSubsetSCE", package = "singleCellTK")
mouseBrainSubsetSCE <- diffAbundanceFET(inSCE = mouseBrainSubsetSCE,
  cluster = "tissue",
  variable = "level1class",
  case = "oligodendrocytes",
  control = "microglia",
  analysisName = "diffAbund")
result <- getDiffAbundanceResults(mouseBrainSubsetSCE, "diffAbund")

getEnrichRResult<-  Get or Set EnrichR Result

Description

Get or Set EnrichR Result

Usage

getEnrichRResult(inSCE, analysisName) <- value
getEnrichRResult(inSCE, analysisName)

## S4 method for signature 'SingleCellExperiment'
getEnrichRResult(inSCE, analysisName)

## S4 replacement method for signature 'SingleCellExperiment'
getEnrichRResult(inSCE, analysisName) <- value

Arguments

inSCE  A SingleCellExperiment object.
analysisName  A string that identifies each specific analysis
value  The EnrichR result table
getFindMarkerTopTable

**Value**

For getter method, a data.frame of the EnrichR result; For setter method, inSCE with EnrichR results updated.

**See Also**

runEnrichR

**Examples**

data("mouseBrainSubsetSCE")
if (Biobase::testBioCConnection()) {
    mouseBrainSubsetSCE <- runEnrichR(mouseBrainSubsetSCE, features = "Cmtm5", 
        db = "GO_Cellular_Component_2017", 
        analysisName = "analysis1")

    result <- getEnrichRResult(mouseBrainSubsetSCE, "analysis1")
}

getFindMarkerTopTable  Fetch the table of top markers that pass the filtering

**Description**

Fetch the table of top markers that pass the filtering

**Usage**

getFindMarkerTopTable(
    inSCE,
    log2fcThreshold = 1,
    fdrThreshold = 0.05,
    minClustExprPerc = 0.7,
    maxCtrlExprPerc = 0.4,
    minMeanExpr = 1,
    topN = 10
)

findMarkerTopTable(
    inSCE,
    log2fcThreshold = 1,
    fdrThreshold = 0.05,
    minClustExprPerc = 0.7,
    maxCtrlExprPerc = 0.4,
    minMeanExpr = 1,
    topN = 10
)
Arguments

- `inSCE` (SingleCellExperiment inherited object):
  - **log2fcThreshold**: Only use DEGs with the absolute values of log2FC larger than this value. Default 1
  - **fdrThreshold**: Only use DEGs with FDR value smaller than this value. Default 0.05
  - **minClustExprPerc**: A numeric scalar. The minimum cutoff of the percentage of cells in the cluster of interests that expressed the marker gene. Default 0.7.
  - **maxCtrlExprPerc**: A numeric scalar. The maximum cutoff of the percentage of cells out of the cluster (control group) that expressed the marker gene. Default 0.4.
  - **minMeanExpr**: A numeric scalar. The minimum cutoff of the mean expression value of the marker in the cluster of interests. Default 1.
  - **topN**: An integer. Only to fetch 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.

Details

Users have to run `runFindMarker` prior to using this function to extract a top marker table.

Value

An organized data.frame object, with the top marker gene information.

See Also

`runFindMarker`, `plotFindMarkerHeatmap`

Examples

```r
data("mouseBrainSubsetSCE", package = "singleCellTK")
mouseBrainSubsetSCE <- runFindMarker(mouseBrainSubsetSCE, 
  useAssay = "logcounts", 
  cluster = "level1class")
getFindMarkerTopTable(mouseBrainSubsetSCE)
```

description:

List geneset names from `geneSetCollection`

getGenesetNamesFromCollection

List geneset names from `geneSetCollection`
getMSigDBTable

Usage

getGenesetNamesFromCollection(inSCE, geneSetCollectionName)

Arguments

inSCE Input SingleCellExperiment object.

Arguments
geneSetCollectionName The name of an imported geneSetCollection.

Value

A character vector of available genesets from the collection.

getMSigDBTable Shows MSigDB categories

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 importGeneSetsFromM-SigDB.

Usage

getMSigDBTable()

Value

data.frame, containing MSigDB categories

Author(s)

Joshua D. Campbell

See Also

importGeneSetsFromMSigDB for importing MSigDB gene sets.

Examples

getMSigDBTable()
Description

List pathway analysis result names

Usage

getPathwayResultNames(inSCE, stopIfNone = FALSE, verbose = FALSE)

Arguments

inSCE Input SingleCellExperiment object.
stopIfNone Whether to stop and raise an error if no results found. If FALSE, will return an empty character vector.
verbose Show warning if no result found. Default FALSE

Details

Pathway analysis results will be stored as matrices in reducedDims slot of inSCE. This function lists the result names stored in metadata slot when analysis is performed.

Value

A character vector of valid pathway analysis result names.

Examples

data(scExample)
getPathwayResultNames(sce)

Description

Stores and returns table of SCTK QC outputs to metadata.

Stores and returns table of QC metrics generated from QC algorithms within the metadata slot of the SingleCellExperiment object.
getSceParams

Usage

getSampleSummaryStatsTable(inSCE, statsName, ...)

setSampleSummaryStatsTable(inSCE, statsName, ...) <- value

## S4 method for signature 'SingleCellExperiment'
getSampleSummaryStatsTable(inSCE, statsName, ...)

## S4 replacement method for signature 'SingleCellExperiment'
setSampleSummaryStatsTable(inSCE, statsName, ...) <- value

Arguments

inSCE Input SingleCellExperiment object with saved assay data and/or colData data. Required.
statsName A character value indicating the slot that stores the stats table within the metadata of the SingleCellExperiment object. Required.
... Other arguments passed to the function.
value The summary table for QC statistics generated from SingleCellTK to be added to the SCE object.

Value

For getSampleSummaryStatsTable, A matrix/array object. Contains a summary table for QC statistics generated from SingleCellTK. For setSampleSummaryStatsTable<-, A SingleCellExperiment object where the summary table is updated in the metadata slot.

Examples

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

getSceParams  Extract QC parameters from the SingleCellExperiment object

Description

Extract QC parameters from the SingleCellExperiment object
Usage

getSceParams(
  inSCE,
  skip = c("runScrublet", "runDecontX", "runBarcodeRanksMetaOutput", "genesets", "runSoupX"),
  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.

getSeuratVariableFeatures

Get variable feature names after running runSeuratFindHVG function

Description

Get variable feature names after running runSeuratFindHVG function

Usage

gtSeuratVariableFeatures(inSCE)

Arguments

inSCE Input SingleCellExperiment object.

Value

A list of variable feature names.
**getSoupX<-**  
*Get or Set SoupX Result*

**Description**

S4 method for getting and setting SoupX results that cannot be appended to either `rowData(inSCE)` or `colData(inSCE)`.

S4 method for getting and setting SoupX results that cannot be appended to either `rowData(inSCE)` or `colData(inSCE)`.

**Usage**

```r
getSoupX(inSCE, sampleID, background = FALSE) <- value
getSoupX(inSCE, sampleID = NULL, background = FALSE)
## S4 method for signature 'SingleCellExperiment'
getSoupX(inSCE, sampleID = NULL, background = FALSE)
## S4 replacement method for signature 'SingleCellExperiment'
getSoupX(inSCE, sampleID, background = FALSE) <- value
```

**Arguments**

- **inSCE**
  - A `SingleCellExperiment` object. For getter method, `runSoupX` must have been already applied.

- **sampleID**
  - Character vector. For getter method, the samples that should be included in the returned list. Leave this `NULL` for all samples. Default `NULL`. For setter method, only one sample allowed.

- **background**
  - Logical. Whether background was applied when running `runSoupX`. Default `FALSE`.

- **value**
  - Dedicated list object of SoupX results.

**Value**

For getter method, a list with SoupX results for specified samples. For setter method, `inSCE` with SoupX results updated.

For getter method, a list with SoupX results for specified samples. For setter method, `inSCE` with SoupX results updated.

**See Also**

`runSoupX`, `plotSoupXResults`
getTopHVG

## Not run:
sce <- importExampleData("pbmc3k")
sce <- runSoupX(sce, sample = "sample")
soupXResults <- getSoupX(sce)
## End(Not run)

### Description
Extracts or select the top variable genes from an input SingleCellExperiment object. Note that the variability metrics must be computed using the runFeatureSelection method before extracting the feature names of the top variable features. getTopHVG only returns a character vector of the HVG selection, while with setTopHVG, a logical vector of the selection will be saved in the rowData, and optionally, a subset object for the HVGs can be stored in the altExps slot at the same time.

### Usage
```
getTopHVG(
  inSCE,
  method = c("vst", "dispersion", "mean.var.plot", "modelGeneVar", "seurat", "seurat_v3", "cell_ranger"),
  hvgNumber = 2000,
  useFeatureSubset = NULL,
  featureDisplay = metadata(inSCE)$featureDisplay
)
```
```
setTopHVG(
  inSCE,
  method = c("vst", "dispersion", "mean.var.plot", "modelGeneVar", "seurat", "seurat_v3", "cell_ranger"),
  hvgNumber = 2000,
  featureSubsetName = NULL,
  genes = NULL,
  genesBy = NULL,
  altExp = FALSE
)
```

### Arguments
- **inSCE**: Input SingleCellExperiment object
- **method**: Specify which method to use for variable gene extraction from Seurat "vst", "mean.var.plot", "dispersion" or Scran "modelGeneVar" or Scanpy "seurat", "cell_ranger", "seurat_v3". Default "vst"
getTopHVG

hvgNumber Specify the number of top variable genes to extract.

useFeatureSubset Get the feature names in the HVG list set by setTopHVG. Will ignore method and hvgNumber if not NULL. Default NULL.

featureDisplay A character string for the rowData variable name to indicate what type of feature ID should be displayed. If set by setSCTKDisplayRow, will by default use it. If NULL, will use rownames(inSCE).

featureSubsetName A character string for the rowData variable name to store a logical index of selected features. Default NULL, will be determined basing on other parameters.

genes A customized character vector of gene list to be set as a rowData variable. Will ignore method and hvgNumber if set. Default NULL.

genesBy If setting customized genes, where should it be found in rowData? Leave NULL for matching rownames. Default NULL.

altExp TRUE for also creating a subset inSCE object with the selected HVGs and store this subset in the altExps slot, named by hvgListName. Default FALSE.

Value

getTopHVG A character vector of the top hvgNumber variable feature names

setTopHVG The input inSCE object with the logical vector of HVG selection updated in rowData, and related parameter updated in metadata. If altExp is TRUE, an altExp is also added

Author(s)

Irzam Sarfraz, Yichen Wang

See Also

runFeatureSelection, runSeuratFindHVG, runModelGeneVar, plotTopHVG

Examples

data("scExample", package = "singleCellTK")
sce <- runSeuratFindHVG(sce)
hvgs <- getTopHVG(sce, hvgNumber = 10)
sce <- setTopHVG(sce, method = "vst", hvgNumber = 5)
Description

SCTK allows user to access all TSCAN related results with "getTSCANResults". See details.

Usage

getTSCANResults(x, analysisName = NULL, pathName = NULL)

## S4 method for signature 'SingleCellExperiment'
getTSCANResults(x, analysisName = NULL, pathName = NULL)

getTSCANResults(x, analysisName, pathName = NULL) <- value

## S4 replacement method for signature 'SingleCellExperiment'
getTSCANResults(x, analysisName, pathName = NULL) <- value

listTSCANResults(x)

## S4 method for signature 'SingleCellExperiment'
listTSCANResults(x)

listTSCANTerminalNodes(x)

## S4 method for signature 'SingleCellExperiment'
listTSCANTerminalNodes(x)

Arguments

x Input SingleCellExperiment object.
analysisName Algorithm name implemented, should be one of "Pseudotime", "DEG", or "ClusterDEAnalysis".
pathName Sub folder name within the analysisName. See details.
value Value to be stored within the pathName or analysisName

Details

When analysisName = "Pseudotime", returns the list result from runTSCAN, including the MST structure.

When analysisName = "DEG", returns the list result from runTSCANDEG, including DataFrames containing genes that increase/decrease along each the pseudotime paths. pathName indicates the path index, the available options of which can be listed by listTSCANTerminalNodes.

When analysisName = "ClusterDEAnalysis", returns the list result from runTSCANClusterDEAnalysis. Here pathName needs to match with the useCluster argument when running the algorithm.
importAlevin

Value

Get or set TSCAN results

Examples

data("mouseBrainSubsetSCE", package = "singleCellTK")
mouseBrainSubsetSCE <- runTSCAN(inSCE = mouseBrainSubsetSCE,
   useReducedDim = "PCA_logcounts")
results <- getTSCANResults(mouseBrainSubsetSCE, "Pseudotime")

importAlevin

Construct SCE object from Salmon-Alevin output

Description

Construct SCE object from Salmon-Alevin output

Usage

importAlevin(
   alevinDir = NULL,
   sampleName = "sample",
   delayedArray = FALSE,
   class = c("Matrix", "matrix"),
   rowNamesDedup = TRUE
)

Arguments

alevinDir Character. The output directory of salmon-Alevin pipeline. It should contain subfolder named 'alevin', which contains the count data which is stored in 'quants_mat.gz'. Default NULL.
sampleName Character. A user-defined sample name for the sample to be imported. The 'sampleName' will be appended to the begining of cell barcodes. Default is 'sample'.
delayedArray Boolean. Whether to read the expression matrix asDelayedArrayobject or not. Default FALSE.
class Character. The class of the expression matrix stored in the SCE object. Can be one of "Matrix" (as returned byreadMMfunction), or "matrix" (as returned bymatrixfunction). Default "Matrix".
rowNamesDedup Boolean. Whether to deduplicate rownames. Default TRUE.

Value

A SingleCellExperimentobject containing the count matrix, the feature annotations, and the cell annotation (which includes QC metrics stored in 'featureDump.txt').
importAnnData  

Create a SingleCellExperiment Object from Python AnnData.h5ad files

**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**

```r
importAnnData(
  sampleDirs = NULL,
  sampleNames = NULL,
  delayedArray = FALSE,
  class = c("Matrix", "matrix"),
  rowNamesDedup = TRUE
)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>sampleDirs</code></td>
<td>Folder containing the .h5ad file. Can be one of -</td>
</tr>
<tr>
<td></td>
<td>• Default current working directory.</td>
</tr>
<tr>
<td></td>
<td>• Full path to the directory containing the .h5ad file. E.g <code>sampleDirs</code></td>
</tr>
<tr>
<td></td>
<td>= '/path/to/sample'</td>
</tr>
<tr>
<td></td>
<td>• A vector of folder paths for the samples to import. E.g. <code>sampleDirs</code></td>
</tr>
<tr>
<td></td>
<td>= c('/path/to/sample1', '/path/to/sample2', '/path/to/sample3')</td>
</tr>
<tr>
<td></td>
<td>importAnnData will return a single SCE object containing all the samples</td>
</tr>
<tr>
<td></td>
<td>with the sample name appended to each colname in colData.</td>
</tr>
<tr>
<td><code>sampleNames</code></td>
<td>The prefix/name of the .h5ad file without the .h5ad extension e.g. if 'sam-</td>
</tr>
<tr>
<td></td>
<td>ple.h5ad' is the filename, pass <code>sampleNames</code> = 'sample'. Can be one of-</td>
</tr>
<tr>
<td></td>
<td>• Default sample.</td>
</tr>
<tr>
<td></td>
<td>• A vector of samples to import. Length of vector must be equal to length</td>
</tr>
<tr>
<td></td>
<td>of <code>sampleDirs</code> vector E.g. <code>sampleDirs</code> = c('sample1', 'sample2', 'sample3')</td>
</tr>
<tr>
<td></td>
<td>importAnnData will return a single SCE object containing all the samples</td>
</tr>
<tr>
<td></td>
<td>with the sample name appended to each colname in colData.</td>
</tr>
<tr>
<td><code>delayedArray</code></td>
<td>Boolean. Whether to read the expression matrix as DelayedArray object. De-</td>
</tr>
<tr>
<td></td>
<td>fault FALSE.</td>
</tr>
<tr>
<td><code>class</code></td>
<td>Character. The class of the expression matrix stored in the SCE object. C-</td>
</tr>
<tr>
<td></td>
<td>an be one of &quot;Matrix&quot; (as returned by readMM function), or &quot;matrix&quot; (as</td>
</tr>
<tr>
<td></td>
<td>returned by matrix function). Default &quot;Matrix&quot;.</td>
</tr>
<tr>
<td><code>rowNamesDedup</code></td>
<td>Boolean. Whether to deduplicate rownames. Default TRUE.</td>
</tr>
</tbody>
</table>
importBUStools

Construct SCE object from BUStools output

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

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

importBUStools

Construct SCE object from BUStools output

Description

Read the barcodes, features (genes), and matrix from BUS tools output. Import them as one SingleCellExperiment object. Note the cells in the output files for BUS tools 0.39.4 are not filtered.

Usage

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

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 FALSE.
rowNamesDedup  Boolean. Whether to deduplicate rownames. 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 | \n# bustools sort -T tmp/ -t 4 -p - | \n# bustools count -o genecount/genes \
# -g ./transcripts_to_genes.txt \n
importCellRanger

# -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

Construct SCE object from Cell Ranger output

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 = FALSE,
  rowNamesDedup = TRUE
)

importCellRangerV2(
  cellRangerDirs = NULL,
  sampleDirs = NULL,
  sampleNames = NULL,
  dataTypeV2 = c("filtered", "raw"),
  class = c("Matrix", "matrix"),
  delayedArray = FALSE,
  reference = NULL,
  cellRangerOutsV2 = NULL,
  rowNamesDedupV2 = NULL
)

importCellRangerV3(
importCellRanger

cellRangerDirs = NULL,
sampleDirs = NULL,
sampleNames = NULL,
dataType = c("filtered", "raw"),
class = c("Matrix", "matrix"),
delayedArray = FALSE,
rowNamesDedup = 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(cellRangerDirs,
list.dirs, recursive = 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. Supercedes dataType. If NULL, dataType will
be used to determine Cell Ranger output directory. If not NULL, dataType will
be ingored and cellRangerOuts specifies the paths. Must have length 1 or the
same length as length(unlist(sampleDirs)) if sampleDirs is not NULL. Other-
wise, make sure the length and order match the output of unlist(lapply(cellRangerDirs,
list.dirs, recursive = FALSE)). Reference genome names might need to be
appended for CellRanger version below 3.0.0 if reads were mapped to mul-
tiple 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/"

dataType  Character. The type of data to import. Can be one of "filtered" (which is
equivalent to cellRangerOuts = "outs/filtered_feature_bc_matrix/" or
cellRangerOuts = "outs/filtered_gene_bc_matrix/") or "raw" (which is
equivalent to cellRangerOuts = "outs/raw_feature_bc_matrix/" or cellRangerOuts
= "outs/raw_gene_bc_matrix/"). Default "filtered" which imports the counts
for filtered cell barcodes only.
importCellRanger

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 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)).

featuresFileNames
Character vector. Filenames for the feature annotation files. They are usually named features.tsv.gz or genes.tsv. Must have length 1 or 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(cellRangerDirs, list.dirs, recursive = FALSE)).

barcodesFileNames
Character vector. Filename for the cell barcode list files. They are usually named barcodes.tsv.gz or barcodes.tsv. Must have length 1 or 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(cellRangerDirs, list.dirs, recursive = FALSE)).

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", gzipped must have length 1 or 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(cellRangerDirs, list.dirs, recursive = 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 FALSE.

rowNamesDedup
Boolean. Whether to deduplicate rownames. Default TRUE.

dataTypeV2
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 cellRangerOuts is specified, dataTypeV2 and reference will be ignored.

reference
Character vector. The reference genome names. Default NULL. If not NULL, it must have 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.
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".

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. importCellRangerV3 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"
)

c sce <- importCellRangerV3(
  cellRangerDirs = system.file("extdata/", package = "singleCellTK"),
  sampleDirs = "hgmm_1k_v3_20x20",
  sampleNames = "hgmm1kv3",
  dataType = "filtered"
)
importCellRangerV3Sample

Usage

importCellRangerV2Sample(
    dataDir = NULL,
    sampleName = NULL,
    class = c("Matrix", "matrix"),
    delayedArray = FALSE,
    rowNamesDedup = 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 FALSE.
rowNamesDedup Boolean. Whether to deduplicate rownames. 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

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

Value

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

Examples

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

---

**importDropEst**

Create a `SingleCellExperiment` Object from DropEst output

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.
importDropEst

Usage
importDropEst(
  sampleDirs = NULL,
  dataType = c("filtered", "raw"),
  rdsFileName = "cell.counts",
  sampleNames = NULL,
  delayedArray = FALSE,
  class = c("Matrix", "matrix"),
  rowNamesDedup = TRUE
)

Arguments

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

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_per_cb" 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/E XAMPLES.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 TENxPBMCData packages. See 'Details' for a list of available datasets.

Usage

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

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.
rowNamesDedup  Boolean. Whether to deduplicate rownames. Default TRUE.

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).


"NestorowaHSCData" Retrieved with NestorowaHSCData. Returns a dataset of 1920 mouse haematopoietic stem cells from Nestorowa et al. 2015

"pbmc3k" Retrieved with TENxPBMCData. 2,700 peripheral blood mononuclear cells (PBMCs) from 10X Genomics.

"pbmc4k" Retrieved with TENxPBMCData. 4,340 peripheral blood mononuclear cells (PBMCs) from 10X Genomics.

"pbmc6k" Retrieved with TENxPBMCData. 5,419 peripheral blood mononuclear cells (PBMCs) from 10X Genomics.
"pbmc8k" Retrieved with TENxPBMCData. 8,381 peripheral blood mononuclear cells (PBMCs) from 10X Genomics.

"pbmc33k" Retrieved with TENxPBMCData. 33,148 peripheral blood mononuclear cells (PBMCs) from 10X Genomics.

"pbmc68k" Retrieved with TENxPBMCData. 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("pbmc33k")
Arguments

assayFile The path to a file in .mtx, .txt, .csv, .tab, or .tsv format.
annotFile The path to a text file that contains columns of annotation information for each cell in the assayFile. This file should have the same number of rows as there are columns in the assayFile. If multiple samples are represented in the dataset, this should be denoted by a column called 'sample' within the annotFile.
featureFile 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 assayFile. This is optional.
assayName The name of the assay that you are uploading. The default is "counts".
inputDataFrames If TRUE, assayFile, annotFile and featureFile should be data.frames object (or its inheritance) 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 asDelayedArray object or not. Default FALSE.
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. This should match to the colnames of the assayFile. 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. This should match to the rownames of the assayFile. 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 are TRUE or FALSE.
rowNamesDedup Boolean. Whether to deduplicate rownames. Default TRUE.

Details

Creates a SingleCellExperiment object from a counts file in various formats, and files of cell and feature annotation.

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",
  noMatchError = TRUE
)

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 identifies 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".

noMatchError Boolean. Show an error if a collection does not have any matching features. Default TRUE.

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)
gs1 <- GSEABase::GeneSet(setName = "geneset1",
geneIds = rownames(sce)[seq(10)])
gs2 <- GSEABase::GeneSet(setName = "geneset2",
geneIds = rownames(sce)[seq(11,20)])
gsc <- GSEABase::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",
noMatchError = TRUE
)
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".

noMatchError Boolean. Show an error if a collection does not have any matching features. Default TRUE.

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.
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", noMatchError = TRUE)

Arguments

inSCE Input SingleCellExperiment object.
geneSetList 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 be become the gene set names in the GeneSetCollection object.
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 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

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)
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.

Examples

data(scExample)

# Generate gene sets from 'rownames'
gs1 <- rownames(sce)[seq(10)]
gs2 <- rownames(sce)[seq(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,  # Input SingleCellExperiment object.
  categoryIDs,  # Character vector containing the MSigDB gene set ids. The column ID in the table returned by getMSigDBTable() shows the list of possible gene set IDs that can be obtained.
  species = "Homo sapiens",  # Character. Species available can be found using the function msigdbr_show_species. Default "Homo sapiens".
  mapping = c("gene_symbol", "human_gene_symbol", "entrez_gene"),  # Character. One of "gene_symbol", "human_gene_symbol", or "entrez_gene".
  by = "rownames",  # 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 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. See featureIndex for more information. Default "rownames".
  verbose = TRUE,  # Boolean. Whether to display progress. Default TRUE.
  noMatchError = TRUE  # Boolean. Show an error if a collection does not have any matching features. Default TRUE.
)

Arguments

inSCE: Input SingleCellExperiment object.

categoryIDs: Character vector containing the MSigDB gene set ids. The column ID in the table returned by getMSigDBTable() shows the list of possible gene set IDs that can be obtained.

species: Character. Species available can be found using the function msigdbr_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 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. See featureIndex for more information. Default "rownames".

verbose: Boolean. Whether to display progress. Default TRUE.

noMatchError: Boolean. Show an error if a collection does not have any matching features. Default TRUE.
**Details**

The gene identifiers in gene sets from MSigDB will be retrieved using the `msigdbr` 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**

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

---

**importMitoGeneSet**

*Import mitochondrial gene sets*

**Description**

Imports mitochondrial gene sets 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**

```r
importMitoGeneSet(
    inSCE,
    reference = "human",
    id = "ensembl",
    by = "rownames",
    collectionName = "mito",
    noMatchError = TRUE
)
```
importMitoGeneSet

Arguments

inSCE Input SingleCellExperiment object.
reference Character. Species available are "human" and "mouse".
id Types of gene id. Now it supports "symbol", "entrez", "ensembl" and "ensemblTranscriptID".
by Character. Describes the location within inSCE where the gene identifiers in the mitochondrial gene sets 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 identifies will be mapped to that column in the rowData of inSCE. See featureIndex for more information. Default "rownames".
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 "mito".
noMatchError Boolean. Show an error if a collection does not have any matching features. Default TRUE.

Details

The gene identifiers of mitochondrial genes will be loaded with "data(AllMito)". Currently, it supports human and mouse references. Also, it supports entrez ID, gene symbol, ensemble ID and ensemble transcript ID. 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)

Rui Hong

See Also

importGeneSetsFromList for importing from lists, importGeneSetsFromGMT for importing from GMT files, and GeneSetCollection objects.

Examples

data(scExample)
sce <- importMitoGeneSet(inSCE = sce,
  reference = "human",
  id = "ensembl",
  collectionName = "human_mito",
  by = "rownames")
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**
- **allImportEntries**: object containing the sources and parameters of all the samples being imported (from the UI)
- **delayedArray**: Boolean. Whether to read the expression matrix as DelayedArray object or not. Default FALSE.

**Value**
A list of SingleCellExperiment object containing the droplet or cell data or both, depending on the dataType that users provided.

importOptimus  Construct SCE object from Optimus output

**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 = FALSE,
               rowNamesDedup = TRUE)
importOptimus

Arguments

**OptimusDirs**  
A vector of root directories of Optimus output files. The paths should be something like this: /PATH/TO/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_counts_col_index.npy.

**rowIndexLocation**  
Character. The intermediate path to the feature (gene) index file. Default call-MergeCountFiles/sparse_counts_row_index.npy.

**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.gz). 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 FALSE.

**rowNamesDedup**  
Boolean. Whether to deduplicate rownames. Default TRUE.

Value

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

Examples

```r
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**

```r
importSEQC(
  seqcDirs = NULL,
  samples = NULL,
  prefix = NULL,
  gzipped = FALSE,
  class = c("Matrix", "matrix"),
  delayedArray = FALSE,
  cbNotFirstCol = TRUE,
  feNotFirstCol = TRUE,
  combinedSample = TRUE,
  rowNamesDedup = 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 `samples`.
- **gzipped** Boolean. TRUE if the seqc output files (sparse_counts_barcode.csv, sparse_counts_genes.csv, and sparse_molecule_counts.mtx) were gzip compressed. FALSE otherwise. Default seqc outputs are not gzipped. Default 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 FALSE.
- **cbNotFirstCol** Boolean. TRUE if first column of sparse_counts_barcode.csv is row index and it will be removed. FALSE the first column will be kept.
- **feNotFirstCol** Boolean. TRUE if first column of sparse_counts_genes.csv is row index and it will be removed. FALSE the first column will be kept.
combinedSample  Boolean. If TRUE, importSEQC returns a SingleCellExperiment object containing the combined count matrix, feature annotations and the cell annotations. If FALSE, importSEQC returns a list containing multiple SingleCellExperiment objects. Each SingleCellExperiment contains count matrix, feature annotations and cell annotations for each sample.

rowNamesDedup  Boolean. Whether to deduplicate rownames. Only applied if combinedSample is TRUE or only one seqcDirs specified. Default TRUE.

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

```r
# 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  

Construct SCE object from STARsolo outputs

Description

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

Usage

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

Arguments

STARsoloDirs  A vector of root directories of STARsolo output files. The paths should be something like this: /PATH/TO/prefixSolo.out. For example: ./Solo.out. Each sample 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 STARsoloDirs.
STARsoloOuts  Character. The intermediate folder to filtered or raw cell barcode, feature, and matrix files for each of samples. Default "Gene". It can be either Gene or GeneFull as the main folder from which data needs to be imported.
matrixFileNames  Filenames for the Market Exchange Format (MEX) sparse matrix file (.mtx file). Must have length 1 or the same length as samples.
featuresFileNames  Filenames for the feature annotation file. 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 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 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 object or not. Default FALSE.
rowNamesDedup  Boolean. Whether to deduplicate rownames. Default TRUE.
iterateSimulations

Returns significance data from a snapshot.

Description

Returns significance data from a snapshot.

Usage

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

listSampleSummaryStatsTables

Lists the table of SCTK QC outputs stored within the metadata.

Description

Returns a character vector of the tables within the metadata slot of the SingleCellExperiment object.

Usage

listSampleSummaryStatsTables(inSCE, ...)

## S4 method for signature 'SingleCellExperiment'
listSampleSummaryStatsTables(inSCE, ...)

Arguments

inSCE         Input SingleCellExperiment object with saved table within the metadata data. Required.
...           Other arguments passed to the function.
mergeSCEColData

**Value**

A character vector. Contains a list of summary tables within the SingleCellExperiment object.

**Examples**

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

mergeSCEColData  
**Merging colData from two singleCellExperiment objects**

**Description**

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

**Usage**

```r
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**

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

List of mitochondrial genes of multiple reference

**Description**

A list of gene set that contains mitochondrial genes of multiple reference (hg38, hg19, mm10 and mm9). It contains multiple types of gene identifier: gene symbol, entrez ID, ensemble ID and ensemble transcript ID. It's used for the function 'importMitoGeneSet'.

**Usage**

```r
data("MitoGenes")
```

**Format**

A list

**Value**

List of mitochondrial genes of multiple reference

**Examples**

```r
data("MitoGenes")
```

---

**mouseBrainSubsetSCE**

Example Single Cell RNA-Seq data in SingleCellExperiment Object, GSE60361 subset

**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**

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

**Format**

SingleCellExperiment

**Value**

A subset of 30 cells from a single cell RNA-Seq experiment
Source

DOI: 10.1126/science.aaa1934

Examples

data("mouseBrainSubsetSCE")

msigdb_table

MSigDB gene set Category table

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

data("msigdb_table")

Format

A data.frame.

Value

A table of gene set categories

Examples

data("msigdb_table")

plotBarcodeRankDropsResults

Plots for runBarcodeRankDrops outputs.

Description

A wrapper function which visualizes outputs from the runBarcodeRankDrops function stored in the metadata slot of the SingleCellExperiment object.
Usage

```r
plotBarcodeRankDropsResults(
  inSCE,
  sample = NULL,
  defaultTheme = TRUE,
  dotSize = 0.5,
  titleSize = 18,
  axisSize = 15,
  axisLabelSize = 18,
  legendSize = 15
)
```

Arguments

- `inSCE`: Input `SingleCellExperiment` object with saved dimension reduction components or a variable with saved results from `runBarcodeRankDrops`. Required.
- `sample`: Character vector or colData variable name. 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 `0.5`.
- `axisSize`: Size of x/y-axis ticks. Default `15`.

Value

list of .ggplot objects

Examples

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

Description

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

Plots for `runBarcodeRankDrops` outputs.
plotBarcodeRankScatter

Usage

plotBarcodeRankScatter(
    inSCE, sample = NULL,
    defaultTheme = TRUE,
    dotSize = 0.1,
    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 or colData variable name. 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 0.1.
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 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".
sampleRelHeights If there are multiple samples and combining by "all", the relative heights for each plot. Default 1.
sampleRelWidths If there are multiple samples and combining by "all", the relative widths for each plot. Default 1.

Value

a ggplot object of the scatter plot.
plotBatchCorrCompare

See Also

plotBarcodeRankDropsResults, runBarcodeRankDrops

Examples

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

plotBatchCorrCompare

Plot comparison of batch corrected result against original assay

Description

Plot comparison of batch corrected result against original assay

Usage

plotBatchCorrCompare(
  inSCE,
  corrMat,
  batch = NULL,
  condition = NULL,
  origAssay = NULL,
  origLogged = NULL,
  method = NULL,
  matType = NULL
)

Arguments

inSCE SingleCellExperiment inherited object.
corrMat A single character indicating the name of the corrected matrix.
batch A single character. The name of batch annotation column in colData(inSCE).
condition A single character. The name of an additional covariate annotation column in colData(inSCE).
origAssay A single character indicating what the original assay used for batch correction is.
origLogged Logical scalar indicating whether origAssay is log-normalized.
method A single character indicating the name of the batch correction method. Only used for the titles of plots.
matType A single character indicating the type of the batch correction result matrix, choose from "assay", "altExp", "reducedDim".
plotBatchVariance

Details

Four plots will be combined. Two of them are violin/box-plots for percent variance explained by the batch variation, and optionally the covariate, for original and corrected. The other two are UMAPs of the original assay and the correction result matrix. If SCTK batch correction methods are performed in advance, this function will automatically detect necessary input. Otherwise, users can also customize the input. Future improvement might include solution to reduce redundant UMAP calculation.

Value

An object of class "gtable", combining four ggplots.

Author(s)

Yichen Wang

Examples

data("sceBatches")
logcounts(sceBatches) <- log1p(counts(sceBatches))
sceBatches <- runLimmaBC(sceBatches)
plotBatchCorrCompare(sceBatches, "LIMMA", condition = "cell_type")

plotBatchVariance  Plot the percent of the variation that is explained by batch and condition in the data

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
)
plotBcdsResults

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

data('sceBatches', package = 'singleCellTK')
plotBatchVariance(sceBatches,
useAssay="counts",
batch="batch",
condition = "cell_type")

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.
plotBcdsResults

Usage

plotBcdsResults(
inSCE, sample = NULL, shape = NULL, groupBy = NULL, combinePlot = "all", violin = TRUE, boxplot = FALSE, dots = TRUE, reducedDimName = "UMAP", xlab = NULL, ylab = NULL, dim1 = NULL, dim2 = NULL, bin = NULL, binLabel = NULL, defaultTheme = TRUE, dotSize = 0.5, summary = "median", summaryTextSize = 3, transparency = 1, baseSize = 15, titleSize = NULL, axisLabelSize = NULL, axisSize = NULL, legendSize = NULL, legendTitleSize = NULL, relHeights = 1, relWidths = c(1, 1, 1), plotNCols = NULL, plotNRows = NULL, labelSamples = TRUE, samplePerColumn = TRUE, sampleRelHeights = 1, sampleRelWidths = 1)

Arguments

inSCE Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results from runBcds. Required.
sample Character vector or colData variable name. Indicates which sample each cell belongs to. Default NULL.
shape If provided, add shapes based on the value. Default NULL.
groupBy Groupings for each numeric value. A user may input a vector equal length to the number of the samples in inSCE, 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 inSCE. Default "UMAP".

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. Similar to dim1. Default is NULL.

bin Numeric vector. If single value, will divide the numeric values into bin groups. If more than one value, will bin numeric values using values as a cut point. Default NULL.

binLabel Character vector. Labels for the bins created by bin. Default NULL.

defaultTheme Removes grid in plot and sets axis title size to 10 when TRUE. Default TRUE.

dotSize Size of dots. Default 0.5.

summary Adds a summary statistic, as well as a crossbar to the violin plot. Options are "mean" or "median". Default NULL.

summaryTextSize The text size of the summary statistic displayed above the violin plot. Default 3.

transparency Transparency of the dots, values will be 0-1. Default 1.

baseSize The base font size for all text. Default 12. Can be overwritten by titleSize, axisSize, and axisLabelSize, legendSize, legendTitleSize.

titleSize Size of title of plot. Default NULL.

axisLabelSize Size of x/y-axis labels. Default NULL.

axisSize Size of x/y-axis ticks. Default NULL.

legendSize size of legend. Default NULL.

legendTitleSize size of legend title. Default NULL.

relHeights Relative heights of plots when combine is set. Default 1.

relWidths Relative widths of plots when combine is set. Default c(1, 1, 1).

plotNCols Number of columns when plots are combined in a grid. Default NULL.

plotNRows Number of rows when plots are combined in a grid. Default NULL.

labelSamples Will label sample name in title of plot if TRUE. Default TRUE.

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. Default 1.

sampleRelWidths
If there are multiple samples and combining by "all", the relative widths for each plot. Default 1.

Value
list of .ggplot objects

See Also
runBcds

Examples

data(scExample, package="singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
sce <- runQuickUMAP(sce)
sce <- runBcds(sce)
plotBcdsResults(inSCE=sce, reducedDimName="UMAP")
**plotClusterAbundance**

*Plot the differential Abundance*

**Description**

Plot the differential Abundance

**Usage**

```r
plotClusterAbundance(inSCE, cluster, variable, combinePlot = c("all", "none"))
```

**Arguments**

- **inSCE** A `SingleCellExperiment` object.
- **cluster** A single character, specifying the name to store the cluster label in `colData`.
- **variable** A single character, specifying the name to store the phenotype labels in `colData`.
- **combinePlot** Must be either "all" or "none". "all" will combine all plots into a single `ggplot` object. Default "all".

---

**plotClusterAbundance Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>inSCE</code></td>
<td>The single cell experiment to use.</td>
</tr>
<tr>
<td><code>useAssay</code></td>
<td>The assay to use.</td>
</tr>
<tr>
<td><code>featureNames</code></td>
<td>A string or vector of strings with each gene to aggregate.</td>
</tr>
<tr>
<td><code>displayName</code></td>
<td>A string that is the name of the column used for genes.</td>
</tr>
<tr>
<td><code>groupNames</code></td>
<td>The name of a colData entry that can be used as groupNames.</td>
</tr>
<tr>
<td><code>title</code></td>
<td>The title of the bubble plot</td>
</tr>
<tr>
<td><code>xlab</code></td>
<td>The x-axis label</td>
</tr>
<tr>
<td><code>ylab</code></td>
<td>The y-axis label</td>
</tr>
<tr>
<td><code>colorLow</code></td>
<td>The color to be used for lowest value of mean expression</td>
</tr>
<tr>
<td><code>colorHigh</code></td>
<td>The color to be used for highest value of mean expression</td>
</tr>
<tr>
<td><code>scale</code></td>
<td>Option to scale the data. Default: <code>FALSE</code>. Selected assay will not be scaled.</td>
</tr>
</tbody>
</table>

**Value**

A `ggplot` of the bubble plot.

**Examples**

```r
data("scExample")
plotBubble(inSCE=sce, useAssay="counts", featureNames=c("B2M", "MALAT1"),
displayName="feature_name", groupNames="type", title="cell type test",
xlab="gene", ylab="cluster", colorLow="white", colorHigh="blue")
```

---

---

---

---

---

---

---
plotCxdsResults

Details

This function will visualize the differential abundance in two given variables, by making bar plots that presents the cell counting and fraction in different cases.

Value

When combinePlot = "none", a list with 4 ggplot objects; when combinePlot = "all", a single ggplot object with for subplots.

Examples

data("mouseBrainSubsetSCE", package = "singleCellTK")
plotClusterAbundance(inSCE = mouseBrainSubsetSCE,
cluster = "tissue",
variable = "level1class")

Description

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

Usage

plotCxdsResults(
inSCE,
sample = NULL,
shape = NULL,
groupBy = NULL,
combinePlot = "all",
violin = TRUE,
boxplot = FALSE,
dots = TRUE,
reducedDimName = "UMAP",
xlab = NULL,
ylab = NULL,
dim1 = NULL,
dim2 = NULL,
bin = NULL,
binLabel = NULL,
defaultTheme = TRUE,
dotSize = 0.5,
summary = "median",
summaryTextSize = 3,
transparency = 1,
Arguments

inSCE Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results from runCxds. Required.
sample Character vector or colData variable name. Indicates which sample each cell belongs to. Default NULL.
shape If provided, add shapes based on the value. Default NULL.
groupBy Groupings for each numeric value. A user may input a vector equal length to the number of the samples in inSCE, 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 inSCE. Default "UMAP".
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. Similar to dim1. Default is NULL.
bin Numeric vector. If single value, will divide the numeric values into bin groups. If more than one value, will bin numeric values using values as a cut point. Default NULL.
binLabel Character vector. Labels for the bins created by bin. Default NULL.
defaultTheme Removes grid in plot and sets axis title size to 10 when TRUE. Default TRUE.
plotCxdsResults

dotSize       Size of dots. Default 0.5.
summary      Adds a summary statistic, as well as a crossbar to the violin plot. Options are "mean" or "median". Default NULL.
summaryTextSize     The text size of the summary statistic displayed above the violin plot. Default 3.
transparency Transparency of the dots, values will be 0-1. Default 1.
baseSize     The base font size for all text. Default 12. Can be overwritten by titleSize, axisSize, and axisLabelSize, legendSize, legendTitleSize.
titleSize    Size of title of plot. Default NULL.
axisLabelSize Size of x/y-axis labels. Default NULL.
axisSize     Size of x/y-axis ticks. Default NULL.
legendSize   size of legend. Default NULL.
legendTitleSize size of legend title. Default NULL.
relHeights   Relative heights of plots when combine is set. Default 1.
relWidths    Relative widths of plots when combine is set. Default c(1, 1, 1).
plotNCols    Number of columns when plots are combined in a grid. Default NULL.
plotNRows    Number of rows when plots are combined in a grid. Default NULL.
labelSamples Will label sample name in title of plot if TRUE. Default TRUE.
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. Default 1.
sampleRelWidths If there are multiple samples and combining by "all", the relative widths for each plot. Default 1.

Value
    list of .ggplot objects

See Also
    runCxds

Examples
    data(scExample, package="singleCellTK")
    sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
    sce <- runQuickUMAP(sce)
    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,
  bgResult = FALSE,
  shape = NULL,
  groupBy = NULL,
  combinePlot = "all",
  violin = TRUE,
  boxplot = FALSE,
  dots = TRUE,
  reducedDimName = "UMAP",
  xlab = NULL,
  ylab = NULL,
  dim1 = NULL,
  dim2 = NULL,
  bin = NULL,
  binLabel = NULL,
  defaultTheme = TRUE,
  dotSize = 0.5,
  summary = "median",
  summaryTextSize = 3,
  transparency = 1,
  baseSize = 15,
  titleSize = NULL,
  axisLabelSize = NULL,
  axisSize = NULL,
  legendSize = NULL,
  legendTitleSize = NULL,
  relHeights = 1,
  relWidths = c(1, 1, 1),
  plotNCols = NULL,
  plotNRows = NULL,
  labelSamples = TRUE,
  labelClusters = TRUE,
  clusterLabelSize = 3.5,
  samplePerColumn = TRUE,
  sampleRelHeights = 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.

**bgResult**  
Boolean. If TRUE, will plot decontX results generated with raw/droplet matrix. Default FALSE.

**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. Default = "UMAP"

**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 0.5.

**summary**  
Adds a summary statistic, as well as a crossbar to the violin plot. Options are "mean" or "median". Default NULL.

**summaryTextSize**  
The text size of the summary statistic displayed above the violin plot. Default 3.

**transparency**  
Transparency of the dots, values will be 0-1. Default 1.
`plotDecontXResults`  

```r
plotDecontXResults
```

**baseSize**  
The base font size for all text. Default 12. Can be overwritten by `titleSize`, `axisSize`, and `axisLabelSize`, `legendSize`, `legendTitleSize`.

**titleSize**  
Size of title of plot. Default NULL.

**axisLabelSize**  
Size of x/y-axis labels. Default NULL.

**axisSize**  
Size of x/y-axis ticks. Default NULL.

**legendSize**  
Size of legend. Default NULL.

**legendTitleSize**  
Size of legend title. Default NULL.

**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.

**labelSamples**  
Will label sample name in title of plot if TRUE. Default TRUE.

**labelClusters**  
Logical. Whether the cluster labels are plotted. Default FALSE.

**clusterLabelSize**  
Numeric. Determines the size of cluster label when `labelClusters` is set to TRUE. Default 3.5.

**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**

```r
data(scExample, package="singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
sce <- runDecontX(sce)
plotDecontXResults(inSCE=sce, reducedDimName="decontX_UMAP")
```
plotDEGHeatmap

Heatmap visualization of DEG result

Description

Heatmap visualization of DEG result

Usage

plotDEGHeatmap(
inSCE, useResult, onlyPos = FALSE, log2fcThreshold = 0.25, fdrThreshold = 0.05, minGroup1MeanExp = NULL, maxGroup2MeanExp = NULL, minGroup1ExprPerc = NULL, maxGroup2ExprPerc = NULL, useAssay = NULL, doLog = FALSE, featureAnnotations = NULL, cellAnnotations = NULL, featureAnnotationColor = NULL, cellAnnotationColor = NULL, rowDataName = NULL, colDataName = NULL, colSplitBy = "condition", rowSplitBy = "regulation", rowLabel = S4Vectors::metadata(inSCE)$featureDisplay, title = paste0("DE Analysis: ", useResult), ...
)

Arguments

inSCE SingleCellExperiment inherited object.
useResult character. A string specifying the analysisName used when running a differential expression analysis function.
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 0.25.
fdrThreshold numeric. Only plot DEGs with FDR value smaller than this value. Default 0.05.
`plotDEGHeatmap`

- `minGroup1MeanExp` numeric. Only plot DEGs with mean expression in group1 greater than this value. Default NULL.
- `maxGroup2MeanExp` numeric. Only plot DEGs with mean expression in group2 less than this value. Default NULL.
- `minGroup1ExprPerc` numeric. Only plot DEGs expressed in greater than this fraction of cells in group1. Default NULL.
- `maxGroup2ExprPerc` numeric. Only plot DEGs expressed in less than this fraction of cells in group2. Default NULL.
- `useAssay` character. A string specifying an assay of expression value to plot. By default the assay used for `runMAST()` will be used. Default NULL.
- `doLog` Logical scalar. Whether to do log(assay + 1) transformation on the assay used for the analysis. Default FALSE.
- `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".
- `rowLabel` FALSE for not displaying; a variable in `rowData` to display feature identifiers stored there; if have run `setSCTKDisplayRow`, display the specified feature name; TRUE for the rownames of inSCE; NULL for auto-display rownames when the number of filtered feature is less than 60. Default looks for `setSCTKDisplayRow` information.
- `title` character. Main title of the heatmap. Default "DE Analysis: <useResult>".
- ... Other arguments passed to `plotSCEHeatmap`
Details

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.

Value

A ggplot object

Author(s)

Yichen Wang

Examples

data("sceBatches")
logcounts(sceBatches) <- log1p(counts(sceBatches))
sce.w <- subsetSCECols(sceBatches, colData = "batch == 'w'")
sce.w <- runWilcox(sce.w, class = "cell_type", classGroup1 = "alpha",
    groupName1 = "w.alpha", groupName2 = "w.beta",
    analysisName = "w.aVSb")
plotDEGHeatmap(sce.w, "w.aVSb")

plotDEGRegression

Create linear regression plot to show the expression the of top DEGs

Description

Create linear regression plot to show the expression the of top DEGs

Usage

plotDEGRegression(
inSCE,
useResult,
threshP = FALSE,
labelBy = NULL,
nrow = 6,
col = 6,
defaultTheme = TRUE,
isLogged = TRUE,
check_sanity = TRUE
)
**Arguments**

- `inSCE` *SingleCellExperiment* inherited object.
- `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 when performing DE analysis. Default `FALSE`.
- `labelBy` A single character for a column of `rowData` in `SCE` as where to search for the labeling text. Default `NULL`.
- `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`.

**Details**

Any of the differential expression analysis method from SCTK should be performed prior to using this function.

**Value**

A `ggplot` object of linear regression

**Examples**

```r
data("sceBatches")
logcounts(sceBatches) <- log1p(counts(sceBatches))
sce.w <- subsetSCECols(sceBatches, colData = "batch == 'w'")
sce.w <- runWilcox(sce.w, class = "cell_type", classGroup1 = "alpha",
                   groupName1 = "w.alpha", groupName2 = "w.beta",
                   analysisName = "w.aVSb")
plotDEGRegression(sce.w, "w.aVSb")
```

---

**plotDEGViolin**

Generate violin plot to show the expression of top DEGs

**Description**

Generate violin plot to show the expression of top DEGs
Usage

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

Arguments

inSCE SingleCellExperiment inherited object.
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.
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.

Details

Any of the differential expression analysis method from SCTK should be performed prior to using this function.

Value

A ggplot object of violin plot

Examples

data("sceBatches")
logcounts(sceBatches) <- log1p(counts(sceBatches))
sce.w <- subsetSCEcols(sceBatches, colData = "batch == 'w'")
sce.w <- runWilcox(sce.w, class = "cell_type", classGroup1 = "alpha", 
  groupName1 = "w.alpha", groupName2 = "w.beta", 
  analysisName = "w.aVSb")
plotDEGViolin(sce.w, "w.aVSb")
plotDEGVolcano

Generate volcano plot for DEGs

Description
Generate volcano plot for DEGs

Usage
plotDEGVolcano(
  inSCE,
  useResult,
  labelTopN = 10,
  log2fcThreshold = 0.25,
  fdrThreshold = 0.05,
  featureDisplay = S4Vectors::metadata(inSCE)$featureDisplay
)

Arguments
inSCE SingleCellExperiment inherited object.
useResult character. A string specifying the analysisName used when running a differential expression analysis function.
labelTopN Integer, label this number of top DEGs that pass the filters. FALSE for not labeling. Default 10.
log2fcThreshold numeric. Label genes with the absolute values of log2FC greater than this value as regulated. Default 0.25.
fdrThreshold numeric. Label genes with FDR value less than this value as regulated. Default 0.05.
featureDisplay A character string to indicate a variable in rowData(inSCE) for feature labeling. NULL for using rownames. Default metadata(inSCE)$featureDisplay (see setSCTKDisplayRow)

Details
Any of the differential expression analysis method from SCTK should be performed prior to using this function to generate volcano plots.

Value
A ggplot object of volcano plot

See Also
runDEAnalysis, plotDEGHeatmap
Examples

data("sceBatches")
sceBatches <- scaterlogNormCounts(sceBatches, "logcounts")
sce.w <- subsetSCECols(sceBatches, colData = "batch == 'w'")
sce.w <- runWilcox(sce.w, class = "cell_type", classGroup1 = "alpha",
                   groupName1 = "w.alpha", groupName2 = "w.beta",
                   analysisName = "w.aVSb")
plotDEGVolcano(sce.w, "w.aVSb")

plotDimRed

Plot dimensionality reduction from computed metrics including PCA, ICA, tSNE and UMAP

Description

Plot dimensionality reduction from computed metrics including PCA, ICA, tSNE and UMAP

Usage

plotDimRed(
  inSCE,
  useReduction,
  showLegend = FALSE,
  xDim = 1,
  yDim = 2,
  xAxisLabel = NULL,
  yAxisLabel = NULL
)

Arguments

inSCE Input SCE object
useReduction Reduction to plot
showLegend If legends should be plotted or not
xDim Numeric value indicating the dimension to use for X-axis. Default is 1 (refers to PC1).
yDim Numeric value indicating the dimension to use for Y-axis. Default is 2 (refers to PC2).
xAxisLabel Specify the label for x-axis. Default is NULL which will specify the label as 'x'.
yAxisLabel Specify the label for y-axis. Default is NULL which will specify the label as 'y'.

Value

plot object
Examples

```r
data("mouseBrainSubsetSCE", package = "singleCellTK")
plotDimRed(mouseBrainSubsetSCE, "PCA_logcounts")
```

**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**

```r
plotDoubletFinderResults(
inSCE,
sample = NULL,
shape = NULL,
groupBy = NULL,
combinePlot = "all",
violin = TRUE,
boxplot = FALSE,
dots = TRUE,
reducedDimName = "UMAP",
xlab = NULL,
ylab = NULL,
dim1 = NULL,
dim2 = NULL,
bin = NULL,
binLabel = NULL,
defaultTheme = TRUE,
dotSize = 0.5,
summary = "median",
summaryTextSize = 3,
transparency = 1,
baseSize = 15,
titleSize = NULL,
axisLabelSize = NULL,
axisSize = NULL,
legendSize = NULL,
legendTitleSize = NULL,
relHeights = 1,
relWidths = c(1, 1, 1),
plotNCols = NULL,
plotNRows = NULL,
labelSamples = TRUE,
)```
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 or colData variable name. Indicates which sample each cell belongs to. Default NULL.

shape If provided, add shapes based on the value. Default NULL.

groupBy Groupings for each numeric value. A user may input a vector equal length to the number of the samples in inSCE, 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 inSCE. Default "UMAP".

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. Similar to dim1. Default is NULL.

bin Numeric vector. If single value, will divide the numeric values into bin groups. If more than one value, will bin numeric values using values as a cut point. Default NULL.

binLabel Character vector. Labels for the bins created by bin. Default NULL.

defaultTheme Removes grid in plot and sets axis title size to 10 when TRUE. Default TRUE.

dotSize Size of dots. Default 0.5.

summary Adds a summary statistic, as well as a crossbar to the violin plot. Options are "mean" or "median". Default NULL.

summaryTextSize The text size of the summary statistic displayed above the violin plot. Default 3.

transparency Transparency of the dots, values will be 0-1. Default 1.

baseSize The base font size for all text. Default 12. Can be overwritten by titleSize, axisSize, and axisLabelSize, legendSize, legendTitleSize.

titleSize Size of title of plot. Default NULL.
**plotEmptyDropsResults**

Plots for `runEmptyDrops` outputs.

**Description**

A wrapper function which visualizes outputs from the `runEmptyDrops` function stored in the `colData` slot of the `SingleCellExperiment` object.

**Value**

list of `.ggplot` objects

**See Also**

`runDoubletFinder`

**Examples**

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

plotEmptyDropsResults(
  inSCE,
  sample = NULL,
  combinePlot = "all",
  fdrCutoff = 0.01,
  defaultTheme = TRUE,
  dotSize = 0.5,
  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 runEmptyDrops. Required.
sample Character vector or colData variable name. Indicates which sample each cell belongs to. Default NULL.
combinePlot Must be either "all", "sample", or object, "none". "all" will combine all plots into a single .ggplot 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 0.5.
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. Default 1.
relWidths Relative widths of plots when combine is set. Default 1.
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. Default 1.

sampleRelWidths
If there are multiple samples and combining by "all", the relative widths for each plot. Default 1.

Value
list of .ggplot objects

See Also
runEmptyDrops, plotEmptyDropsScatter

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 col-
Data slot of the SingleCellExperiment object via scatter plots.

Usage

plotEmptyDropsScatter(
inSCE,
sample = NULL,
fdrCutoff = 0.01,
defaultTheme = TRUE,
dotSize = 0.1,
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

inSCE Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results from runEmptyDrops. Required.
sample Character vector or colData variable name. Indicates which sample each cell belongs to. Default NULL.
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 0.1.
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.
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 "all".
relHeights Relative heights of plots when combine is set. Default 1.
relWidths Relative widths of plots when combine is set. Default 1.
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. Default 1.
sampleRelWidths If there are multiple samples and combining by "all", the relative widths for each plot. Default 1.

Value

a ggplot object of the scatter plot.
See Also

runEmptyDrops, plotEmptyDropsResults

Examples

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

plotFindMarkerHeatmap  Plot a heatmap to visualize the result of runFindMarker

Description

This function will first reads the result saved in metadata slot, named by "findMarker" and generated by runFindMarker. 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

plotFindMarkerHeatmap(
  inSCE,
  orderBy = "size",
  log2fcThreshold = 1,
  fdrThreshold = 0.05,
  minClustExprPerc = 0.7,
  maxCtrlExprPerc = 0.4,
  minMeanExpr = 1,
  topN = 10,
  decreasing = TRUE,
  rowLabel = TRUE,
  rowDataName = NULL,
  colDataName = NULL,
  featureAnnotations = NULL,
  cellAnnotations = NULL,
  featureAnnotationColor = NULL,
  cellAnnotationColor = NULL,
  colSplitBy = NULL,
  rowSplitBy = "marker",
  rowDend = FALSE,
  colDend = FALSE,
  title = "Top Marker Heatmap",
)
plotFindMarkerHeatmap

Arguments

inSCE SingleCellExperiment inherited object.

orderBy 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".

log2fcThreshold Only use DEGs with the absolute values of log2FC larger than this value. Default 1

fdrThreshold Only use DEGs with FDR value smaller than this value. Default 0.05

minClustExprPerc A numeric scalar. The minimum cutoff of the percentage of cells in the cluster of interests that expressed the marker gene. Default 0.7.

maxCtrlExprPerc A numeric scalar. The maximum cutoff of the percentage of cells out of the cluster (control group) that expressed the marker gene. Default 0.4.

minMeanExpr A numeric scalar. The minimum cutoff of the mean expression value of the marker in the cluster of interests. Default 1.

topN 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.
rowLabel  TRUE for displaying rownames of inSCE, a rowData variable to use other feature identifiers, or a vector for customized row labels. Use FALSE for not displaying. 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 runFindMarker 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.
rowDend  Whether to display row dendrogram. Default FALSE.
colDend  Whether to display column dendrogram. Default FALSE.
title  Text of the title, at the top of the heatmap. Default "Top Marker Heatmap".
...  Other arguments passed to plotSCEHeatmap.

Value
A Heatmap object

Author(s)
Yichen Wang

See Also
runFindMarker, getFindMarkerTopTable
Examples

data("sceBatches")
logcounts(sceBatches) <- log1p(counts(sceBatches))
sce.w <- subsetSCECols(sceBatches, colData = "batch == 'w'")
sce.w <- runFindMarker(sce.w, method = "wilcox", cluster = "cell_type")
plotFindMarkerHeatmap(sce.w)

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",  
doPlot = TRUE,  
isLogged = TRUE,  
check_sanity = TRUE
)

Arguments

inSCE SingleCellExperiment object
useAssay character, default "logcounts"
doPlot Logical scalar. Whether to directly plot in the plotting area. If FALSE, will return a graphical object which can be visualized with grid.draw(). 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

Plot the thresholding onto the plotting region if plot == TRUE or a graphical object if plot == FALSE.

Examples

data("mouseBrainSubsetSCE")
plotMASTThresholdGenes(mouseBrainSubsetSCE)
**plotPathway**

*Generate violin plots for pathway analysis results*

**Description**

Generate violin plots for pathway analysis results

**Usage**

```r
plotPathway(
  inSCE,                         # Input SingleCellExperiment object. With runGSVA() or runVAM() applied in advance.
  resultName,                   # A single character of the name of a score matrix, which should be found in getPathwayResultNames(inSCE).
  geneset,                      # A single character specifying the geneset of interest. Should be found in the geneSetCollection used for performing the analysis.
  groupBy = NULL,              # Either a single character specifying a column of colData(inSCE) or a vector of equal length as the number of cells. Default NULL.
  boxplot = FALSE,             # Boolean, Whether to add a boxplot. Default FALSE.
  violin = TRUE,               # Boolean, Whether to add a violin plot. Default TRUE.
  dots = TRUE,                 # Boolean, If TRUE, will plot dots for each violin plot. Default TRUE.
  summary = "median",         # Adds a summary statistic, as well as a crossbar to the violin plot. Options are "mean" or "median", and NULL for not adding. Default "median".
  axisSize = 10,               # Size of x/y-axis ticks. Default 10.
  axisLabelSize = 10,
  dotSize = 0.5,
  transparency = 1,
  defaultTheme = TRUE,
  gridLine = FALSE,
  title = geneset,
  titleSize = NULL
)
```

**Arguments**

- `inSCE`: Input `SingleCellExperiment` object. With `runGSVA()` or `runVAM()` applied in advance.
- `resultName`: A single character of the name of a score matrix, which should be found in `getPathwayResultNames(inSCE)`. Default `NULL`.
- `geneset`: A single character specifying the geneset of interest. Should be found in the geneSetCollection used for performing the analysis. Default `NULL`.
- `groupBy`: Either a single character specifying a column of `colData(inSCE)` or a vector of equal length as the number of cells. Default `NULL`.
- `boxplot`: Boolean, Whether to add a boxplot. Default `FALSE`.
- `violin`: Boolean, Whether to add a violin plot. Default `TRUE`.
- `dots`: Boolean, If `TRUE`, will plot dots for each violin plot. Default `TRUE`.
- `summary`: Adds a summary statistic, as well as a crossbar to the violin plot. Options are "mean" or "median", and `NULL` for not adding. Default "median".
axisLabelSize  Size of x/y-axis labels. Default 10.
dotSize  Size of dots. Default 0.5.
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.
dotSize  Size of dots. Default 0.5.
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.
title  Title of plot. Default using geneset.
titleSize  Size of the title of the plot. Default 15.

Details
runGSVA() or runVAM() should be applied in advance of using this function. Users can group the data by specifying groupby.

Value
A ggplot object for the violin plot

Examples

data("scExample", package = "singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
sce <- scaterlogNormCounts(sce, assayName = "logcounts")
gs1 <- rownames(sce)[seq(10)]
gs2 <- rownames(sce)[seq(11,20)]
gs <- list("geneset1" = gs1, "geneset2" = gs2)
sce <- importGeneSetsFromList(inSCE = sce, geneSetList = gs,
  by = "rownames")
sce <- runVAM(inSCE = sce, geneSetCollectionName = "GeneSetCollection",
  useAssay = "logcounts")
plotPathway(sce, "VAM_GeneSetCollection_CDF", "geneset1")

plotPCA  Plot PCA run data from its components.

Description
Plot PCA run data from its components.

Usage

plotPCA(  
inSCE,
  colorBy = NULL,
  shape = NULL,
  pcX = "PC1",
  pcY = "PC2",)
plotRunPerCellQCResults

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,
plotRunPerCellQCResults

dots = TRUE,
dotSize = 0.5,
summary = "median",
summaryTextSize = 3,
baseSize = 15,
axisSize = NULL,
axisLabelSize = NULL,
transparency = 1,
defaultTheme = TRUE,
titleSize = NULL,
relHeights = 1,
relWidths = 1,
labelSamples = TRUE,
plotNCols = NULL,
plotNRows = 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 or colData variable name. Indicates which sample each cell belongs to. Default NULL.
groupBy Groupings for each numeric value. Users may input a vector equal length to the number of the samples in inSCE, 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 0.5.
summary Adds a summary statistic, as well as a crossbar to the violin plot. Options are "mean" or "median". Default "median".
summaryTextSize The text size of the summary statistic displayed above the violin plot. Default 3.
baseSize The base font size for all text. Default 15. Can be overwritten by titleSize, axisSize, and axisLabelSize.
axisSize Size of x/y-axis ticks. Default NULL.
axisLabelSize Size of x/y-axis labels. Default NULL.
transparency Transparency of the dots, values will be 0-1. Default 1.
plotScanpyDotPlot

defaultTheme: Removes grid in plot and sets axis title size to 10 when TRUE. Default TRUE.
titleSize: Size of title of plot. Default NULL.
relHeights: Relative heights of plots when combine is set. Default 1.
relWidths: Relative widths of plots when combine is set. Default 1.
labelSamples: Will label sample name in title of plot if TRUE. Default TRUE.
plotNCols: Number of columns when plots are combined in a grid. Default NULL.
plotNRows: Number of rows when plots are combined in a grid. Default NULL.
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. Default 1.
sampleRelWidths
  If there are multiple samples and combining by "all", the relative widths for
  each plot. Default 1.

Value
  list of .ggplot objects

See Also
  runPerCellQC

Examples

data(scExample, package = "singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
sce <- runPerCellQC(sce)
plotRunPerCellQCResults(inSCE = sce)

plotScanpyDotPlot

data(scExample, package = "singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
sce <- runPerCellQC(sce)
plotRunPerCellQCResults(inSCE = sce)
Usage

plotScanpyDotPlot(
  inSCE,
  useAssay = NULL,
  features,
  groupBy,
  standardScale = NULL,
  title = "",
  vmin = NULL,
  vmax = NULL,
  colorBarTitle = "Mean expression in group"
)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inSCE</td>
<td>Input SingleCellExperiment object.</td>
</tr>
<tr>
<td>useAssay</td>
<td>Assay to use for plotting. By default it will use counts assay.</td>
</tr>
<tr>
<td>features</td>
<td>Genes to plot. Sometimes is useful to pass a specific list of var names (e.g. genes). The var_names could be a dictionary or a list.</td>
</tr>
<tr>
<td>groupBy</td>
<td>The key of the observation grouping to consider.</td>
</tr>
<tr>
<td>standardScale</td>
<td>Whether or not to standardize the given dimension between 0 and 1, meaning for each variable or group, subtract the minimum and divide each by its maximum. Default NULL means that it doesn’t perform any scaling.</td>
</tr>
<tr>
<td>title</td>
<td>Provide title for the figure.</td>
</tr>
<tr>
<td>vmin</td>
<td>The value representing the lower limit of the color scale. Values smaller than vmin are plotted with the same color as vmin. Default NULL</td>
</tr>
<tr>
<td>vmax</td>
<td>The value representing the upper limit of the color scale. Values larger than vmax are plotted with the same color as vmax. Default NULL</td>
</tr>
<tr>
<td>colorBarTitle</td>
<td>Title for the color bar.</td>
</tr>
</tbody>
</table>

Value

plot object

Examples

data(scExample, package = "singleCellTK")
## Not run:
sce <- runScanpyNormalizeData(sce, useAssay = "counts")
sce <- runScanpyFindHVG(sce, useAssay = "scanpyNormData", method = "seurat")
sce <- runScanpyScaleData(sce, useAssay = "scanpyNormData")
sce <- runScanpyPCA(sce, useAssay = "scanpyScaledData")
sce <- runScanpyFindClusters(sce, useReducedDim = "scanpyPCA")
sce <- runScanpyUMAP(sce, useReducedDim = "scanpyPCA")
markers <- c("MALAT1", "RPS27", "CST3")
plotScanpyDotPlot(sce, features = markers, groupBy = 'Scanpy_louvain_1')

## End(Not run)
plotScanpyEmbedding

Description

plotScanpyEmbedding

Usage

plotScanpyEmbedding(
  inSCE,
  reducedDimName,
  useAssay = NULL,
  color = NULL,
  legend = "right margin",
  title = ""
)

Arguments

inSCE: Input SingleCellExperiment object.
reducedDimName: Name of reducedDims object containing embeddings. Eg. scanpyUMAP.
useAssay: Specify name of assay to use. Default is NULL, which will use scaled assay by default.
color: Keys for annotations of observations/cells or variables/genes.
legend: Location of legend, either 'on data', 'right margin' or a valid keyword for the loc parameter of Legend.
title: Provide title for panels either as string or list of strings

Value

plot object

Examples

data(scExample, package = "singleCellTK")
## Not run:
sce <- runScanpyNormalizeData(sce, useAssay = "counts")
sce <- runScanpyFindHVG(sce, useAssay = "scanpyNormData", method = "seurat")
sce <- runScanpyScaleData(sce, useAssay = "scanpyNormData")
sce <- runScanpyPCA(sce, useAssay = "scanpyScaledData")
sce <- runScanpyFindClusters(sce, useReducedDim = "scanpyPCA")
sce <- runScanpyUMAP(sce, useReducedDim = "scanpyPCA")
plotScanpyEmbedding(sce, reducedDimName = "scanpyUMAP", color = 'Scanpy_louvain_1')
## End(Not run)
plotScanpyHeatmap

Description

plotScanpyHeatmap

Usage

plotScanpyHeatmap(
  inSCE,
  useAssay = NULL,
  features,
  groupBy,
  standardScale = "var",
  vmin = NULL,
  vmax = NULL
)

Arguments

inSCE Input SingleCellExperiment object.
useAssay Assay to use for plotting. By default it will use counts assay.
features Genes to plot. Sometimes is useful to pass a specific list of var names (e.g.
genomes). The var_names could be a dictionary or a list.
groupBy The key of the observation grouping to consider.
standardScale Whether or not to standardize the given dimension between 0 and 1, meaning for
each variable or group, subtract the minimum and divide each by its maximum.
Default NULL means that it doesn’t perform any scaling.
vmin The value representing the lower limit of the color scale. Values smaller than
vmin are plotted with the same color as vmin. Default NULL
vmax The value representing the upper limit of the color scale. Values larger than
vmax are plotted with the same color as vmax. Default NULL

Value

plot object

Examples

data(scExample, package = "singleCellTK")
## Not run:
sce <- runScanpyNormalizeData(sce, useAssay = "counts")
sce <- runScanpyFindHVG(sce, useAssay = "scanpyNormData", method = "seurat")
sce <- runScanpyScaleData(sce, useAssay = "scanpyNormData")
sce <- runScanpyPCA(sce, useAssay = "scanpyScaledData")
sce <- runScanpyFindClusters(sce, useReducedDim = "scanpyPCA")
sce <- runScanpyUMAP(sce, useReducedDim = "scanpyPCA")
markers <- c("MALAT1", "RPS27", "CST3")
plotScanpyHeatmap(sce, features = markers, groupBy = 'Scanpy_louvain_1')

## End(Not run)

---

plotScanpyHVG

**Description**

plotScanpyHVG

**Usage**

plotScanpyHVG(inSCE, log = FALSE)

**Arguments**

- **inSCE**: Input SingleCellExperiment object.
- **log**: Plot on logarithmic axes. Default FALSE.

**Value**

plot object

**Examples**

data(scExample, package = "singleCellTK")

## Not run:
sce <- runScanpyNormalizeData(sce, useAssay = "counts")
sce <- runScanpyFindHVG(sce, useAssay = "scanpyNormData", method = "seurat")
plotScanpyHVG(sce)

## End(Not run)
plotScanpyMarkerGenes

Description
plotScanpyMarkerGenes

Usage
plotScanpyMarkerGenes(
  inSCE, 
  groups = NULL,
  nGenes = 10,
  nCols = 4,
  sharey = FALSE
)

Arguments
inSCE Input SingleCellExperiment object.
groups The groups for which to show the gene ranking. Default NULL means that all groups will be considered.
nGenes Number of genes to show. Default 10
nCols Number of panels shown per row. Default 4
sharey Controls if the y-axis of each panels should be shared. Default FALSE allows each panel to have its own y-axis range.

Value
plot object

Examples
data(scExample, package = "singleCellTK")
## Not run:
sce <- runScanpyNormalizeData(sce, useAssay = "counts")
sce <- runScanpyFindHVG(sce, useAssay = "scanpyNormData", method = "seurat")
sce <- runScanpyScaleData(sce, useAssay = "scanpyNormData")
sce <- runScanpyPCA(sce, useAssay = "scanpyScaledData")
sce <- runScanpyFindClusters(sce, useReducedDim = "scanpyPCA")
sce <- runScanpyFindMarkers(sce, colDataName = "Scanpy_louvain_1")
plotScanpyMarkerGenes(sce, groups = '0')

## End(Not run)
plotScanpyMarkerGenesDotPlot

Description

plotScanpyMarkerGenesDotPlot

Usage

plotScanpyMarkerGenesDotPlot(
  inSCE,
  groups = NULL,
  nGenes = 10,
  groupBy,
  log2fcThreshold = NULL,
  parameters = "logfoldchanges",
  standardScale = NULL,
  features = NULL,
  title = "",
  vmin = NULL,
  vmax = NULL,
  colorBarTitle = "log fold change"
)

Arguments

inSCE Input SingleCellExperiment object.
groups The groups for which to show the gene ranking. Default NULL means that all groups will be considered.
nGenes Number of genes to show. Default 10
groupBy The key of the observation grouping to consider. By default, the groupby is chosen from the rank genes groups parameter.
log2fcThreshold Only output DEGs with the absolute values of log2FC larger than this value. Default NULL.
parameters The options for marker genes results to plot are: ‘scores’, ‘logfoldchanges’, ‘pvals’, ‘pvals_adj’, ‘log10_pvals’, ‘log10_pvals_adj’. If NULL provided then it uses mean gene value to plot.
standardScale Whether or not to standardize the given dimension between 0 and 1, meaning for each variable or group, subtract the minimum and divide each by its maximum. Default NULL means that it doesn’t perform any scaling.
features Genes to plot. Sometimes is useful to pass a specific list of var names (e.g. genes) to check their fold changes or p-values, instead of the top/bottom genes. The gene names could be a dictionary or a list. Default NULL.
### plotScanpyMarkerGenesHeatmap

#### Description

plotScanpyMarkerGenesHeatmap

#### Usage

```r
plotScanpyMarkerGenesHeatmap(
  inSCE,  
  groups = NULL,  
  groupBy,  
  nGenes = 10,  
  features = NULL,  
  log2fcThreshold = NULL
)
```

#### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>title</td>
<td>Provide title for the figure.</td>
</tr>
<tr>
<td>vmin</td>
<td>The value representing the lower limit of the color scale. Values smaller than vmin are plotted with the same color as vmin. Default NULL</td>
</tr>
<tr>
<td>vmax</td>
<td>The value representing the upper limit of the color scale. Values larger than vmax are plotted with the same color as vmax. Default NULL</td>
</tr>
<tr>
<td>colorBarTitle</td>
<td>Title for the color bar.</td>
</tr>
</tbody>
</table>

#### Examples

```r
data(scExample, package = "singleCellTK")
## Not run:
sce <- runScanpyNormalizeData(sce, useAssay = "counts")
sce <- runScanpyFindHVG(sce, useAssay = "scanpyNormData", method = "seurat")
sce <- runScanpyScaleData(sce, useAssay = "scanpyNormData")
sce <- runScanpyPCA(sce, useAssay = "scanpyScaledData")
sce <- runScanpyFindClusters(sce, useReducedDim = "scanpyPCA")
sce <- runScanpyFindMarkers(sce, colDataName = "Scanpy_louvain_1")
plotScanpyMarkerGenesDotPlot(sce, groupBy = 'Scanpy_louvain_1')

## End(Not run)
```
Arguments

inSCE: Input SingleCellExperiment object.
groups: The groups for which to show the gene ranking. Default NULL means that all groups will be considered.
groupBy: The key of the observation grouping to consider. By default, the groupby is chosen from the rank genes groups parameter.
nGenes: Number of genes to show. Default 10
features: Genes to plot. Sometimes is useful to pass a specific list of var names (e.g., genes). The var_names could be a dictionary or a list.
l2fcThreshold: Only output DEGs with the absolute values of log2FC larger than this value. Default NULL.

Value

plot object

Examples

data(scExample, package = "singleCellTK")

## Not run:
sce <- runScanpyNormalizeData(sce, useAssay = "counts")
sce <- runScanpyFindHVG(sce, useAssay = "scanpyNormData", method = "seurat")
sce <- runScanpyScaleData(sce, useAssay = "scanpyNormData")
sce <- runScanpyPCA(sce, useAssay = "scanpyScaledData")
sce <- runScanpyFindClusters(sce, useReducedDim = "scanpyPCA")
sce <- runScanpyFindMarkers(sce, colDataName = "Scanpy_louvain_1")
plotScanpyMarkerGenesHeatmap(sce, groupBy = "Scanpy_louvain_1")

## End(Not run)
plotScanpyMarkerGenesMatrixPlot

log2fcThreshold = NULL,
parameters = "logfoldchanges",
standardScale = "var",
features = NULL,
title = "",
vmin = NULL,
vmax = NULL,
colorBarTitle = "log fold change"
)

Arguments

inSCE Input SingleCellExperiment object.
groups The groups for which to show the gene ranking. Default NULL means that all groups will be considered.
nGenes Number of genes to show. Default 10
groupBy The key of the observation grouping to consider. By default, the groupby is chosen from the rank genes groups parameter.
log2fcThreshold Only output DEGs with the absolute values of log2FC larger than this value. Default NULL.
parameters The options for marker genes results to plot are: ‘scores’, ‘logfoldchanges’, ‘pvals’, ‘pvals_adj’, ‘log10_pvals’, ‘log10_pvals_adj’. If NULL provided then it uses mean gene value to plot.
standardScale Whether or not to standardize the given dimension between 0 and 1, meaning for each variable or group, subtract the minimum and divide each by its maximum. Default NULL means that it doesn’t perform any scaling.
features Genes to plot. Sometimes is useful to pass a specific list of var names (e.g. genes) to check their fold changes or p-values, instead of the top/bottom genes. The var_names could be a dictionary or a list. Default NULL

title Provide title for the figure.
vmin The value representing the lower limit of the color scale. Values smaller than vmin are plotted with the same color as vmin. Default NULL
vmax The value representing the upper limit of the color scale. Values larger than vmax are plotted with the same color as vmax. Default NULL
colorBarTitle Title for the color bar.

Value
plot object

Examples

data(scExample, package = "singleCellTK")
## Not run:
sce <- runScanpyNormalizeData(sce, useAssay = "counts")
plotScanpyMarkerGenesViolin

Description

plotScanpyMarkerGenesViolin

Usage

plotScanpyMarkerGenesViolin(inSCE, groups = NULL, features = NULL, nGenes = 10)

Arguments

inSCE Input SingleCellExperiment object.
groups The groups for which to show the gene ranking. Default NULL means that all groups will be considered.
features List of genes to plot. Is only useful if interested in a custom gene list
nGenes Number of genes to show. Default 10

Value

plot object

Examples

data(scExample, package = "singleCellTK")
## Not run:
sce <- runScanpyNormalizeData(sce, useAssay = "counts")
sce <- runScanpyFindHVG(sce, useAssay = "scanpyNormData", method = "seurat")
sce <- runScanpyScaleData(sce, useAssay = "scanpyNormData")
sce <- runScanpyPCA(sce, useAssay = "scanpyScaledData")
sce <- runScanpyFindClusters(sce, useReducedDim = "scanpyPCA")
sce <- runScanpyFindMarkers(sce, colDataName = "Scanpy_louvain_1")
plotScanpyMarkerGenesViolin(sce, groups = '0')

## End(Not run)
Description

plotScanpyMatrixPlot

Usage

plotScanpyMatrixPlot(
  inSCE,
  useAssay = NULL,
  features,
  groupBy,
  standardScale = NULL,
  title = "",
  vmin = NULL,
  vmax = NULL,
  colorBarTitle = "Mean expression in group"
)

Arguments

- **inSCE**: Input SingleCellExperiment object.
- **useAssay**: Assay to use for plotting. By default it will use counts assay.
- **features**: Genes to plot. Sometimes is useful to pass a specific list of var names (e.g. genes). The var_names could be a dictionary or a list.
- **groupBy**: The key of the observation grouping to consider.
- **standardScale**: Whether or not to standardize the given dimension between 0 and 1, meaning for each variable or group, subtract the minimum and divide each by its maximum. Default NULL means that it doesn’t perform any scaling.
- **title**: Provide title for the figure.
- **vmin**: The value representing the lower limit of the color scale. Values smaller than vmin are plotted with the same color as vmin. Default NULL.
- **vmax**: The value representing the upper limit of the color scale. Values larger than vmax are plotted with the same color as vmax. Default NULL.
- **colorBarTitle**: Title for the color bar.

Value

plot object
Examples

```r
data(scExample, package = "singleCellTK")
## Not run:
sce <- runScanpyNormalizeData(sce, useAssay = "counts")
sce <- runScanpyFindHVG(sce, useAssay = "scanpyNormData", method = "seurat")
sce <- runScanpyScaleData(sce, useAssay = "scanpyNormData")
sce <- runScanpyPCA(sce, useAssay = "scanpyScaledData")
sce <- runScanpyFindClusters(sce, useReducedDim = "scanpyPCA")
sce <- runScanpyUMAP(sce, useReducedDim = "scanpyPCA")
markers <- c("MALAT1", "RPS27", "CST3")
plotScanpyMatrixPlot(sce, features = markers, groupBy = 'Scanpy_louvain_1')
## End(Not run)
```

Description

plotScanpyPCA

Usage

```r
plotScanpyPCA(
inSCE, 
reducedDimName = "scanpyPCA", 
color = NULL, 
title = "", 
legend = "right margin"
)
```

Arguments

- `inSCE`: Input SingleCellExperiment object.
- `reducedDimName`: Name of new reducedDims object containing Scanpy PCA.
- `color`: Keys for annotations of observations/cells or variables/genes.
- `title`: Provide title for panels either as string or list of strings
- `legend`: Location of legend, either 'on data', 'right margin' or a valid keyword for the loc parameter of Legend.

Value

plot object
plotScanpyPCAGeneRanking

**Examples**

data(scExample, package = "singleCellTK")
## Not run:
sce <- runScanpyNormalizeData(sce, useAssay = "counts")
sce <- runScanpyFindHVG(sce, useAssay = "scanpyNormData", method = "seurat")
sce <- runScanpyScaleData(sce, useAssay = "scanpyNormData")
sce <- runScanpyPCA(sce, useAssay = "scanpyScaledData")
plotScanpyPCA(sce)

## End(Not run)

**Description**

plotScanpyPCAGeneRanking

**Usage**

plotScanpyPCAGeneRanking(inSCE, PC_comp = "1,2,3", includeLowest = TRUE)

**Arguments**

- **inSCE**: Input SingleCellExperiment object.
- **PC_comp**: For example, '1,2,3' means [1, 2, 3], first, second, third principal component.
- **includeLowest**: Whether to show the variables with both highest and lowest loadings. Default TRUE

**Value**

plot object

**Examples**

data(scExample, package = "singleCellTK")
## Not run:
sce <- runScanpyNormalizeData(sce, useAssay = "counts")
sce <- runScanpyFindHVG(sce, useAssay = "scanpyNormData", method = "seurat")
sce <- runScanpyScaleData(sce, useAssay = "scanpyNormData")
sce <- runScanpyPCA(sce, useAssay = "scanpyScaledData")
plotScanpyPCAGeneRanking(sce)

## End(Not run)
**Description**

plotScanpyPCA Variance

**Usage**

```r
plotScanpyPCAVariance(inSCE, nPCs = 50, log = FALSE)
```

**Arguments**

- `inSCE`: Input SingleCellExperiment object.
- `nPCs`: Number of PCs to show. Default 50.
- `log`: Plot on logarithmic scale. Default FALSE

**Value**

plot object

**Examples**

```r
data(scExample, package = "singleCellTK")
## Not run:
sce <- runScanpyNormalizeData(sce, useAssay = "counts")
sce <- runScanpyFindHVG(sce, useAssay = "scanpyNormData", method = "seurat")
sce <- runScanpyScaleData(sce, useAssay = "scanpyNormData")
sce <- runScanpyPCA(sce, useAssay = "scanpyScaledData")
plotScanpyPCAVariance(sce)

## End(Not run)
```

---

**Description**

plotScanpyViolin

**Usage**

```r
plotScanpyViolin
```

**Value**

plot object

**Examples**

```r
data(scExample, package = "singleCellTK")
## Not run:
sce <- runScanpyNormalizeData(sce, useAssay = "counts")
sce <- runScanpyFindHVG(sce, useAssay = "scanpyNormData", method = "seurat")
sce <- runScanpyScaleData(sce, useAssay = "scanpyNormData")
sce <- runScanpyPCA(sce, useAssay = "scanpyScaledData")
plotScanpyViolin

## End(Not run)
```
Usage

plotScanpyViolin(
inSCE,
useAssay = NULL,
features,
groupBy,
xlabel = "",
ylabel = NULL
)

Arguments

inSCE_Input SingleCellExperiment object.
useAssay_Assay to use for plotting. By default it will use counts assay.
features_Genes to plot. Sometimes is useful to pass a specific list of var names (e.g. genes). The var_names could be a dictionary or a list.
groupBy_The key of the observation grouping to consider.
xlabel_Label of the x axis. Defaults to groupBy.
ylabel_Label of the y axis. If NULL and groupBy is NULL, defaults to 'value'. If NULL and groupBy is not NULL, defaults to features.

Value

plot object

Examples

data(scExample, package = "singleCellTK")
## Not run:
sce <- runScanpyNormalizeData(sce, useAssay = "counts")
sce <- runScanpyFindHVG(sce, useAssay = "scanpyNormData", method = "seurat")
sce <- runScanpyScaleData(sce, useAssay = "scanpyNormData")
sce <- runScanpyPCA(sce, useAssay = "scanpyScaledData")
sce <- runScanpyFindClusters(sce, useReducedDim = "scanpyPCA")
sce <- runScanpyUMAP(sce, useReducedDim = "scanpyPCA")
markers <- c("MALAT1", "RPS27", "CST3")
plotScanpyViolin(sce, features = markers, groupBy = "Scanpy_louvain_1")

## End(Not run)
**Description**

A wrapper function which visualizes outputs from the `runScDb1Finder` function stored in the colData slot of the `SingleCellExperiment` object via various plots.

**Usage**

```r
plotScDb1FinderResults(
  inSCE,  # Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results from runScDb1Finder. Required.
  sample = NULL,  # Sample name(s) to plot.
  shape = NULL,  # Shape of data points. Can be a vector or a function that returns a vector.
  groupBy = NULL,  # Variable to group data by. Can be a character or a function that returns a vector.
  combinePlot = "all",  # How to combine plots. Can be "all", "bySample", or "byGroup".
  violin = TRUE,  # Whether to use violin plots.
  boxplot = FALSE,  # Whether to use boxplots.
  dots = TRUE,  # Whether to use dot plots.
  reducedDimName = "UMAP",  # Name of the reduced dimension.
  xlab = NULL,  # X-axis label.
  ylab = NULL,  # Y-axis label.
  dim1 = NULL,  # Dimension 1.
  dim2 = NULL,  # Dimension 2.
  bin = NULL,  # Binning for violin plots.
  binLabel = NULL,  # Label for binning.
  defaultTheme = TRUE,  # Whether to use the default theme.
  dotSize = 0.5,  # Size of dots.
  summary = "median",  # Summary statistic for violin plots.
  summaryTextSize = 3,  # Text size for summary statistic.
  transparency = 1,  # Transparency of plots.
  baseSize = 15,  # Base size of text.
  titleSize = NULL,  # Title size.
  axisLabelSize = NULL,  # Axis label size.
  axisSize = NULL,  # Axis size.
  legendSize = NULL,  # Legend size.
  legendTitleSize = NULL,  # Legend title size.
  relHeights = 1,  # Relative heights.
  relWidths = c(1, 1, 1),  # Relative widths.
  plotNCols = NULL,  # Number of columns for plots.
  plotNRows = NULL,  # Number of rows for plots.
  labelSamples = TRUE,  # Whether to label samples.
  samplePerColumn = TRUE,  # Whether to plot samples per column.
  sampleRelHeights = 1,  # Relative heights for samples.
  sampleRelWidths = 1  # Relative widths for samples.
)
```

**Arguments**

- `inSCE` Input `SingleCellExperiment` object with saved dimension reduction components or a variable with saved results from `runScDb1Finder`. Required.
sample: Character vector or colData variable name. Indicates which sample each cell belongs to. Default NULL.
shape: If provided, add shapes based on the value. Default NULL.
groupBy: Groupings for each numeric value. A user may input a vector equal length to the number of the samples in inSCE, 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 inSCE. Default "UMAP".
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. Similar to dim1. Default is NULL.
bin: Numeric vector. If single value, will divide the numeric values into bin groups. If more than one value, will bin numeric values using values as a cut point. Default NULL.
binLabel: Character vector. Labels for the bins created by bin. Default NULL.
defaultTheme: Removes grid in plot and sets axis title size to 10 when TRUE. Default TRUE.
dotSize: Size of dots. Default 0.5.
summary: Adds a summary statistic, as well as a crossbar to the violin plot. Options are "mean" or "median". Default NULL.
summaryTextSize: The text size of the summary statistic displayed above the violin plot. Default 3.
transparency: Transparency of the dots, values will be 0-1. Default 1.
baseSize: The base font size for all text. Default 12. Can be overwritten by titleSize, axisSize, and axisLabelSize, legendSize, legendTitleSize.
titleSize: Size of title of plot. Default NULL.
axisLabelSize: Size of x/y-axis labels. Default NULL.
axisSize: Size of x/y-axis ticks. Default NULL.
legendSize: size of legend. Default NULL.
legendTitleSize: size of legend title. Default NULL.
relHeights: Relative heights of plots when combine is set. Default 1.
relWidths: Relative widths of plots when combine is set. Default c(1, 1, 1).
plotScdsHybridResults

A wrapper function which visualizes outputs from the runCxdsBcdsHybrid 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,
)
plotScdsHybridResults

dots = TRUE,
reducedDimName = "UMAP",
xlab = NULL,
ylab = NULL,
dim1 = NULL,
dim2 = NULL,
bin = NULL,
binLabel = NULL,
defaultTheme = TRUE,
dotSize = 0.5,
summary = "median",
summaryTextSize = 3,
transparency = 1,
baseSize = 15,
titleSize = NULL,
axisLabelSize = NULL,
axisSize = NULL,
legendSize = NULL,
legendTitleSize = NULL,
relHeights = 1,
relWidths = c(1, 1, 1),
plotNCols = NULL,
plotNRows = NULL,
labelSamples = TRUE,
samplePerColumn = TRUE,
sampleRelHeights = 1,
sampleRelWidths = 1
)

Arguments

inSCE Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results from runCxsBcdsHybrid. 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.
plotScdsHybridResults

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 0.5.

summary
Adds a summary statistic, as well as a crossbar to the violin plot. Options are
"mean" or "median". Default NULL.

summaryTextSize
The text size of the summary statistic displayed above the violin plot. Default 3.

transparency
Transparency of the dots, values will be 0-1. Default 1.

baseSize
The base font size for all text. Default 12. Can be overwritten by titleSize,
axisSize, and axisLabelSize, legendSize, legendTitleSize.

titleSize
Size of title of plot. Default NULL.

axisLabelSize
Size of x/y-axis labels. Default NULL.

axisSize
Size of x/y-axis ticks. Default NULL.

legendSize
Size of legend. Default NULL.

legendTitleSize
Size of legend title. Default NULL.

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.

labelSamples
Will label sample name in title of plot if TRUE. Default TRUE.

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.
plotSCEBarAssayData

Value

list of .ggplot objects

Examples

data(scExample, package="singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
sce <- runQuickUMAP(sce)
sce <- runCxdsBcdsHybrid(sce)
plotScdsHybridResults(inSCE=sce, reducedDimName="UMAP")

Description

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

Usage

plotSCEBarAssayData(  
inSCE,  
feature,  
sample = NULL,  
useAssay = "counts",  
featureLocation = NULL,  
featureDisplay = NULL,  
groupBy = NULL,  
xlab = NULL,  
ylab = NULL,  
axisSize = 10,  
axisLabelSize = 10,  
dotSize = 0.1,  
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.

feature Name of feature stored in assay of SingleCellExperiment object.
plotSCEBarAssayData

- `sample` Character vector. Indicates which sample each cell belongs to.
- `useAssay` Indicate which assay to use. Default "counts".
- `featureLocation` Indicates which column name of rowData to query gene.
- `featureDisplay` Indicates which column name of rowData to use to display feature for visualization.
- `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 0.1.
- `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

data("mouseBrainSubsetSCE")
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**

```r
plotSCEBarColData(
  inSCE,  
coldata,  
sample = NULL,  
groupBy = NULL,  
dots = TRUE,  
xlab = NULL,  
ylab = NULL,  
axisSize = 10,  
axisLabelSize = 10,  
dotSize = 0.1,  
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.
- `coldata`  
  colData value that will be plotted.
- `sample`  
  Character vector. Indicates which sample each cell belongs to.
- `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 0.1.
Plot mean feature value in each batch of a SingleCellExperiment object

Description
Plot mean feature value in each batch of a SingleCellExperiment object

Usage

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

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".

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')
plotSCEBatchFeatureMean(sceBatches, useAssay = "counts")

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,
  slotName,
  itemName,
  sample = NULL,
  feature = NULL,
  dimension = NULL,
  groupBy = NULL,
  xlab = NULL,
  ylab = NULL,
  axisSize = 10,
  axisLabelSize = 10,
  defaultTheme = TRUE,
Arguments

inSCE
Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results. Required.

slotName

itemName
Desired vector within the slot used for plotting. Required.

sample
Character vector. Indicates which sample each cell belongs to.

feature
Desired name of feature stored in assay of SingleCellExperiment object. Only used when "assays" slotName is selected. Default NULL.

dimension
Desired dimension stored in the specified reducedDims. Either an integer which indicates the column or a character vector specifies column name. By default, the 1st dimension/column will be used. Only used when "reducedDims" slotName is selected. 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.

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 object of the density plot.
plotSCEDensityAssayData

Density plot of assay data.

Examples

```r
data("mouseBrainSubsetSCE")
plotSCEDensity(
inSCE = mouseBrainSubsetSCE, slotName = "assays",
itemName = "counts", feature = "Apoe", groupBy = "sex"
)
```

Description

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

Usage

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

Arguments

- `inSCE`: Input `SingleCellExperiment` object with saved dimension reduction components or a variable with saved results. Required.
- `feature`: Name of feature stored in assay of SingleCellExperiment object.
- `sample`: Character vector. Indicates which sample each cell belongs to.
- `useAssay`: Character vector. Indicates which assay to use. Default "counts".
- `featureLocation`: Indicates which column name of rowData to query gene.
plotSCEDensityColData

featureDisplay Indicates which column name of rowData to use to display feature for visualization.

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.

cutoff Numeric value. The plot will be annotated with a vertical line if set. 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 density plot of assay data.

Examples

data("mouseBrainSubsetSCE")
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,  
coldata,  
sample = NULL,  
groupBy = NULL,  
xlab = NULL,  
ylab = NULL,  
baseSize = 12,  
axisSize = NULL,  
axisLabelSize = NULL,  
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.
coldata colData value that will be plotted.
sample Character vector. Indicates which sample each cell belongs to.
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.
baseSize The base font size for all text. Default 12. Can be overwritten by titleSize, axisSize, and axisLabelSize, legendSize, legendTitleSize.
axisSize Size of x/y-axis ticks. Default NULL.
axisLabelSize Size of x/y-axis labels. Default NULL.
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.
plotSCEDimReduceColData

Value

a ggplot of the density plot of colData.

Examples

data("mouseBrainSubsetSCE")
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, colorBy, reducedDimName, sample = NULL, groupBy = NULL, conditionClass = NULL, shape = NULL, xlab = NULL, ylab = NULL, baseSize = 12, axisSize = NULL, axisLabelSize = NULL, dim1 = NULL, dim2 = NULL, bin = NULL, binLabel = NULL, dotSize = 0.1, transparency = 1, colorScale = NULL, colorLow = "white", colorMid = "gray", colorHigh = "blue", defaultTheme = TRUE, title = NULL, titleSize = 15,
labelClusters = TRUE,
clusterLabelSize = 3.5,
legendTitle = NULL,
legendTitleSize = NULL,
legendSize = NULL,
combinePlot = "none",
plotLabels = NULL
)

Arguments

inSCE Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results. Required.

colorBy Color by a condition(any column of the annotation data). Required.

reducedDimName Saved dimension reduction matrix name in the SingleCellExperiment object. Required.

sample Character vector. Indicates which sample each cell belongs to.

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.

xlab Character vector. Label for x-axis. Default NULL.

ylab Character vector. Label for y-axis. Default NULL.

baseSize The base font size for all text. Default 12. Can be overwritten by titleSize, axisSize, and axisLabelSize, legendSize, legendTitleSize.

axisSize Size of x/y-axis ticks. Default NULL.

axisLabelSize Size of x/y-axis labels. 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.

dotSize Size of dots. Default 0.1.

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.
clusterLabelSize Numeric. Determines the size of cluster label when 'labelClusters' is set to
TRUE. Default 3.5.
legendTitle title of legend. Default NULL.
legendTitleSize size of legend title. Default 12.
legendSize size of legend. Default NULL. 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

data("mouseBrainSubsetSCE")
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, 
feature, 
reducedDimName, 
sample = NULL, 
featureLocation = NULL, 
featureDisplay = NULL, 
shape = NULL, 
useAssay = "logcounts", 
xlab = NULL, 
ylab = NULL, 
axisSize = 10, 
axisLabelSize = 10, 
dim1 = NULL, 
dim2 = NULL, 
bin = NULL, 
binLabel = NULL, 
dotSize = 0.1, 
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

inSCE Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results. Required.
plotSCEDimReduceFeatures

**feature**  
Name of feature stored in assay of SingleCellExperiment object.

**reducedDimName**  
saved dimension reduction name in the SingleCellExperiment object. Required.

**sample**  
Character vector. Indicates which sample each cell belongs to.

**featureLocation**  
Indicates which column name of rowData to query gene.

**featureDisplay**  
Indicates which column name of rowData to use to display feature for visualization.

**shape**  
add shapes to each condition. Default NULL.

**useAssay**  
Indicate which assay to use. The default is "logcounts"

**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 0.1.

**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.

**legendTitle**  
title of legend. Default NULL.

**legendSize**  
size of legend. Default 10.

**legendTitleSize**  
size of legend title. Default 12.

**groupBy**  
Facet wrap the scatterplot based on value. 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 reduced dimension plot of feature data.

**Examples**

data("mouseBrainSubsetSCE")
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",
useReducedDim = NULL,
doLog = FALSE,
featureIndex = NULL,
cellIndex = NULL,
scale = TRUE,
trim = c(-2, 2),
featureIndexBy = "rownames",
cellIndexBy = "rownames",
rowDataName = NULL,
colDataName = NULL,
aggregateRow = NULL,
aggregateCol = NULL,
featureAnnotations = NULL,
cellAnnotations = NULL,
featureAnnotationColor = NULL,
cellAnnotationColor = NULL,
palette = c("ggplot", "celda", "random"),
Arguments

inSCE  
SingleCellExperiment inherited object.

useAssay  
character. A string indicating the assay name that provides the expression level to plot. Only for plotSCEHeatmap.

useReducedDim  
character. A string indicating the reducedDim name that provides the expression level to plot. Only for plotSCEDimReduceHeatmap.

doLog  
Logical scalar. Whether to do \( \log(\text{assay} + 1) \) transformation on the assay indicated by useAssay. Default FALSE.

featureIndex  
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 featureIndexBy. Default NULL.

cellIndex  
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 cellIndexBy. Default NULL.

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 c(-2, 2)

featureIndexBy  
A single character specifying a column name of rowData(inSCE), or a vector of the same length as nrow(inSCE), where we search for the non-rowname feature indices. Not applicable for plotSCEDimReduceHeatmap. Default "rownames".

cellIndexBy  
A single character specifying a column name of colData(inSCE), or a vector of the same length as ncol(inSCE), where we search for the non-rowname cell indices. Default "rownames".

rowDataName  
character. The column name(s) in rowData that need to be added to the annotation. Not applicable for plotSCEDimReduceHeatmap. Default NULL.

colDataName  
character. The column name(s) in colData that need to be added to the annotation. Default NULL.
aggregateRow Feature variable for aggregating the heatmap by row. Can be a vector or a rowData column name for feature variable. Multiple variables are allowed. Default NULL.

aggregateCol Cell variable for aggregating the heatmap by column. Can be a vector or a colData column name for cell variable. Multiple variables are allowed. 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.

palette Choose from "ggplot", "celda" or "random" to generate unique category colors.

rowSplitBy character. Do semi-heatmap based on the grouping of this(these) annotation(s). Should exist in either rowDataName or names(featureAnnotations). 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 NULL.

rowLabel Use a logical for whether to display all the feature names, a single character to display a column of rowData(inSCE) annotation, a vector of the same length as full/subset nrow(inSCE) to display customized info. Default FALSE.

colLabel Use a logical for whether to display all the cell names, a single character to display a column of colData(inSCE) annotation, a vector of the same length as full/subset ncol(inSCE) to display customized info. Default FALSE.

rowLabelSize A number for the font size of feature names. Default 8

colLabelSize A number for the font size of cell names. Default 8

rowDend Whether to display row dendrogram. Default TRUE.

colDend Whether to display column dendrogram. Default TRUE.

title The main title of the whole plot. Default NULL.

rowTitle The subtitle for the rows. Default "Genes".

colTitle The subtitle for the columns. Default "Cells".

rowGap A numeric value or a unit object. For the gap size between rows of the splitted heatmap. Default grid::unit(0, 'mm').
A numeric value or a unit object. For the gap size between columns of the splitted heatmap. Default `grid::unit(0, 'mm')`.

border
A logical scalar. Whether to show the border of the heatmap or splitted heatmaps. Default `TRUE`.

colorScheme
A function. A function that generates color code by giving a value. Can be generated by `colorRamp2`. Default `NULL`.

... Other arguments passed to `Heatmap`.

Value
A `ggplot` object.

Author(s)
Yichen Wang

Examples
```r
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
```r
plotSCEScatter(
inSCE, annotation, reducedDimName = NULL, slot = NULL, sample = NULL, feature = NULL, groupBy = NULL, shape = NULL, conditionClass = NULL, xlab = NULL, ylab = NULL, axisSize = 10, axisLabelSize = 10, dim1 = NULL, dim2 = NULL,
```
Arguments

- **inSCE**: Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results. Required.
- **annotation**: Desired vector within the slot used for plotting. Default NULL.
- **reducedDimName**: 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.
- **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.
- **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.
plotSCEScatter

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 0.1.

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

data("mouseBrainSubsetSCE")
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,  
  slotName,  
  itemName,  
  feature = NULL,  
  sample = NULL,  
  dimension = NULL,  
  groupBy = NULL,  
  violin = TRUE,  
  boxplot = TRUE,  
  dots = TRUE,  
  plotOrder = NULL,  
  xlab = NULL,  
  ylab = NULL,  
  axisSize = 10,  
  axisLabelSize = 10,  
  dotSize = 0.1,  
  transparency = 1,  
  defaultTheme = TRUE,  
  gridLine = FALSE,  
  summary = NULL,  
  title = NULL,  
  titleSize = NULL,  
  hcutoff = NULL,  
  hcolor = "red",  
  hsize = 1,  
  hlinetype = 1,  
  vcutoff = NULL,  
  vcolor = "red",  
  vsize = 1,  
  vlinetype = 1,  
  combinePlot = "none",  
  plotLabels = NULL  
)

Arguments

inSCE Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results. Required.
### plotSCEViolin

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>slotName</td>
<td>Desired slot of SingleCellExperiment used for plotting. Possible options: &quot;assays&quot;, &quot;colData&quot;, &quot;metadata&quot;, &quot;reducedDims&quot;. Required.</td>
</tr>
<tr>
<td>itemName</td>
<td>Desired name vector within the slot used for plotting. Required.</td>
</tr>
<tr>
<td>feature</td>
<td>Desired name of feature stored in assay of SingleCellExperiment object. Only used when &quot;assays&quot; slotName is selected. Default NULL.</td>
</tr>
<tr>
<td>sample</td>
<td>Character vector. Indicates which sample each cell belongs to.</td>
</tr>
<tr>
<td>dimension</td>
<td>Desired dimension stored in the specified reducedDims. Either an integer which indicates the column or a character vector specifies column name. By default, the 1st dimension/column will be used. Only used when &quot;reducedDims&quot; slotName is selected. Default NULL.</td>
</tr>
<tr>
<td>groupBy</td>
<td>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.</td>
</tr>
<tr>
<td>violin</td>
<td>Boolean. If TRUE, will plot the violin plot. Default TRUE.</td>
</tr>
<tr>
<td>boxplot</td>
<td>Boolean. If TRUE, will plot boxplots for each violin plot. Default TRUE.</td>
</tr>
<tr>
<td>dots</td>
<td>Boolean. If TRUE, will plot dots for each violin plot. Default TRUE.</td>
</tr>
<tr>
<td>plotOrder</td>
<td>Character vector. If set, reorders the violin plots in the order of the character vector when 'groupBy' is set. Default NULL.</td>
</tr>
<tr>
<td>xlab</td>
<td>Character vector. Label for x-axis. Default NULL.</td>
</tr>
<tr>
<td>ylab</td>
<td>Character vector. Label for y-axis. Default NULL.</td>
</tr>
<tr>
<td>axisSize</td>
<td>Size of x/y-axis ticks. Default 10.</td>
</tr>
<tr>
<td>axisLabelSize</td>
<td>Size of x/y-axis labels. Default 10.</td>
</tr>
<tr>
<td>dotSize</td>
<td>Size of dots. Default 0.1.</td>
</tr>
<tr>
<td>transparency</td>
<td>Transparency of the dots, values will be 0-1. Default 1.</td>
</tr>
<tr>
<td>defaultTheme</td>
<td>Removes grid in plot and sets axis title size to 10 when TRUE. Default TRUE.</td>
</tr>
<tr>
<td>gridLine</td>
<td>Adds a horizontal grid line if TRUE. Will still be drawn even if defaultTheme is TRUE. Default FALSE.</td>
</tr>
<tr>
<td>summary</td>
<td>Adds a summary statistic, as well as a crossbar to the violin plot. Options are &quot;mean&quot; or &quot;median&quot;. Default NULL.</td>
</tr>
<tr>
<td>title</td>
<td>Title of plot. Default NULL.</td>
</tr>
<tr>
<td>titleSize</td>
<td>Size of title of plot. Default 15.</td>
</tr>
<tr>
<td>hcutoff</td>
<td>Adds a horizontal line with the y-intercept at given value. Default NULL.</td>
</tr>
<tr>
<td>hcolor</td>
<td>Character. A color available from 'colors()'. Controls the color of the horizontal cutoff line, if drawn. Default 'black'.</td>
</tr>
<tr>
<td>hsize</td>
<td>Size of horizontal line, if drawn. Default 0.5.</td>
</tr>
<tr>
<td>hlinetype</td>
<td>Type of horizontal line, if drawn. can be specified with either an integer or a name (0 = blank, 1 = solid, 2 = dashed, 3 = dotted, 4 = dotdash, 5 = longdash, 6 = twodash). Default 1.</td>
</tr>
<tr>
<td>vcutoff</td>
<td>Adds a vertical line with the x-intercept at given value. Default NULL.</td>
</tr>
<tr>
<td>vcolor</td>
<td>Character. A color available from 'colors()'. Controls the color of the vertical cutoff line, if drawn. Default 'black'.</td>
</tr>
</tbody>
</table>
plotSCEViolinAssayData

value

vsize

Size of vertical line, if drawn. Default 0.5.

vlinetype

Type of vertical line, if drawn. can be specified with either an integer or a name
(0 = blank, 1 = solid, 2 = dashed, 3 = dotted, 4 = dotdash, 5 = longdash, 6 = twodash). Default 1.

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

data("mouseBrainSubsetSCE")
plotSCEViolin(
inSCE = mouseBrainSubsetSCE, slotName = "assays",
itemName = "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,
feature,
sample = NULL,
useAssay = "counts",
featureLocation = NULL,
featureDisplay = NULL,
groupBy = NULL,
vviolin = TRUE,
boxplot = TRUE,
dots = TRUE,
plotOrder = NULL,
xlab = NULL,
ylab = NULL,
axisSize = 10,
Arguments

**inSCE**  
Input `SingleCellExperiment` object with saved dimension reduction components or a variable with saved results. Required.

**feature**  
Name of feature stored in assay of `SingleCellExperiment` object.

**sample**  
Character vector. Indicates which sample each cell belongs to.

**useAssay**  
Indicate which assay to use. Default "counts".

**featureLocation**  
Indicates which column name of rowData to query gene.

**featureDisplay**  
Indicates which column name of rowData to use to display feature for visualization.

**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.

**plotOrder**  
Character vector. If set, reorders the violin plots in the order of the character vector when 'groupBy' is set. 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 0.1.
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.
hcutoff  Adds a horizontal line with the y-intercept at given value. Default NULL.
hcolor  Character. A color available from `colors()`. Controls the color of the horizontal cutoff line, if drawn. Default 'black'.
hsize  Size of horizontal line, if drawn. Default 0.5.
hlinetype  Type of horizontal line, if drawn. can be specified with either an integer or a name (0 = blank, 1 = solid, 2 = dashed, 3 = dotted, 4 = dotdash, 5 = longdash, 6 = twodash). Default 1.
v_cutoff  Adds a vertical line with the x-intercept at given value. Default NULL.
vcolor  Character. A color available from `colors()`. Controls the color of the vertical cutoff line, if drawn. Default 'black'.
vsize  Size of vertical line, if drawn. Default 0.5.
vlinetype  Type of vertical line, if drawn. can be specified with either an integer or a name (0 = blank, 1 = solid, 2 = dashed, 3 = dotted, 4 = dotdash, 5 = longdash, 6 = twodash). Default 1.
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

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

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

Usage

```r
plotSCEViolinColData(
  inSCE, 
  coldata, 
  sample = NULL, 
  groupBy = NULL, 
  violin = TRUE, 
  boxplot = TRUE, 
  dots = TRUE, 
  plotOrder = NULL, 
  xlab = NULL, 
  ylab = NULL, 
  baseSize = 12, 
  axisSize = NULL, 
  axisLabelSize = NULL, 
  dotSize = 0.1, 
  transparency = 1, 
  defaultTheme = TRUE, 
  gridLine = FALSE, 
  summary = NULL, 
  summaryTextSize = 3, 
  title = NULL, 
  titleSize = NULL, 
  hcutoff = NULL, 
  hcolor = "red", 
  hsize = 1, 
  hlinetype = 1, 
  vcutoff = NULL, 
  vcolor = "red", 
  vsize = 1, 
  vlinetype = 1, 
  combinePlot = "none", 
  plotLabels = NULL
)
```

Arguments

- **inSCE**: Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results. Required.
coldata  colData value that will be plotted.
sample  Character vector. Indicates which sample each cell belongs to.
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.
plotOrder  Character vector. If set, reorders the violin plots in the order of the character vector when 'groupBy' is set. Default NULL.
xlab  Character vector. Label for x-axis. Default NULL.
ylab  Character vector. Label for y-axis. Default NULL.
baseSize  The base font size for all text. Default 12. Can be overwritten by titleSize, axisSize, and axisLabelSize.
axisSize  Size of x/y-axis ticks. Default NULL.
axisLabelSize  Size of x/y-axis labels. Default NULL.
dotSize  Size of dots. Default 0.1.
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.
summaryTextSize  The text size of the summary statistic displayed above the violin plot. Default 3.
title  Title of plot. Default NULL.
titleSize  Size of title of plot. Default 15.
hcutoff  Adds a horizontal line with the y-intercept at given value. Default NULL.
hcolor  Character. A color available from ‘colors()’. Controls the color of the horizontal cutoff line, if drawn. Default 'black'.
hsize  Size of horizontal line, if drawn. Default 0.5.
hlinetype  Type of horizontal line, if drawn. can be specified with either an integer or a name (0 = blank, 1 = solid, 2 = dashed, 3 = dotted, 4 = dotdash, 5 = longdash, 6 = twodash). Default 1.
v cutoff  Adds a vertical line with the x-intercept at given value. Default NULL.
vcolor  Character. A color available from ‘colors()’. Controls the color of the vertical cutoff line, if drawn. Default 'black'.
vs size  Size of vertical line, if drawn. Default 0.5.
vlinetype  Type of vertical line, if drawn. can be specified with either an integer or a name (0 = blank, 1 = solid, 2 = dashed, 3 = dotted, 4 = dotdash, 5 = longdash, 6 = twodash). Default 1.
**plotScrubletResults**

`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**

```r
data("mouseBrainSubsetSCE")
plotSCEViolinColData(
  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**

```r
plotScrubletResults(
  inSCE, reducedDimName, sample = NULL, shape = NULL, groupBy = NULL, combinePlot = "all", violin = TRUE, boxplot = FALSE, dots = TRUE, xlab = NULL, ylab = NULL, dim1 = NULL, dim2 = NULL, bin = NULL, binLabel = NULL, defaultTheme = TRUE, dotSize = 0.5, summary = "median",
)```
summaryTextSize = 3,
transparency = 1,
baseSize = 15,
titleSize = NULL,
axisLabelSize = NULL,
axisSize = NULL,
legendSize = NULL,
legendTitleSize = NULL,
relHeights = 1,
relWidths = c(1, 1, 1),
plotNCols = NULL,
plotNRows = NULL,
labelSamples = TRUE,
samplePerColumn = TRUE,
sampleRelHeights = 1,
sampleRelWidths = 1
)

Arguments

inSCE  Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results from runCxsds. Required.
reducedDimName  Saved dimension reduction name in inSCE. Default "UMAP".
sample  Character vector or colData variable name. Indicates which sample each cell belongs to. Default NULL.
shape  If provided, add shapes based on the value. Default NULL.
groupBy  Groupings for each numeric value. A user may input a vector equal length to the number of the samples in inSCE, 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".
vviolin  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.
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. Similar to dim1. Default is NULL.
bin  Numeric vector. If single value, will divide the numeric values into bin groups. If more than one value, will bin numeric values using values as a cut point. Default NULL.
binLabel  Character vector. Labels for the bins created by bin. Default NULL.
plotScrubletResults

defaultTheme: Removes grid in plot and sets axis title size to 10 when TRUE. Default TRUE.
dotSize: Size of dots. Default 0.5.
summary: Adds a summary statistic, as well as a crossbar to the violin plot. Options are "mean" or "median". Default NULL.
summaryTextSize: The text size of the summary statistic displayed above the violin plot. Default 3.
transparency: Transparency of the dots, values will be 0-1. Default 1.
titleSize: Size of title of plot. Default NULL.
axisLabelSize: Size of x/y-axis labels. Default NULL.
axisSize: Size of x/y-axis ticks. Default NULL.
legendSize: Size of legend. Default NULL.
legendTitleSize: Size of legend title. Default NULL.
relHeights: Relative heights of plots when combine is set. Default 1.
relWidths: Relative widths of plots when combine is set. Default c(1, 1, 1).
plotNCols: Number of columns when plots are combined in a grid. Default NULL.
plotNRows: Number of rows when plots are combined in a grid. Default NULL.
labelSamples: Will label sample name in title of plot if TRUE. Default TRUE.
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. Default 1.
sampleRelWidths: If there are multiple samples and combining by "all", the relative widths for each plot. Default 1.

Value
list of .ggplot objects

See Also
runScrublet

Examples

data(scExample, package="singleCellTK")
# Not run:
sce <- subsetSECEols(sce, colData = "type != 'EmptyDroplet'")
sce <- runQuickUMAP(sce)
sce <- runScrublet(sce)
plotScrubletResults(inSCE=sce, reducedDimName="UMAP")

## End(Not run)

### plotSeuratElbow

**plotSeuratElbow** Computes the plot object for elbow plot from the pca slot in the input sce object

#### Description

plotSeuratElbow Computes the plot object for elbow plot from the pca slot in the input sce object

#### Usage

```r
plotSeuratElbow(
  inSCE,
  significantPC = NULL,
  reduction = "pca",
  ndims = 20,
  externalReduction = NULL,
  interactive = TRUE
)
```

#### 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".
- **ndims** Number of components to use. Default 20.
- **externalReduction** Pass DimReduc object if PCA/ICA computed through other libraries. Default NULL.
- **interactive** Logical value indicating if the returned object should be an interactive plotly object if TRUE or a ggplot object if set to FALSE. Default is TRUE.

#### Value

- plot object
Examples

```r
data(scExample, package = "singleCellTK")
## Not run:
sce <- runSeuratNormalizeData(sce, useAssay = "counts")
sce <- runSeuratFindHVG(sce, useAssay = "counts")
sce <- runSeuratScaleData(sce, useAssay = "counts")
sce <- runSeuratPCA(sce, useAssay = "counts")
plotSeuratElbow(sce)
## End(Not run)
```

---

**plotSeuratGenes**

Compute and plot visualizations for marker genes

**Description**

Compute and plot visualizations for marker genes

**Usage**

```r
plotSeuratGenes(
inSCE,  # Input SingleCellExperiment object.
useAssay = "seuratNormData",  # Specify the name of the assay that will be scaled by this function.
plotType,  # Specify the type of the plot to compute. Options are limited to "ridge", "violin", "feature", "dot" and "heatmap".
features,  # Specify the features to compute the plot against.
groupVariable,  # Specify the column name from the colData slot that should be used as grouping variable.
reducedDimName = "seuratUMAP",  # saved dimension reduction name in the SingleCellExperiment object. Default seuratUMAP.
splitBy = NULL,  # Specify the features to compute the plot against.
ncol = 1,  # Specify the column name from the colData slot that should be used as grouping variable.
useReduction = c("umap", "pca", "ica", "tsne"),  # Options are limited to "ridge", "violin", "feature", "dot" and "heatmap".
combine = FALSE  # Combine the plots.
)  # }  
```
splitBy Specify the column name from the colData slot that should be used to split samples. Default is NULL.

cols Specify two colors to form a gradient between. Default is c("lightgrey", "blue").

ncol Visualizations will be adjusted in "ncol" number of columns. Default is 1.

useReduction Dimentionality reduction to plot. One of "pca", "ica", "tsne", or "umap". Default "umap".

combine A logical value that indicates if the plots should be combined together into a single plot if TRUE, else if FALSE returns separate ggplot objects for each feature. Only works when plotType parameter is "feature", "violin" or "ridge". For "heatmap" and "dot", plots for all features are always combined into a single plot. Default FALSE.

Value
Plot object

plotSeuratHeatmap plotSeuratHeatmap Modifies the heatmap plot object so it contains specified number of heatmaps in a single plot

Description
plotSeuratHeatmap Modifies the heatmap plot object so it contains specified number of heatmaps in a single plot

Usage
plotSeuratHeatmap(plotObject, dims, ncol, labels)

Arguments
plotObject plot object computed from runSeuratHeatmap() 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
plotSeuratHVG  

plotSeuratHVG: Plot highly variable genes from input sce object (must have highly variable genes computations stored)

Description

plotSeuratHVG: Plot highly variable genes from input sce object (must have highly variable genes computations stored)

Usage

plotSeuratHVG(inSCE, labelPoints = 0)

Arguments

inSCE: (sce) object that contains the highly variable genes computations
labelPoints: Numeric value indicating the number of top genes that should be labeled. Default is 0, which will not label any point.

Value

plot object

Examples

data(scExample, package = "singleCellTK")  
## Not run:  
sce <- runSeuratNormalizeData(sce, useAssay = "counts")  
sce <- runSeuratFindHVG(sce, useAssay = "counts")  
plotSeuratHVG(sce)

## End(Not run)

plotSeuratJackStraw  

plotSeuratJackStraw: Computes the plot object for jackstraw plot from the pca slot in the input sce object

Description

plotSeuratJackStraw: Computes the plot object for jackstraw plot from the pca slot in the input sce object
Usage

plotSeuratJackStraw(
  inSCE,
  dims = NULL,
  xmax = 0.1,
  ymax = 0.3,
  externalReduction = 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.
xmax X-axis maximum on each QQ plot. Default 0.1.
ymax Y-axis maximum on each QQ plot. Default 0.3.
externalReduction Pass DimReduc object if PCA/ICA computed through other libraries. Default NULL.

Value

plot object

Examples

data(scExample, package = "singleCellTK")
## Not run:
sce <- runSeuratNormalizeData(sce, useAssay = "counts")
sce <- runSeuratFindHVG(sce, useAssay = "counts")
sce <- runSeuratScaleData(sce, useAssay = "counts")
sce <- runSeuratPCA(sce, useAssay = "counts")
sce <- runSeuratJackStraw(sce, useAssay = "counts")
plotSeuratJackStraw(sce)
## End(Not run)

plotSeuratReduction

plotSeuratReduction Plots the selected dimensionality reduction method

Description

plotSeuratReduction Plots the selected dimensionality reduction method
plotSoupXResults

Usage

plotSeuratReduction(
inSCE,
useReduction = c("pca", "ica", "tsne", "umap"),
showLegend = FALSE,
groupBy = NULL,
splitBy = NULL
)

Arguments

inSCE (sce) object which has the selected dimensionality reduction algorithm already computed and stored
useReduction Dimentionality reduction to plot. One of "pca", "ica", "tsne", or "umap". Default "umap".
showLegend Select if legends and labels should be shown on the output plot or not. Either "TRUE" or "FALSE". Default FALSE.
groupBy Specify a colData column name that be used for grouping. Default is NULL.
splitBy Specify a colData column name that be used for splitting the output plot. Default is NULL.

Value

plot object

Examples

data(scExample, package = "singleCellTK")
## Not run:
sce <- runSeuratNormalizeData(sce, useAssay = "counts")
sce <- runSeuratFindHVG(sce, useAssay = "counts")
sce <- runSeuratScaleData(sce, useAssay = "counts")
sce <- runSeuratPCA(sce, useAssay = "counts")
plotSeuratReduction(sce, useReduction = "pca")
## End(Not run)

plotSoupXResults  Plot SoupX Result

Description

This function will generate a combination of plots basing on the correction done by SoupX. For each sample, there will be a UMAP with cluster labeling, followed by a number of UMAPs showing the change in selected top markers. The cluster labeling is what should be used for SoupX to estimate the contamination. The Soup Fraction is calculated by subtracting the gene expression value of the output corrected matrix from that of the original input matrix, and then divided by the input.
plotSoupXResults

Usage

plotSoupXResults(
inSCE, 
sample = NULL, 
background = FALSE, 
reducedDimName = NULL, 
plotNCols = 3, 
plotNRows = 2, 
baseSize = 8, 
combinePlot = c("all", "sample", "none"), 
xlab = NULL, 
ylab = NULL, 
dim1 = NULL, 
dim2 = NULL, 
labelClusters = FALSE, 
clusterLabelSize = 3.5, 
defaultTheme = TRUE, 
dotSize = 0.5, 
transparency = 1, 
titleSize = NULL, 
axisLabelSize = NULL, 
axisSize = NULL, 
legendSize = NULL, 
legendTitleSize = NULL)

Arguments

inSCE A SingleCellExperiment object. With runSoupX already applied.
sample Character vector. Indicates which sample each cell belongs to. Default NULL.
background Logical. Whether background was applied when running runSoupX. Default FALSE.
reducedDimName Character. The embedding to use for plotting. Leave it NULL for using the sample-specific UMAPs generated when running runSoupX. Default NULL.
plotNCols Integer. Number of columns for the plot grid per sample. Will determine the number of top markers to show together with plotNRows. Default 3.
plotNRows Integer. Number of rows for the plot grid per sample. Will determine the number of top markers to show together with plotNCols. Default 2.
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".
xlab Character vector. Label for x-axis. Default NULL.
ylab Character vector. Label for y-axis. Default NULL.
**plotTopHVG**

Plot highly variable genes

**Description**

Plot highly variable genes

---

**dim1**

See `plotSCEDimReduceColData`. Default NULL.

**dim2**

See `plotSCEDimReduceColData`. Default NULL.

**labelClusters**

Logical. Whether the cluster labels are plotted. Default FALSE.

**clusterLabelSize**

Numeric. Determines the size of cluster label when `labelClusters` is set to TRUE. Default 3.5.

**defaultTheme**

Logical. Adds grid to plot when TRUE. Default TRUE.

**dotSize**

Numeric. Size of dots. Default 0.5.

**transparency**

Numeric. Transparency of the dots, values will be from 0 to 1. Default 1.

**titleSize**


**axisLabelSize**

Numeric. Size of x/y-axis labels. Default NULL.

**axisSize**

Numeric. Size of x/y-axis ticks. Default NULL.

**legendSize**

Numeric. Size of legend. Default NULL.

**legendTitleSize**

Numeric. Size of legend title. Default NULL.

---

**Value**

ggplot object of the combination of UMAPs. See description.

**See Also**

`runSoupX`

---

**Examples**

```r
## Not run:
sce <- importExampleData("pbmc3k")
sce <- runSoupX(sce, sample = "sample")
plotSoupXResults(sce, sample = "sample")

## End(Not run)
```
Usage

plotTopHVG(
  inSCE,
  method = c("vst", "mean.var.plot", "dispersion", "modelGeneVar"),
  hvgNumber = NULL,
  useFeatureSubset = NULL,
  labelsCount = 20,
  featureDisplay = metadata(inSCE)$featureDisplay,
  labelSize = 2,
  dotSize = 2,
  textSize = 12
)

Arguments

inSCE Input SingleCellExperiment object containing the computations.
method Select either "vst", "mean.var.plot", "dispersion" or "modelGeneVar".
hvgNumber Specify the number of top genes to highlight in red. Default NULL. See details.
useFeatureSubset A character string for the rowData variable name to store a logical index of selected features. Default NULL. See details.
labelsCount Specify the number of data points/genes to label. Should be less than hvgNumber. Default 20. See details.
featureDisplay A character string for the rowData variable name to indicate what type of feature ID should be displayed. If set by setSCTKDisplayRow, will by default use it. If NULL, will use rownames(inSCE).
labelSize Numeric, size of the text label on top HVGs. Default 2.
dotSize Numeric, size of the dots of the features. Default 2.
textSize Numeric, size of the text of axis title, axis label, etc. Default 12.

Details

When hvgNumber = NULL and useFeature = NULL, only plot the mean VS variance/dispersion scatter plot. When only hvgNumber set, label the top hvgNumber HVGs ranked by the metrics calculated by method. When useFeatureSubset set, label the features in the subset on the scatter plot created with method and ignore hvgNumber.

Value

ggplot of HVG metrics and top HVG labels

See Also

runFeatureSelection, runSeuratFindHVG, runModelGeneVar, getTopHVG
Examples

data("mouseBrainSubsetSCE", package = "singleCellTK")
mouseBrainSubsetSCE <- runModelGeneVar(mouseBrainSubsetSCE)
plotTopHVG(mouseBrainSubsetSCE, method = "modelGeneVar")

Description

A wrapper function which plot the top features expression identified by runTSCANClusterDEAnalysis on the 2D embedding of the cells cluster used in the analysis. The related MST edges are overlaid.

Usage

plotTSCANClusterDEG(
  inSCE,
  useCluster,
  pathIndex = NULL,
  useReducedDim = "UMAP",
  topN = 9,
  useAssay = NULL,
  featureDisplay = metadata(inSCE)$featureDisplay,
  combinePlot = c("all", "none")
)

Arguments

inSCE Input SingleCellExperiment object.
useCluster Choose a cluster used for identifying DEG with runTSCANClusterDEAnalysis. Required.
pathIndex Specifies one of the branching paths from useCluster and plot the top DEGs on this path. Usually presented by the terminal cluster of a path. By default NULL plot top DEGs of all paths.
useReducedDim A single character for the matrix of 2D embedding. Should exist in reducedDims slot. Default "UMAP".
topN Integer. Use top N genes identified. Default 9.
useAssay A single character for the feature expression matrix. Should exist in assayNames(inSCE). Default NULL for using the one used in runTSCANClusterDEAnalysis.
featureDisplay Specify the feature ID type to display. Users can set default value with setSCTKDisplayRow. NULL or "rownames" specifies the rownames of inSCE. Other character values indicates rowData variable.
combinePlot Must be either "all" or "none". "all" will combine plots of each feature into a single ggplot object, while "none" will output a list of plots. Default "all".
Value

A .ggplot object of cell scatter plot, colored by the expression of a gene identified by runTSCANClusterDEAnalysis, with the layer of trajectory.

Author(s)

Yichen Wang

Examples

data("mouseBrainSubsetSCE", package = "singleCellTK")
mouseBrainSubsetSCE <- runTSCAN(inSCE = mouseBrainSubsetSCE, 
    useReducedDim = "PCA_logcounts")
mouseBrainSubsetSCE <- runTSCANClusterDEAnalysis(inSCE = mouseBrainSubsetSCE, 
    useCluster = 1)
plotTSCANClusterDEG(mouseBrainSubsetSCE, useCluster = 1, 
    useReducedDim = "TSNE_logcounts")

plotTSCANClusterPseudo

Plot TSCAN pseudotime rooted from given cluster

Description

This function finds all paths that root from a given cluster useCluster. For each path, this function plots the recomputed pseudotime starting from the root on a scatter plot which contains cells only in this cluster. MST has to be pre-calculated with runTSCAN.

Usage

plotTSCANClusterPseudo(
    inSCE, 
    useCluster, 
    useReducedDim = "UMAP", 
    combinePlot = c("all", "none")
)

Arguments

inSCE Input SingleCellExperiment object.
useCluster The cluster to be regarded as the root, has to existing in colData(inSCE)$TSCAN_clusters.
useReducedDim Saved dimension reduction name in the SingleCellExperiment object. Required.
combinePlot Must be either "all" or "none". "all" will combine plots of pseudotime along each path into a single .ggplot object, while "none" will output a list of plots. Default "all".
plotTSCANDimReduceFeatures

Value

combinePlot = "all"
   A ggplot object

combinePlot = "none"
   A list of ggplot

Author(s)

Nida Pervaiz

Examples

data("mouseBrainSubsetSCE", package = "singleCellTK")
mouseBrainSubsetSCE <- runTSCAN(inSCE = mouseBrainSubsetSCE,
   useReducedDim = "PCA_logcounts")
plotTSCANClusterPseudo(mouseBrainSubsetSCE, useCluster = 1,
   useReducedDim = "TSNE_logcounts")

Description

A wrapper function which plots all cells or cells in chosen cluster. Each point is a cell colored by the expression of a feature of interest, the relevant edges of the MST are overlaid on top.

Usage

plotTSCANDimReduceFeatures(
   inSCE,
   features,
   useReducedDim = "UMAP",
   useAssay = "logcounts",
   by = "rownames",
   useCluster = NULL,
   featureDisplay = metadata(inSCE)$featureDisplay,
   combinePlot = c("all", "none")
)

Arguments

inSCE Input SingleCellExperiment object.
features Choose the feature of interest to explore the expression level on the trajectory. Required.
useReducedDim A single character for the matrix of 2D embedding. Should exist in reducedDims slot. Default "UMAP".
**Description**

A wrapper function which visualizes outputs from the `runTSCANDEG` function. Plots the genes that increase or decrease in expression with increasing pseudotime along the path in the MST. `runTSCANDEG` has to be run in advance with using the same `pathIndex` of interest.
Usage

```
plotTSCANPseudotimeGenes(
  inSCE,  # Input SingleCellExperiment object.
  pathIndex,  # Path index for which the pseudotime values should be used. Should have being used in runTSCANDEG.
  direction = c("increasing", "decreasing"),  # Should we show features with expression increasing or decreasing along the increase in TSCAN pseudotime? Choices are "increasing" or "decreasing".
  topN = 10,  # An integer. Only to plot this number of top genes that are increasing/decreasing in expression with increasing pseudotime along the path in the MST. Default 10
  useAssay = NULL,  # A single character to specify a feature expression matrix in assays slot. The expression of top features from here will be visualized. Default NULL use the one used for runTSCANDEG.
  featureDisplay = metadata(inSCE)$featureDisplay  # Specify the feature ID type to display. Users can set default value with setSCTKDisplayRow. NULL or "rownames" specifies the rownames of inSCE. Other character values indicates rowData variable.
)
```

Arguments

- **inSCE**: Input SingleCellExperiment object.
- **pathIndex**: Path index for which the pseudotime values should be used. Should have being used in runTSCANDEG.
- **direction**: Should we show features with expression increasing or decreasing along the increase in TSCAN pseudotime? Choices are "increasing" or "decreasing".
- **topN**: An integer. Only to plot this number of top genes that are increasing/decreasing in expression with increasing pseudotime along the path in the MST. Default 10
- **useAssay**: A single character to specify a feature expression matrix in assays slot. The expression of top features from here will be visualized. Default NULL use the one used for runTSCANDEG.
- **featureDisplay**: Specify the feature ID type to display. Users can set default value with setSCTKDisplayRow. NULL or "rownames" specifies the rownames of inSCE. Other character values indicates rowData variable.

Value

A .ggplot object with the facets of the top genes. Expression on y-axis, pseudotime on x-axis.

Author(s)

Nida Pervaiz

Examples

```r
data("mouseBrainSubsetSCE", package = "singleCellTK")
mouseBrainSubsetSCE <- runTSCAN(inSCE = mouseBrainSubsetSCE,  # runTSCAN(inSCE = mouseBrainSubsetSCE, useReducedDim = "PCA_logcounts")
terminalNodes <- listTSCANTerminalNodes(mouseBrainSubsetSCE)  # listTSCANTerminalNodes(mouseBrainSubsetSCE)
mouseBrainSubsetSCE <- runTSCANDEG(inSCE = mouseBrainSubsetSCE,  # runTSCANDEG(inSCE = mouseBrainSubsetSCE, pathIndex = terminalNodes[1])
plotTSCANPseudotimeGenes(mouseBrainSubsetSCE,  # plotTSCANPseudotimeGenes(mouseBrainSubsetSCE, pathIndex = terminalNodes[1], useAssay = "logcounts")
```
plotTSCANPseudotimeHeatmap

Plot heatmap of genes with expression change along TSCAN pseudotime

Description

A wrapper function which visualizes outputs from the runTSCANDEG function. Plots the top genes that change in expression with increasing pseudotime along the path in the MST. runTSCANDEG has to be run in advance with using the same pathIndex of interest.

Usage

plotTSCANPseudotimeHeatmap(
  inSCE,  # Input SingleCellExperiment object.
  pathIndex,  # Path index for which the pseudotime values should be used. Should have being used in runTSCANDEG.
  direction = c("both", "increasing", "decreasing"),  # Should we show features with expression increasing or decreasing along the increase in TSCAN pseudotime? Choices are "both", "increasing" or "decreasing".
  topN = 50,  # An integer. Only to plot this number of top genes along the path in the MST, in terms of FDR value. Use NULL to cancel the top N subscription. Default 30.
  log2fcThreshold = NULL,  # Only output DEGs with the absolute values of log2FC larger than this value. Default NULL.
  useAssay = NULL,  # A single character to specify a feature expression matrix in assays slot. The expression of top features from here will be visualized. Default NULL use the one used for runTSCANDEG.
  featureDisplay = metadata(inSCE)$featureDisplay  # Whether to display feature ID and what ID type to display. Users can set default ID type by setSCTKDisplayRow. NULL will display when number of features to display is less than 60. FALSE for no display. Variable name in rowData to indicate ID type. "rownames" or TRUE for using rownames(inSCE).
)

Arguments

inSCE Input SingleCellExperiment object.
pathIndex Path index for which the pseudotime values should be used. Should have being used in runTSCANDEG.
direction Should we show features with expression increasing or decreasing along the increase in TSCAN pseudotime? Choices are "both", "increasing" or "decreasing".
topN An integer. Only to plot this number of top genes along the path in the MST, in terms of FDR value. Use NULL to cancel the top N subscription. Default 30.
log2fcThreshold Only output DEGs with the absolute values of log2FC larger than this value. Default NULL.
useAssay A single character to specify a feature expression matrix in assays slot. The expression of top features from here will be visualized. Default NULL use the one used for runTSCANDEG.
featureDisplay Whether to display feature ID and what ID type to display. Users can set default ID type by setSCTKDisplayRow. NULL will display when number of features to display is less than 60. FALSE for no display. Variable name in rowData to indicate ID type. "rownames" or TRUE for using rownames(inSCE).

Value

A ComplexHeatmap in ggplot class
plotTSCANResults

Author(s)
Nida Pervaiz

Examples

```r
data("mouseBrainSubsetSCE", package = "singleCellTK")
mouseBrainSubsetSCE <- runTSCAN(inSCE = mouseBrainSubsetSCE, 
  useReducedDim = "PCA_logcounts")
terminalNodes <- listTSCANTerminalNodes(mouseBrainSubsetSCE)
mouseBrainSubsetSCE <- runTSCANDEG(inSCE = mouseBrainSubsetSCE, 
  pathIndex = terminalNodes[1])
plotTSCANPseudotimeHeatmap(mouseBrainSubsetSCE, 
  pathIndex = terminalNodes[1])
data("mouseBrainSubsetSCE", package = "singleCellTK")
mouseBrainSubsetSCE <- runTSCAN(inSCE = mouseBrainSubsetSCE, 
  useReducedDim = "PCA_logcounts")
plotTSCANResults(inSCE = mouseBrainSubsetSCE, 
  useReducedDim = "TSNE_logcounts")
```

plotTSCANResults    Plot MST pseudotime values on cell 2D embedding

Description
A wrapper function which visualizes outputs from the `runTSCAN` function. Plots the pseudotime ordering of the cells and project them onto the MST.

Usage

```r
plotTSCANResults(inSCE, useReducedDim = "UMAP")
```

Arguments

- `inSCE` Input `SingleCellExperiment` object.
- `useReducedDim` Saved dimension reduction name in `inSCE` object. Required.

Value

A `.ggplot` object with the pseudotime ordering of the cells colored on a cell 2D embedding, and the MST path drawn on it.

Author(s)
Nida Pervaiz

Examples

```r
data("mouseBrainSubsetSCE", package = "singleCellTK")
mouseBrainSubsetSCE <- runTSCAN(inSCE = mouseBrainSubsetSCE, 
  useReducedDim = "PCA_logcounts")
plotTSCANResults(inSCE = mouseBrainSubsetSCE, 
  useReducedDim = "TSNE_logcounts")
```
plotTSNE  

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

Description

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

Usage

```r
plotTSNE(
  inSCE, 
  colorBy = NULL,
  shape = NULL,
  reducedDimName = "TSNE",
  runTSNE = FALSE,
  useAssay = "counts"
)
```

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

```r
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

```r
plotUMAP(
  inSCE,
  colorBy = NULL,
  shape = NULL,
  reducedDimName = "UMAP",
  runUMAP = FALSE,
  useAssay = "counts"
)
```

Arguments

- `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

```r
data(scExample, package = "singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
sce <- runQuickUMAP(sce)
plotUMAP(sce)
```
qcInputProcess

Create SingleCellExperiment object from command line input arguments

Description

Create SingleCellExperiment object from command line input arguments

Usage

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

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>preproc</td>
<td>Method used to preprocess the data. It’s one of the path provided in –preproc argument.</td>
</tr>
<tr>
<td>samplename</td>
<td>The sample name of the data. It’s one of the path provided in –sample argument.</td>
</tr>
<tr>
<td>path</td>
<td>Base path of the dataset. It’s one of the path provided in –bash_path argument.</td>
</tr>
<tr>
<td>raw</td>
<td>The directory contains droplet matrix, gene and cell barcodes information. It’s one of the path provided in –raw_data_path argument.</td>
</tr>
<tr>
<td>fil</td>
<td>The directory contains cell matrix, gene and cell barcodes information. It’s one of the path provided in –cell_data_path argument.</td>
</tr>
<tr>
<td>ref</td>
<td>The name of reference used by cellranger. Only need for CellrangerV2 data.</td>
</tr>
<tr>
<td>rawFile</td>
<td>The full path of the RDS file or Matrix file of the raw gene count matrix. It’s one of the path provided in –raw_data argument.</td>
</tr>
<tr>
<td>filFile</td>
<td>The full path of the RDS file or Matrix file of the cell count matrix. It’s one of the path provided in –cell_data argument.</td>
</tr>
<tr>
<td>flatFiles</td>
<td>The full paths of the matrix, barcode, and features (in that order) files used to construct an SCE object.</td>
</tr>
<tr>
<td>dataType</td>
<td>Type of the input. It can be &quot;Both&quot;, &quot;Droplet&quot; or &quot;Cell&quot;. It’s one of the path provided in –genome argument.</td>
</tr>
</tbody>
</table>
readSingleCellMatrix

Value

A list of `SingleCellExperiment` object containing the droplet or cell data or both, depending on the data type that users provided.

---

### Description

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

### Usage

```r
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, .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

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

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

**Usage**

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

**Arguments**

- **inSCE**: A SingleCellExperiment object containing the filtered count matrix with the output from runCellQC function.
- **output_file**: Character. The 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**: Character. The 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**: Character. The description of the data set and experiment design. It would be shown at the top of QC HTML report. Default is NULL.
- **useReducedDim**: Character. The name of the saved dimension reduction slot including cells from all samples in the SingleCellExperiment object. Default is NULL.

**Value**

.html file

**Examples**

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

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

Usage

```r
reportClusterAbundance(
  inSCE, 
  cluster, 
  variable, 
  output_dir = ".", 
  output_file = "plotClusterAbundance_Report", 
  pdf = FALSE, 
  showSession = TRUE
)
```

Arguments

- `inSCE`: A SingleCellExperiment object.
- `cluster`: A single character, specifying the name to store the cluster label in colData.
- `variable`: A single character, specifying the name to store the phenotype labels in colData.
- `output_dir`: name of the output directory to save the rendered file. If NULL the file is stored to the current working directory. Default NULL.
- `output_file`: name of the generated file. If NULL then the output file name will be based on the name of the Rmarkdown template. Default NULL.
- `pdf`: A logical value indicating if a pdf should also be generated for each figure in the report. Default is TRUE.
- `showSession`: A logical value indicating if session information should be displayed or not. Default is TRUE.

Value

An HTML file of the report will be generated at the path specified in the arguments.
reportDiffAbundanceFET

*Get diffAbundanceFET*.html report

**Description**

A function to generate .html Rmarkdown report containing the visualizations of the diffAbundance-FET function output

**Usage**

```r
reportDiffAbundanceFET(
  inSCE, cluster, variable, control, case, analysisName,
  output_dir = ".", output_file = "DifferentialAbundanceFET_Report",
  pdf = FALSE, showSession = TRUE
)
```

**Arguments**

- **inSCE** A `SingleCellExperiment` object.
- **cluster** A single character, specifying the name to store the cluster label in `colData`.
- **variable** A single character, specifying the name to store the phenotype labels in `colData`.
- **control** character. Specifying one or more categories that can be found in the vector specified by `variable`.
- **case** character. Specifying one or more categories that can be found in the vector specified by `variable`.
- **analysisName** A single character. Will be used for naming the result table, which will be saved in metadata slot.
- **output_dir** name of the output directory to save the rendered file. If NULL the file is stored to the current working directory. Default NULL.
- **output_file** name of the generated file. If NULL then the output file name will be based on the name of the Rmarkdown template. Default NULL.
- **pdf** A logical value indicating if a pdf should also be generated for each figure in the report. Default is TRUE.
- **showSession** A logical value indicating if session information should be displayed or not. Default is TRUE.
Value

An HTML file of the report will be generated at the path specified in the arguments.

---

**reportDiffExp** 

*Get runDEAnalysis .html report*

---

Description

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

Usage

```r
reportDiffExp(
  inSCE,
  study,
  useReducedDim,
  featureDisplay = NULL,
  output_file = NULL,
  output_dir = NULL
)
```

Arguments

- **inSCE**: A **SingleCellExperiment** object containing the output from **runDEAnalysis** function.
- **study**: The specific analysis to visualize, used as **analysisName** argument when running differential expression.
- **useReducedDim**: Specify an embedding for visualizing the relationship between the conditions.
- **featureDisplay**: The feature ID type to use for displaying. Should exist as a variable name of **rowData**. Default NULL use rownames of **inSCE**.
- **output_file**: name of the generated file. If NULL then the output file name will be based on the name of the Rmarkdown template. Default NULL.
- **output_dir**: name of the output directory to save the rendered file. If NULL the file is stored to the current working directory. Default NULL.

Value

Saves the HTML report in the specified output directory.
**reportDropletQC**

*Get runDropletQC .html report*

**Description**

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

**Usage**

```r
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**

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

**reportFindMarker**

*Get runFindMarker .html report*

---

**Description**

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

**Usage**

```r
reportFindMarker(inSCE, output_file = NULL, output_dir = NULL)
```

**Arguments**

- `inSCE`: A `SingleCellExperiment` object containing the output from `runFindMarker` function.
- `output_file`: name of the generated file. If NULL then the output file name will be based on the name of the Rmarkdown template. Default NULL.
- `output_dir`: name of the output directory to save the rendered file. If NULL the file is stored to the current working directory. Default NULL.

**Value**

An HTML file of the report will be generated at the path specified in the arguments.

---

**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**

```r
reportQCTool(
  inSCE, 
  algorithm = c("BarcodeRankDrops", "EmptyDrops", "QCMetrics", "Scrublet", "ScDb1Finder", 
                  "Cxds", "Bcds", "CxdsBcdsHybrid", "DoubletFinder", "DecontX", "SoupX"),
  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, runScDbfFinder, runCxs, runBcds, runCxsBcdsHybrid, runDecontX, runBarcodeRankDrops, runEmptyDrops


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 <- runQuickUmap(sce)
reportQCTool(inSCE = sce, algorithm = "DecontX")
## End(Not run)

reportSeurat Generates an HTML report for the complete Seurat workflow and returns the SCE object with the results computed and stored inside the object.

Description

Generates an HTML report for the complete Seurat workflow and returns the SCE object with the results computed and stored inside the object.

Usage

reportSeurat(
inSCE,
biological.group = NULL,
phenotype.groups = NULL,
selected.markers = NULL,
clustering.resolution = 0.8,
variable.features = 2000,
pc.count = 50,
outputFile = NULL,
outputPath = NULL,
subtitle = NULL,
authors = NULL,
showSession = FALSE,
pdf = FALSE,
runHVG = TRUE,
plotHVG = TRUE,
runDimRed = TRUE,
plotJackStraw = FALSE,
plotElbowPlot = TRUE,
plotHeatmaps = TRUE,
runClustering = TRUE,
plotTSNE = TRUE,
plotUMAP = TRUE,
minResolution = 0.3,
maxResolution = 1.5,
runMSClusters = TRUE,
runMSBioGroup = TRUE,
numTopFeatures = 10,
forceRun = TRUE
)

Arguments

inSCE Input SingleCellExperiment object.

biological.group A character value that specifies the name of the colData() column to use as the main biological group in the Seurat report for marker selection and grouping.

phenotype.groups A character vector that specifies the names of the colData() columns to use for differential expression in addition to the biological.group parameter.

selected.markers A character vector containing the user-specified gene symbols or feature names of marker genes that be used to generate gene plots in addition to the gene markers computed from differential expression.

clustering.resolution A numeric value indicating the user-specified final resolution to use with clustering. Default is 0.8.

variable.features A numeric value indicating the number of top variable features to identify. Default 2000.

pc.count A numeric value indicating the number of principal components to use in the analysis workflow. Default is 50.

outputFile Specify the name of the generated output HTML file. If NULL then the output file name will be based on the name of the Rmarkdown template. Default NULL.
outputPath  Specify the name of the output directory to save the rendered HTML file. If
NULL the file is stored to the current working directory. Default NULL.

subtitle  A character value specifying the subtitle to use in the report. Default NULL.

authors  A character value specifying the names of the authors to use in the report. De-
default NULL.

showSession  A logical value indicating if session information should be displayed or not. De-
default is FALSE.

pdf  A logical value indicating if a pdf should also be generated for each figure in the
report. Default is FALSE.

rnHVG  A logical value indicating if the feature selection computation should be run or
not. Default is TRUE.

plotHVG  A logical value indicating if the plot for the top most variable genes should be
visualized in a mean-to-variance plot. Default is TRUE.

runDimRed  A logical value indicating if PCA should be computed. Default is TRUE.

plotJackStraw  A logical value indicating if JackStraw plot be visualized for the principal com-
ponents. Default is FALSE.

plotElbowPlot  A logical value indicating if the ElbowPlot be visualized for the principal com-
ponents. Default is TRUE.

plotHeatmaps  A logical value indicating if heatmaps should be plotted for the principal com-
ponents. Default is TRUE.

runClustering  A logical value indicating if clustering section should be run in the report. De-
default is TRUE.

plotTSNE  A logical value indicating if TSNE plots should be visualized for clustering
results. Default is TRUE.

plotUMAP  A logical value indicating if the UMAP plots should be visualized for the clus-
tering results. Default is TRUE.

minResolution  A numeric value indicating the minimum resolution to use for clustering. De-
default is 0.3.

maxResolution  A numeric value indicating the maximum resolution to use for clustering. De-
default is 1.5.

runMSClusters  A logical value indicating if marker selection should be run between clusters.
Default is TRUE.

runMSBioGroup  A logical value indicating if marker selection should be run between the biological.group
parameter. Default is TRUE.

numTopFeatures  A numeric value indicating the number of top features to visualize in each group.
Default 10.

forceRun  A logical value indicating if all algorithms should be re-run regardless if they
have been computed previously in the input object. Default is TRUE.

Value

A SingleCellExperiment object with computations stored.
reportSeuratClustering

Generates an HTML report for Seurat Clustering and returns the SCE object with the results computed and stored inside the object.

Description

Generates an HTML report for Seurat Clustering and returns the SCE object with the results computed and stored inside the object.

Usage

reportSeuratClustering(
  inSCE,  
  biological.group = NULL,  
  phenotype.groups = NULL,  
  runClustering = TRUE,  
  plotTSNE = TRUE,  
  plotUMAP = TRUE,  
  minResolution = 0.3,  
  maxResolution = 1.5,  
  numClusters = 10,  
  significant_PC = 10,  
  outputFile = NULL,  
  outputPath = NULL,  
  subtitle = NULL,  
  authors = NULL,  
  showSession = FALSE,  
  pdf = FALSE,  
  forceRun = TRUE
)

Arguments

inSCE Input SingleCellExperiment object.

biological.group A character value that specifies the name of the colData() column to use as the main biological group in the Seurat report for marker selection and grouping.

phenotype.groups A character vector that specifies the names of the colData() columns to use for differential expression in addition to the biological.group parameter.

runClustering A logical value indicating if Clustering should be run or not in the report. Default is TRUE. If FALSE, parameters plotTSNE and plotUMAP are also set to FALSE.

plotTSNE A logical value indicating if TSNE plots should be visualized in the clustering section of the report. Default is TRUE.
plotUMAP A logical value indicating if UMAP plots should be visualized in the clustering section of the report. Default is TRUE.

minResolution A numeric value indicating the minimum resolution to use for clustering. Default 0.3.

maxResolution A numeric value indicating the maximum resolution to use for clustering. Default 1.5.

numClusters temp (to remove)

significant_PC temp (change to pc.use)

outputFile Specify the name of the generated output HTML file. If NULL then the output file name will be based on the name of the Rmarkdown template. Default NULL.

outputPath Specify the name of the output directory to save the rendered HTML file. If NULL the file is stored to the current working directory. Default NULL.

subtitle A character value specifying the subtitle to use in the report. Default NULL.

authors A character value specifying the names of the authors to use in the report. Default NULL.

showSession A logical value indicating if session information should be displayed or not. Default is FALSE.

pdf A logical value indicating if a pdf should also be generated for each figure in the report. Default is FALSE.

forceRun A logical value indicating if all computations previously computed should be re-calculated regardless if these computations are available in the input object. Default is TRUE.

Value

A SingleCellExperiment object with computations stored.

description

Generates an HTML report for Seurat Dimensionality Reduction and returns the SCE object with the results computed and stored inside the object.

Usage

reportSeuratDimRed(
  inSCE,
  pc.count = 50,
  runDimRed = TRUE,
  plotJackStraw = FALSE,
plotElbowPlot = TRUE,
plotHeatmaps = TRUE,
outputFile = NULL,
outputPath = NULL,
subtitle = NULL,
authors = NULL,
showSession = FALSE,
pdf = FALSE,
forceRun = TRUE
)

Arguments

inSCE Input SingleCellExperiment object.

pc.count A numeric value indicating the number of principal components to compute. Default is 50.

runDimRed A logical value indicating if dimenionality reduction should be computed. Default TRUE.

plotJackStraw A logical value indicating if JackStraw plot should be visualized. Default FALSE.

plotElbowPlot A logical value indicating if ElbowPlot should be visualized. Default TRUE.

plotHeatmaps A logical value indicating if heatmaps should be visualized. Default TRUE.

outputFile Specify the name of the generated output HTML file. If NULL then the output file name will be based on the name of the Rmarkdown template. Default NULL.

outputPath Specify the name of the output directory to save the rendered HTML file. If NULL the file is stored to the current working directory. Default NULL.

subtitle A character value specifying the subtitle to use in the report. Default NULL.

authors A character value specifying the names of the authors to use in the report. Default NULL.

showSession A logical value indicating if session information should be displayed or not. Default is FALSE.

pdf A logical value indicating if a pdf should also be generated for each figure in the report. Default is FALSE.

forceRun A logical value indicating if all computations previously computed should be re-calculated regardless if these computations are available in the input object. Default is TRUE.

Value

A SingleCellExperiment object with computations stored.
Generates an HTML report for Seurat Feature Selection and returns the SCE object with the results computed and stored inside the object.

**Description**

Generates an HTML report for Seurat Feature Selection and returns the SCE object with the results computed and stored inside the object.

**Usage**

```r
reportSeuratFeatureSelection(
  inSCE,             # Input SingleCellExperiment object.
  variable.features = 2000, # A numeric value indicating the number of top variable features to identify. Default 2000.
  runHVG = TRUE,     # A logical value indicating if the feature selection algorithm should be run or not. Default TRUE.
  plotHVG = TRUE,    # A logical value indicating if the mean-to-variance plot of the top variable feature should be visualized or not. Default TRUE.
  outputFile = NULL, # Specify the name of the generated output HTML file. If NULL then the output file name will be based on the name of the Rmarkdown template. Default NULL.
  outputPath = NULL, # Specify the name of the output directory to save the rendered HTML file. If NULL the file is stored to the current working directory. Default NULL.
  subtitle = NULL,   # A character value specifying the subtitle to use in the report. Default NULL.
  authors = NULL,    # A character value specifying the names of the authors to use in the report. Default NULL.
  showSession = FALSE, # A logical value indicating if session information should be displayed or not. Default FALSE.
  pdf = FALSE,       # Default TRUE.
  forceRun = TRUE)   # Default TRUE.
```

**Arguments**

- **inSCE**: Input `SingleCellExperiment` object.
- **variable.features**: A numeric value indicating the number of top variable features to identify. Default 2000.
- **runHVG**: A logical value indicating if the feature selection algorithm should be run or not. Default TRUE.
- **plotHVG**: A logical value indicating if the mean-to-variance plot of the top variable feature should be visualized or not. Default TRUE.
- **outputFile**: Specify the name of the generated output HTML file. If NULL then the output file name will be based on the name of the Rmarkdown template. Default NULL.
- **outputPath**: Specify the name of the output directory to save the rendered HTML file. If NULL the file is stored to the current working directory. Default NULL.
- **subtitle**: A character value specifying the subtitle to use in the report. Default NULL.
- **authors**: A character value specifying the names of the authors to use in the report. Default NULL.
- **showSession**: A logical value indicating if session information should be displayed or not. Default FALSE.
A logical value indicating if a pdf should also be generated for each figure in the report. Default is FALSE.

A logical value indicating if all computations previously computed should be re-calculated regardless if these computations are available in the input object. Default is TRUE.

Value

A `SingleCellExperiment` object with computations stored.

Usage

```r
reportSeuratMarkerSelection(
inSCE,
biological.group = NULL,
phenotype.groups = NULL,
selected.markers = NULL,
runMarkerSelection = TRUE,
plotMarkerSelection = TRUE,
umTopFeatures = 10,
outputFile = NULL,
outputPath = NULL,
subtitle = NULL,
authors = NULL,
showSession = FALSE,
pdf = FALSE
)
```

Arguments

- `inSCE` Input `SingleCellExperiment` object.
- `biological.group` A character value that specifies the name of the `colData()` column to use as the main biological group in the Seurat report for marker selection and grouping.
phenotype.groups
A character vector that specifies the names of the `colData()` columns to use for differential expression in addition to the `biological.group` parameter.

selected.markers
A character vector containing the user-specified gene symbols or feature names of marker genes that be used to generate gene plots in addition to the gene markers computed from differential expression.

runMarkerSelection
A logical value indicating if the marker selection computation should be run or not. Default TRUE.

plotMarkerSelection
A logical value indicating if the gene marker plots should be visualized or not. Default TRUE.

numTopFeatures
A numeric value indicating the number of top features to visualize in each group. Default 10.

outputFile
Specify the name of the generated output HTML file. If NULL then the output file name will be based on the name of the Rmarkdown template. Default NULL.

outputPath
Specify the name of the output directory to save the rendered HTML file. If NULL the file is stored to the current working directory. Default NULL.

subtitle
A character value specifying the subtitle to use in the report. Default NULL.

authors
A character value specifying the names of the authors to use in the report. Default NULL.

showSession
A logical value indicating if session information should be displayed or not. Default is FALSE.

pdf
A logical value indicating if a pdf should also be generated for each figure in the report. Default is FALSE.

Value
A `SingleCellExperiment` object with computations stored.

reportSeuratNormalization
Generates an HTML report for Seurat Normalization and returns the SCE object with the results computed and stored inside the object.

Description
Generates an HTML report for Seurat Normalization and returns the SCE object with the results computed and stored inside the object.
Usage

```r
reportSeuratNormalization(
inSCE, 
outputFile = NULL, 
outputPath = NULL, 
subtitle = NULL, 
authors = NULL, 
showSession = FALSE, 
pdf = FALSE, 
forceRun = TRUE
)
```

Arguments

- `inSCE`: Input `SingleCellExperiment` object previously passed through `reportSeuratRun()`.
- `outputFile`: Specify the name of the generated output HTML file. If NULL then the output file name will be based on the name of the Rmarkdown template. Default NULL.
- `outputPath`: Specify the name of the output directory to save the rendered HTML file. If NULL the file is stored to the current working directory. Default NULL.
- `subtitle`: A character value specifying the subtitle to use in the report. Default NULL.
- `authors`: A character value specifying the names of the authors to use in the report. Default NULL.
- `showSession`: A logical value indicating if session information should be displayed or not. Default is FALSE.
- `pdf`: A logical value indicating if a pdf should also be generated for each figure in the report. Default is FALSE.
- `forceRun`: A logical value indicating if all computations previously computed should be re-calculated regardless if these computations are available in the input object. Default is TRUE.

Value

A `SingleCellExperiment` object with computations stored.

---

**reportSeuratResults**  
*Generates an HTML report for Seurat Results (including Clustering & Marker Selection) and returns the SCE object with the results computed and stored inside the object.*

Description

Generates an HTML report for Seurat Results (including Clustering & Marker Selection) and returns the SCE object with the results computed and stored inside the object.
Usage

reportSeuratResults(
  inSCE,
  biological.group = NULL,
  phenotype.groups = NULL,
  selected.markers = NULL,
  clustering.resolution = 0.8,
  pc.count = 50,
  plotTSNE = TRUE,
  plotUMAP = TRUE,
  runClustering = TRUE,
  runMSClusters = TRUE,
  runMSBioGroup = TRUE,
  numTopFeatures = 10,
  outputFile = NULL,
  outputPath = NULL,
  subtitle = NULL,
  authors = NULL,
  showSession = FALSE,
  pdf = FALSE,
  forceRun = TRUE
)

Arguments

inSCE Input SingleCellExperiment object previously passed through reportSeuratRun().

biological.group A character value that specifies the name of the colData() column to use as the main biological group in the Seurat report for marker selection and grouping.

phenotype.groups A character vector that specifies the names of the colData() columns to use for differential expression in addition to the biological.group parameter.

selected.markers A character vector containing the user-specified gene symbols or feature names of marker genes that be used to generate gene plots in addition to the gene markers computed from differential expression.

clustering.resolution A numeric value indicating the user-specified final resolution to use with clustering. Default is 0.8.

pc.count A numeric value indicating the number of principal components to use in the analysis workflow. Default is 50.

plotTSNE A logical value indicating if TSNE plots should be visualized in the clustering section of the report. Default is TRUE.

plotUMAP A logical value indicating if UMAP plots should be visualized in the clustering section of the report. Default is TRUE.
runClustering A logical value indicating if Clustering should be run or not in the report. Default is TRUE. If FALSE, parameters plotTSNE and plotUMAP are also set to FALSE.

runMSClusters A logical value indicating if the marker selection section for identifying marker genes between clusters should be run and visualized in the report. Default TRUE.

runMSBioGroup A logical value indicating if the marker selection section for identifying marker genes between the biological.group parameter should be run and visualized in the report. Default TRUE.

numTopFeatures A numeric value indicating the number of top features to visualize in each group. Default 10.

outputFile Specify the name of the generated output HTML file. If NULL then the output file name will be based on the name of the Rmarkdown template. Default NULL.

outputPath Specify the name of the output directory to save the rendered HTML file. If NULL the file is stored to the current working directory. Default NULL.

subtitle A character value specifying the subtitle to use in the report. Default NULL.

authors A character value specifying the names of the authors to use in the report. Default NULL.

showSession A logical value indicating if session information should be displayed or not. Default is FALSE.

pdf A logical value indicating if a pdf should also be generated for each figure in the report. Default is FALSE.

forceRun A logical value indicating if all computations previously computed should be re-calculated regardless if these computations are available in the input object. Default is TRUE.

Value

A SingleCellExperiment object with computations stored.

---

**Description**

Generates an HTML report for Seurat Run (including Normalization, Feature Selection, Dimensionality Reduction & Clustering) and returns the SCE object with the results computed and stored inside the object.
Usage

```r
reportSeuratRun(
  inSCE, 
  biological.group = NULL, 
  phenotype.groups = NULL, 
  variable.features = 2000, 
  pc.count = 50, 
  runHVG = TRUE, 
  plotHVG = TRUE, 
  runDimRed = TRUE, 
  plotJackStraw = FALSE, 
  plotElbowPlot = TRUE, 
  plotHeatmaps = TRUE, 
  runClustering = TRUE, 
  plotTSNE = TRUE, 
  plotUMAP = TRUE, 
  minResolution = 0.3, 
  maxResolution = 1.5, 
  outputFile = NULL, 
  outputPath = NULL, 
  subtitle = NULL, 
  authors = NULL, 
  showSession = FALSE, 
  pdf = FALSE, 
  forceRun = TRUE
)
```

Arguments

- `inSCE` Input `SingleCellExperiment` object.
- `biological.group` A character value that specifies the name of the `colData()` column to use as the main biological group in the Seurat report for tSNE & UMAP visualization.
- `phenotype.groups` A character value that specifies the name of the `colData()` column to use as additional phenotype variables in the Seurat report for tSNE & UMAP visualization.
- `variable.features` A numeric value indicating the number of top variable genes to identify in the report. Default is 2000.
- `pc.count` A numeric value indicating the number of principal components to use in the analysis workflow. Default is 50.
- `runHVG` A logical value indicating if feature selection should be run in the report. Default is TRUE.
- `plotHVG` A logical value indicating if the top variable genes should be visualized through a mean-to-variance plot. Default is TRUE.
runDimRed A logical value indicating if PCA should be computed in the report. Default is TRUE.

plotJackStraw A logical value indicating if the JackStraw plot should be visualized for the principal components. Default is FALSE.

plotElbowPlot A logical value indicating if the ElbowPlot should be visualized for the principal components. Default is FALSE.

plotHeatmaps A logical value indicating if the Heatmaps should be visualized for the principal components. Default is FALSE.

runClustering A logical value indicating if Clustering should be run over multiple resolutions as defined by the minResolution and maxResolution parameters. Default is TRUE.

plotTSNE A logical value indicating if TSNE plot should be visualized for clusters. Default is TRUE.

plotUMAP A logical value indicating if UMAP plot should be visualized for clusters. Default is TRUE.

minResolution A numeric value indicating the minimum resolution to use for clustering. Default 0.3.

maxResolution A numeric value indicating the maximum resolution to use for clustering. Default 1.5.

outputFile Specify the name of the generated output HTML file. If NULL then the output file name will be based on the name of the Rmarkdown template. Default NULL.

outputPath Specify the name of the output directory to save the rendered HTML file. If NULL the file is stored to the current working directory. Default NULL.

subtitle A character value specifying the subtitle to use in the report. Default NULL.

authors A character value specifying the names of the authors to use in the report. Default NULL.

showSession A logical value indicating if session information should be displayed or not. Default is FALSE.

pdf A logical value indicating if a pdf should also be generated for each figure in the report. Default is FALSE.

forceRun A logical value indicating if all computations previously computed should be re-calculated regardless if these computations are available in the input object. Default is TRUE.

Value

A SingleCellExperiment object with computations stored.
Generates an HTML report for Seurat Scaling and returns the SCE object with the results computed and stored inside the object.

**Description**

Generates an HTML report for Seurat Scaling and returns the SCE object with the results computed and stored inside the object.

**Usage**

```r
reportSeuratScaling(  
inSCE,  
outputFile = NULL,  
outputPath = NULL,  
subtitle = NULL,  
authors = NULL,  
showSession = FALSE,  
pdf = FALSE,  
forceRun = TRUE  
)
```

**Arguments**

- `inSCE`  
  Input `SingleCellExperiment` object.

- `outputFile`  
  Specify the name of the generated output HTML file. If NULL then the output file name will be based on the name of the Rmarkdown template. Default NULL.

- `outputPath`  
  Specify the name of the output directory to save the rendered HTML file. If NULL the file is stored to the current working directory. Default NULL.

- `subtitle`  
  A character value specifying the subtitle to use in the report. Default NULL.

- `authors`  
  A character value specifying the names of the authors to use in the report. Default NULL.

- `showSession`  
  A logical value indicating if session information should be displayed or not. Default is FALSE.

- `pdf`  
  A logical value indicating if a pdf should also be generated for each figure in the report. Default is FALSE.

- `forceRun`  
  A logical value indicating if all computations previously computed should be re-calculated regardless if these computations are available in the input object. Default is TRUE.

**Value**

A `SingleCellExperiment` object with computations stored.
**retrieveSCEIndex**

Retrieve cell/feature index by giving identifiers saved in col/rowData

**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**

```r
retrieveSCEIndex(
  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 = NULL, # Character. In which column to search for features/cells in rowData/colData. Default NULL for search the rownames/colnames
  exactMatch = TRUE, # A logical scalar. Whether to only identify exact matches or to identify partial matches using grep. Default TRUE
  firstMatch = TRUE  # A logical scalar. Whether to only identify the first matches or to return all plausible matches. Default 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, Yichen Wang

**Examples**

```r
data(scExample, package = "singleCellTK")
retrieveSCEIndex(inSCE = sce, IDs = "ENSG00000205542", axis = "row")
```
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 or colData variable name. Indicates which sample each cell belongs to. Default NULL.

useAssay A string specifying which assay in the SCE to use. Default "counts"

lower See barcodeRanks for more information. Default 100.

fitBounds See barcodeRanks for more information. Default NULL.

df See barcodeRanks 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_inflection. Please refer to the documentation of barcodeRanks for details.

See Also

barcodeRanks, runDropletQC, plotBarcodeRankDropsResults

Examples

data(scExample, package = "singleCellTK")
sce <- runBarcodeRankDrops(inSCE = sce)
**runBBKNN**

Apply BBKNN batch effect correction method to SingleCellExperiment object

---

**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**

```r
runBBKNN(
  inSCE,  # Input SingleCellExperiment object
  useAssay = "logcounts",  # A single character indicating the name of the assay requiring batch correction. Default "logcounts".
  batch = "batch",  # A single character indicating a field in colData that annotates the batches of each cell; or a vector/factor with the same length as the number of cells. Default "batch".
  reducedDimName = "BBKNN",  # A single character. The name for the corrected low-dimensional representation. Will be saved to reducedDim(inSCE). Default "BBKNN".
  nComponents = 50L  # 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.
)
```

**Arguments**

- `inSCE`: Input `SingleCellExperiment` object
- `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 of each cell; or a vector/factor with the same length as the number of cells. Default "batch".
- `reducedDimName`: A single character. The name for the corrected low-dimensional representation. Will be saved to `reducedDim(inSCE)`. 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
runBcds

Find doublets/multiplets using bcds.

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

inSCE A SingleCellExperiment object.
sample Character vector or colData variable name. Indicates which sample each cell belongs to. Default NULL.
seed Seed for the random number generator, can be NULL. Default 12345.
ntop See bcds for more information. Default 500.
srat See bcds for more information. Default 1.
verb See bcds for more information. Default FALSE.
retRes See bcds for more information. Default FALSE.
nmax See bcds for more information. Default "tune".

Examples

## Not run:
data('sceBatches', package = 'singleCellTK')
logcounts(sceBatches) <- log1p(counts(sceBatches))
sceBatches <- runBBKNN(sceBatches, useAssay = "logcounts",
  nComponents = 10)

## End(Not run)
runCellQC

- `varImp`: See `bcds` for more information. Default `FALSE`.
- `estNdbl`: See `bcds` for more information. Default `FALSE`.
- `useAssay`: A string specifying which assay in `inSCE` to use. Default "counts"

**Details**

When the argument `sample` is specified, `bcds` will be run on cells from each sample separately. If `sample = NULL`, then all cells will be processed together.

**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.

**See Also**

`bcds`, `plotBcdsResults`, `runCellQC`

**Examples**

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

---

**Description**

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

**Usage**

```r
runCellQC(
inSCE,  
algorithms = c("QCMetrics", "scDb1Finder", "cxds", "bcds", "cxds_bcds_hybrid",  
                 "decontX", "decontX_bg", "soupX", "soupX_bg"),  
    sample = NULL,  
    collectionName = NULL,  
    geneSetList = NULL,  
    geneSetListLocation = "rownames",  
    geneSetCollection = NULL,  
    mitoRef = "human",  
    mitoIDType = "ensembl",  
    mitoPrefix = "MT-",  
    mitoID = NULL,
)```

mitoGeneLocation = "rownames",
useAssay = "counts",
background = NULL,
bgAssayName = NULL,
bgBatch = NULL,
seed = 12345,
paramsList = NULL
)

Arguments

inSCE A SingleCellExperiment object.
algorithms Character vector. Specify which QC algorithms to run. Available options are "QCMetrics", "scrublet", "doubletFinder", "scDblFinder", "cxds", "bcds", "cxds_bcds_hybrid", "decontX" and "soupX".
sample Character vector. Indicates which sample each cell belongs to. Algorithms will be run on cells from each sample separately.
collectionName Character. Name of a GeneSetCollection obtained by using one of the import-GeneSet* functions. Default NULL.
geneSetList See runPerCellQC. Default NULL.
geneSetListLocation See runPerCellQC. Default NULL.
geneSetCollection See runPerCellQC. Default NULL.
mitoRef, mitoIDType, mitoPrefix, mitoID, mitoGeneLocation Arguments used to import mitochondrial genes and quantify their expression. Please see runPerCellQC for detailed information.
useAssay A string specifying which assay contains the count matrix for cells.
background A SingleCellExperiment with the matrix located in the assay slot under bgAssayName. It should have the same structure as inSCE except it contains the matrix of empty droplets instead of cells. When supplied, empirical distribution of transcripts from these empty droplets will be used as the contamination distribution. It is only used in algorithms "decontX" and "soupX". Default NULL.
bgAssayName Character. Name of the assay to use if background is a SingleCellExperiment. If NULL, the function will use the same value as useAssay. It is only used in algorithms "decontX" and "soupX". Default is NULL.
bgBatch Batch labels for background. If background is a SingleCellExperiment object, this can be a single character specifying a name that can be found in colData(background) to directly use the barcode annotation. Its unique values should be the same as those in sample, such that each batch of cells have their corresponding batch of empty droplets as background, pointed by this parameter. It is only used in algorithms "decontX" and "soupX". Default to NULL.
seed Seed for the random number generator. Default 12345.
paramsList A list containing parameters for QC functions. Default NULL.
**runClusterSummaryMetrics**

**Value**

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

**Examples**

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

**runClusterSummaryMetrics**

*Run Cluster Summary Metrics*

**Description**

Calculates the mean expression of percent of cells that express the given genes for each cluster

**Usage**

```r
runClusterSummaryMetrics(
  inSCE, 
  useAssay = "logcounts", 
  featureNames, 
  displayName = NULL, 
  groupNames = "cluster", 
  scale = FALSE )
```

**Arguments**

- **inSCE**: The single cell experiment to use.
- **useAssay**: The assay to use.
- **featureNames**: A string or vector of strings with each gene to aggregate.
- **displayName**: A string that is the name of the column used for genes.
- **groupNames**: The name of a colData entry that can be used as groupNames.
- **scale**: Option to scale the data. Default: FALSE. Selected assay will not be scaled.

**Value**

A dataframe with mean expression and percent of cells in cluster that express for each cluster.
runComBatSeq

Examples

```r
data("scExample")
runClusterSummaryMetrics(inSCE = sge, useAssay = "counts", featureNames = c("B2M", "MALAT1"),
                       displayName = "feature_name", groupNames = "type")
```

runComBatSeq: Apply ComBat-Seq batch effect correction method to SingleCellExperiment object

Description

The ComBat-Seq 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

```r
runComBatSeq(
  inSCE,
  useAssay = "counts",
  batch = "batch",
  covariates = NULL,
  bioCond = NULL,
  useSVA = FALSE,
  assayName = "ComBatSeq",
  shrink = FALSE,
  shrinkDisp = FALSE,
  nGene = NULL
)
```

Arguments

- **inSCE**: Input `SingleCellExperiment` object
- **useAssay**: A single character indicating the name of the assay requiring batch correction. Default "counts".
- **batch**: A single character indicating a field in `colData` that annotates the batches. Default "batch".
- **covariates**: A character vector indicating the fields in `colData` that annotates other covariates, such as the cell types. Default NULL.
- **bioCond**: A single character indicating a field in `colData` that annotates the biological conditions. Default NULL.
- **useSVA**: A logical scalar. Whether to estimate surrogate variables and use them as an empirical control. Default FALSE.
runCxds

assayName A single character. The name for the corrected assay. Will be saved to assay. Default "ComBat".

shrink A logical scalar. Whether to apply shrinkage on parameter estimation. Default FALSE.

shrinkDisp A logical scalar. Whether to apply shrinkage on dispersion. Default FALSE.

nGene An integer. Number of random genes to use in empirical Bayes estimation, only useful when shrink is set to TRUE. Default NULL.

Details

For the parameters covariates and useSVA, when the cell type information is known, it is recommended to specify the cell type annotation to the argument covariates; if the cell types are unknown but expected to be balanced, it is recommended to run with default settings, yet informative covariates could still be useful. If the cell types are unknown and are expected to be unbalanced, it is recommended to set useSVA to TRUE.

Value

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

Examples

data('sceBatches', package = 'singleCellTK')
sceBatches <- sample(sceBatches, 40)
# Cell type known
sceBatches <- runComBatSeq(sceBatches, "counts", "batch",
    covariates = "cell_type",
    assayName = "ComBat_cell_seq")
# Cell type unknown but balanced
#sceBatches <- runComBatSeq(sceBatches, "counts", "batch",
#    assayName = "ComBat_seq")
# Cell type unknown and unbalanced
#sceBatches <- runComBatSeq(sceBatches, "counts", "batch",
#    useSVA = TRUE,
#    assayName = "ComBat_sva_seq")

runCxds

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

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

Arguments

inSCE A SingleCellExperiment object.
sample Character vector or colData variable name. Indicates which sample each cell belongs to. Default NULL.
seed Seed for the random number generator, can be NULL. 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. Default "counts"

Details

When the argument sample is specified, cxds will be run on cells from each sample separately. If sample = NULL, then all cells will be processed together.

Value

A SingleCellExperiment object with cxds output appended to the colData slot. The columns include cxds_score and optionally cxds_call.

See Also

cxds, plotCxdsResults, runCellQC

Examples

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

Find doublets/multiplets using cxds_bcds_hybrid.

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
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 csds and bcds. Default 500.
cxdsArgs See cxds_bcds_hybrid for more information. Default NULL.
bcdsArgs See cxds_bcds_hybrid for more information. Default NULL.
verb See cxds_bcds_hybrid for more information. Default FALSE.
estNdbl See cxds_bcds_hybrid for more information. Default FALSE.
force See cxds_bcds_hybrid for more information. Default FALSE.
useAssay A string specifying which assay in the SCE to use.

Value
A SingleCellExperiment object with cxds_bcds_hybrid output appended to the colData slot. The columns include hybrid_score and optionally hybrid_call. Please refer to the documentation of cxds_bcds_hybrid for details.
Examples

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

runDEAnalysis

Perform differential expression analysis on SCE object

Description

Perform differential expression analysis on SCE object

Usage

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

runDESeq2(
inSCE,
useAssay = "counts",
useReducedDim = NULL,
index1 = NULL,
index2 = NULL,
class = NULL,
classGroup1 = NULL,
classGroup2 = NULL,
analysisName,
groupName1,
groupName2,
covariates = NULL,
fullReduced = TRUE,
onlyPos = FALSE,
log2fcThreshold = NULL,
fdrThreshold = NULL,
minGroup1MeanExp = NULL,
maxGroup2MeanExp = NULL,
minGroup1ExprPerc = NULL,
maxGroup2ExprPerc = NULL,
overwrite = FALSE,
verbose = TRUE
)

runLimmaDE(
inSCE,
useAssay = "logcounts",
useReducedDim = NULL,
index1 = NULL,
index2 = NULL,
runDEAnalysis

  class = NULL,
  classGroup1 = NULL,
  classGroup2 = NULL,
  analysisName,
  groupName1,
  groupName2,
  covariates = NULL,
  onlyPos = FALSE,
  log2fcThreshold = NULL,
  fdrThreshold = NULL,
  minGroup1MeanExp = NULL,
  maxGroup2MeanExp = NULL,
  minGroup1ExprPerc = NULL,
  maxGroup2ExprPerc = NULL,
  overwrite = FALSE,
  verbose = TRUE

)

runANOVA(
  inSCE,
  useAssay = "logcounts",
  useReducedDim = NULL,
  index1 = NULL,
  index2 = NULL,
  class = NULL,
  classGroup1 = NULL,
  classGroup2 = NULL,
  analysisName,
  groupName1,
  groupName2,
  covariates = NULL,
  onlyPos = FALSE,
  log2fcThreshold = NULL,
  fdrThreshold = NULL,
  minGroup1MeanExp = NULL,
  maxGroup2MeanExp = NULL,
  minGroup1ExprPerc = NULL,
  maxGroup2ExprPerc = NULL,
  overwrite = FALSE,
  verbose = TRUE
)

runMAST(
  inSCE,
  useAssay = "logcounts",
  useReducedDim = NULL,
  index1 = NULL,
  index2 = NULL,

class = NULL,
classGroup1 = NULL,
classGroup2 = NULL,
analysisName,
groupName1,
groupName2,
covariates = NULL,
onlyPos = FALSE,
log2fcThreshold = NULL,
fdrThreshold = NULL,
minGroup1MeanExp = NULL,
maxGroup2MeanExp = NULL,
minGroup1ExprPerc = NULL,
maxGroup2ExprPerc = NULL,
overwrite = FALSE,
check_sanity = TRUE,
verbose = TRUE
)

runWilcox(
inSCE,
useAssay = "logcounts",
useReducedDim = NULL,
index1 = NULL,
index2 = NULL,
class = NULL,
classGroup1 = NULL,
classGroup2 = NULL,
analysisName,
groupName1,
groupName2,
covariates = NULL,
onlyPos = FALSE,
log2fcThreshold = NULL,
fdrThreshold = NULL,
minGroup1MeanExp = NULL,
maxGroup2MeanExp = NULL,
minGroup1ExprPerc = NULL,
maxGroup2ExprPerc = NULL,
overwrite = FALSE,
verbose = TRUE
)

Arguments
method
Character. Specify which method to use when using runDEAnalysis(). Choose from "wilcox", "MAST", "DESeq2", "Limma", "ANOVA". Default "wilcox".

... Arguments to pass to specific methods when using the generic runDEAnalysis().
runDEAnalysis

inSCE

useAssay

useReducedDim

index1

index2

class

classGroup1

classGroup2

analysisName

groupName1

groupName2

covariates

fullReduced

onlyPos

log2fcThreshold

fdrThreshold

minGroup1MeanExp

maxGroup2MeanExp

minGroup1ExprPerc

maxGroup2ExprPerc

overwrite

verbose

check_sanity

SingleCellExperiment inherited object.

character. A string specifying which assay to use for the DE regression. Ignored when useReducedDim is specified. Default "counts" for DESeq2, "logcounts" for other methods.

character. A string specifying which reducedDim to use for DE analysis. Will treat the dimensions as features. Default NULL.

Any type of indices that can subset a SingleCellExperiment inherited object by cells. Specifies which cells are of interests. Default NULL.

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.

A vector/factor with ncol(inSCE) elements, or a character scalar that specifies a column name of colData(inSCE). Default NULL.

A vector specifying which "levels" given in class are of interests. Default NULL.

A vector specifying which "levels" given in class is the control group against those specified by classGroup1. If NULL when using annotation specification, classGroup1 cells will be compared with all other cells.

A character scalar naming the DEG analysis. Required

A character scalar naming the group of interests. Required.

A character scalar naming the control group. Required.

A character vector of additional covariates to use when building the model. All covariates must exist in names(colData(inSCE)). Default NULL.

Logical, DESeq2 only argument. Whether to apply LRT (Likelihood ratio test) with a 'full' model. Default TRUE.

Whether to only output DEG with positive log2_FC value. Default FALSE.

Only output DEGs with the absolute values of log2FC greater than this value. Default NULL.

Only output DEGs with FDR value less than this value. Default NULL.

Only output DEGs with mean expression in group1 greater then this value. Default NULL.

Only output DEGs with mean expression in group2 less then this value. Default NULL.

Only output DEGs expressed in greater then this fraction of cells in group1. Default NULL.

Only output DEGs expressed in less then this fraction of cells in group2. Default NULL.

A logical scalar. Whether to overwrite result if exists. Default FALSE.

A logical scalar. Whether to show messages. Default TRUE.

Logical, MAST only argument. Whether to perform MAST’s sanity check to see if the counts are logged. Default TRUE.
Details

SCTK provides Limma, MAST, DESeq2, ANOVA and Wilcoxon test for differential expression analysis, where DESeq2 expects non-negative integer assay input while others expect logcounts.

Condition specification allows two methods: 1. Index level selection. Only use arguments index1 and index2. 2. Annotation level selection. Only use arguments class, classGroup1 and classGroup2.

Value

The input SingleCellExperiment object, where metadata(inSCE)$diffExp is updated with a list named by analysisName, with elements of:

$groupName -- the naming of the two conditions
$useAssay, $useReducedDim -- the matrix name that was used for calculation
$select -- the cell selection indices (logical) for each condition
$result -- a data.frame of the DEGs table
$method -- the method used

See Also

See plotDEGHeatmap, plotDEGRegression, plotDEGViolin and plotDEGVolcano for visualization method after running DE analysis.

Examples

data(scExample, package = "singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
sce <- scaterlogNormCounts(sce, assayName = "logcounts")
sce <- runDEAnalysis(method = "Limma", inSCE = sce, groupName1 = "group1",
                    groupName2 = "group2", index1 = seq(20), index2 = seq(21,40),
                    analysisName = "Limma")

runDecontX

Detecting contamination with DecontX.

Description

A wrapper function for decontX. Identify potential contamination from experimental factors such as ambient RNA.
runDecontX

Usage

runDecontX(
  inSCE,
  sample = NULL,
  useAssay = "counts",
  background = NULL,
  bgAssayName = NULL,
  bgBatch = NULL,
  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 A single character specifying a name that can be found in colData(inSCE) to
directly use the cell annotation; or a character vector with as many elements as
cells to 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'.

background A SingleCellExperiment with the matrix located in the assay slot under bgAssayName.
It should have the same structure as inSCE except it contains the matrix of empty
droplets instead of cells. When supplied, empirical distribution of transcripts
from these empty droplets will be used as the contamination distribution. De-
fault NULL.

bgAssayName Character. Name of the assay to use if background is a SingleCellExperiment.
If NULL, the function will use the same value as useAssay. Default is NULL.

bgBatch Batch labels for background. If background is a SingleCellExperiment ob-
ject, this can be a single character specifying a name that can be found in
colData(background) to directly use the barcode annotation; or a numeric /
character vector that has as many elements as barcodes to indicate which sam-
ple each barcode belongs to. Its unique values should be the same as those in
sample, such that each batch of cells have their corresponding batch of empty
droplets as background, pointed by this parameter. Default to NULL.

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' pack-
age will be used to further reduce the dataset to 2 dimenions and the 'dbscan'
runDecontX function from the 'dbscan' package will be used to identify clusters of broad cell types. Default NULL.


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 estimateDelta = TRUE, 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 delta in each iteration. If estimateDelta = FALSE, then delta is fixed with these values for the entire inference procedure. Fixing delta 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 c(10, 10).

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.


varGenes Integer. The number of variable genes to use in dimensionality reduction before clustering. Variability is calculated using modelGeneVar 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 with_seed. For reproducibility, a default value of 12345 is used. If NULL, no calls to with_seed 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[,sample(ncol(sce),20)])
**runDimReduce**  
*Generic Wrapper function for running dimensionality reduction*

---

**Description**

Generic Wrapper function for running dimensionality reduction

**Usage**

```r
runDimReduce(
  inSCE,
  method = c("scaterPCA", "seuratPCA", "seuratICA", "scanpyPCA", "rTSNE", "seuratTSNE", "scaterUMAP", "seuratUMAP", "scanpyUMAP", "scanpyTSNE"),
  useAssay = NULL,
  useReducedDim = NULL,
  useAltExp = NULL,
  reducedDimName = method,
  nComponents = 20,
  useFeatureSubset = NULL,
  scale = FALSE,
  seed = 12345,
  ...
)
```

**Arguments**

- **inSCE**: Input `SingleCellExperiment` object.
- **method**: One from "scaterPCA", "seuratPCA", "seuratICA", "rTSNE", "seuratTSNE", "scaterUMAP", "seuratUMAP", "scanpyPCA", "scanpyUMAP" and "scanpyTSNE".
- **useAssay**: Assay to use for computation. If `useAltExp` is specified, `useAssay` has to exist in `assays(altExp(inSCE, useAltExp))`. Default "counts".
- **useReducedDim**: The low dimension representation to use for embedding computation. Default NULL.
- **useAltExp**: The subset to use for computation, usually for the selected variable features. Default NULL.
- **reducedDimName**: The name of the result matrix. Required.
- **nComponents**: Specify the number of dimensions to compute with the selected method in case of PCA/ICA and the number of components to use in the case of TSNE/UMAP methods.
- **useFeatureSubset**: Subset of feature to use for dimension reduction. A character string indicating a `rowData` variable that stores the logical vector of HVG selection, or a vector that can subset the rows of `inSCE`. Default NULL.
- **scale**: Logical scalar, whether to standardize the expression values. Default TRUE.
runDoubletFinder

seed           Random seed for reproducibility of results. Default NULL will use global seed in use by the R environment.
...            The other arguments for running a specific algorithm. Please refer to the one you use.

Details

Wrapper function to run one of the available dimensionality reduction algorithms integrated within SCTK from scaterPCA, runSeuratPCA, runSeuratICA, runTSNE, runSeuratTSNE, runUMAP and runSeuratUMAP. Users can use an assay by specifying useAssay, use the assay in an altExp by specifying both useAltExp and useAssay, or use a low-dimensionality representation by specifying useReducedDim.

Value

The input SingleCellExperiment object with reducedDim updated with the result.

Examples

data(scExample, package = "singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
sce <- runNormalization(sce, useAssay = "counts",
                        outAssayName = "logcounts",
                        normalizationMethod = "logNormCounts")
sce <- runDimReduce(inSCE = sce, method = "scaterPCA",
                    useAssay = "logcounts", scale = TRUE,
                    reducedDimName = "PCA")

runDoubletFinder       Generates a doublet score for each cell via doubletFinder

Description

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

Usage

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

Run comprehensive droplet QC

#### Description

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

#### Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inSCE</td>
<td>A <code>SingleCellExperiment</code> object.</td>
</tr>
<tr>
<td>sample</td>
<td>Character vector or colData variable name. Indicates which sample each cell belongs to. Default NULL.</td>
</tr>
<tr>
<td>useAssay</td>
<td>A string specifying which assay in the SCE to use. Default &quot;counts&quot;.</td>
</tr>
<tr>
<td>seed</td>
<td>Seed for the random number generator, can be set to NULL. Default 12345.</td>
</tr>
<tr>
<td>seuratPcs</td>
<td>Numeric vector. The PCs used in seurat function to determine number of clusters. Default 1:15.</td>
</tr>
<tr>
<td>seuratRes</td>
<td>Numeric vector. The resolution parameter used in Seurat, which adjusts the number of clusters determined via the algorithm. Default 1.5.</td>
</tr>
<tr>
<td>formationRate</td>
<td>Doublet formation rate used within algorithm. Default 0.075.</td>
</tr>
<tr>
<td>nCores</td>
<td>Number of cores used for running the function. Default NULL.</td>
</tr>
<tr>
<td>verbose</td>
<td>Boolean. Whether to print messages from Seurat and DoubletFinder. Default FALSE.</td>
</tr>
</tbody>
</table>

#### Value

A `SingleCellExperiment` object containing the `doublet_finder_doublet_score` variable in colData slot.

#### See Also

`runCellQC`, `plotDoubletFinderResults`

#### Examples

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

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 Identify empty droplets using emptyDrops.

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.
runEmptyDrops

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  A SingleCellExperiment object. Must contain a raw counts matrix before empty droplets have been removed.
sample Character vector or colData variable name. Indicates which sample each cell belongs to. Default NULL.
useAssay A string specifying which assay in the SCE to use. Default "counts"
lower See emptyDrops for more information. Default 100.
niters See emptyDrops for more information. Default 10000.
testAmbient See emptyDrops for more information. Default FALSE.
ignore See emptyDrops for more information. Default NULL.
alpha See emptyDrops for more information. Default NULL.
retain See emptyDrops for more information. Default NULL.
barcodeArgs See emptyDrops for more information. Default list().
BPPARAM See emptyDrops for more information. Default BiocParallel::SerialParam().

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.

See Also

runDropletQC, plotEmptyDropsResults, plotEmptyDropsScatter

Examples

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

Run EnrichR on SCE object

Description
Run EnrichR on SCE object

Usage
runEnrichR(
  inSCE,
  features,
  analysisName,
  db = NULL,
  by = "rownames",
  featureName = NULL
)

Arguments

inSCE A SingleCellExperiment object.
features Character vector, selected genes for enrichment analysis.
analysisName A string that identifies each specific analysis.
db Character vector. Selected database name(s) from the enrichR database list. If NULL then EnrichR will be run on all the available databases on the enrichR database. See details. Default NULL
by Character. From where should we find the features? "rownames" for from rownames(inSCE), otherwise, from a column of feature metadata (rowData(inSCE)[[by]]). See details. Default "rownames".
featureName Character. Indicates the actual feature identifiers to be passed to EnrichR. Can be "rownames", a column in feature metadata (rowData(inSCE)[[featureName]]), or a character vector with its length equals to nrow(inSCE). See details. Default "rownames".

Details
EnrichR works by querying the specified features to its online databases, thus it requires the Internet connection.

Available db options could be shown by running enrichR::listEnrichrDbs()$libraryName

This function checks for the existence of features in the SCE object. When features do not have a match in rownames(inSCE), users may try to specify by to pass the check.

EnrichR expects gene symbols/names as the input (i.e. Ensembl ID might not work). When specified features are not qualified for this, users may try to specify featureName to change the identifier type to pass to EnrichR.
runFastMNN

Value

Updates inSCE metadata with a data.frame of enrichment terms overlapping in the respective databases along with p-values, z-scores etc.

See Also

generateEnrichRResult

Examples

data("mouseBrainSubsetSCE")
if (Biobase:testBioCConnection()) {
  mouseBrainSubsetSCE <- runEnrichR(mouseBrainSubsetSCE, features = "Cmtm5",
    db = "GO_Cellular_Component_2017",
    analysisName = "analysis1")
}

runFastMNN

---

**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**

```r
runFastMNN(
  inSCE,  
  useAssay = "logcounts", 
  useReducedDim = NULL, 
  batch = "batch", 
  reducedDimName = "fastMNN", 
  k = 20, 
  propK = NULL, 
  ndist = 3, 
  minBatchSkip = 0, 
  cosNorm = TRUE, 
  nComponents = 50, 
  weights = NULL, 
  BPPARAM = BiocParallel::SerialParam()
)
```
Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>inSCE</td>
<td>Input SingleCellExperiment object</td>
<td></td>
</tr>
<tr>
<td>useAssay</td>
<td>A single character indicating the name of the assay requiring batch correction. Default &quot;logcounts&quot;.</td>
<td></td>
</tr>
<tr>
<td>useReducedDim</td>
<td>A single character indicating the dimension reduction used for batch correction. Will ignore useAssay when using. Default NULL.</td>
<td></td>
</tr>
<tr>
<td>batch</td>
<td>A single character indicating a field in colData that annotates the batches of each cell; or a vector/factor with the same length as the number of cells. Default &quot;batch&quot;.</td>
<td></td>
</tr>
<tr>
<td>reducedDimName</td>
<td>A single character. The name for the corrected low-dimensional representation. Default &quot;fastMNN&quot;.</td>
<td></td>
</tr>
<tr>
<td>k</td>
<td>An integer scalar specifying the number of nearest neighbors to consider when identifying MNNs. See &quot;See Also&quot;. Default 20.</td>
<td></td>
</tr>
<tr>
<td>propK</td>
<td>A numeric scalar in (0, 1) specifying the proportion of cells in each dataset to use for mutual nearest neighbor searching. See &quot;See Also&quot;. Default NULL.</td>
<td></td>
</tr>
<tr>
<td>ndist</td>
<td>A numeric scalar specifying the threshold beyond which neighbours are to be ignored when computing correction vectors. See &quot;See Also&quot;. Default 3.</td>
<td></td>
</tr>
<tr>
<td>minBatchSkip</td>
<td>Numeric scalar specifying the minimum relative magnitude of the batch effect, below which no correction will be performed at a given merge step. See &quot;See Also&quot;. Default 0.</td>
<td></td>
</tr>
<tr>
<td>cosNorm</td>
<td>A logical scalar indicating whether cosine normalization should be performed on useAssay prior to PCA. See &quot;See Also&quot;. Default TRUE.</td>
<td></td>
</tr>
<tr>
<td>nComponents</td>
<td>An integer scalar specifying the number of dimensions to produce. See &quot;See Also&quot;. Default 50.</td>
<td></td>
</tr>
<tr>
<td>weights</td>
<td>The weighting scheme to use. Passed to multiBatchPCA. Default NULL.</td>
<td></td>
</tr>
<tr>
<td>BPPARAM</td>
<td>A BiocParallelParam object specifying whether the SVD should be parallelized.</td>
<td></td>
</tr>
</tbody>
</table>

Value

The input SingleCellExperiment object with reducedDim(inSCE, reducedDimName) updated.

References

Lun ATL, et al., 2016

See Also

fastMNN for using useAssay, and reducedMNN for using useReducedDim

Examples

data('sceBatches', package = 'singleCellTK')
logcounts(sceBatches) <- log1p(counts(sceBatches))
sceCorr <- runFastMNN(sceBatches, useAssay = 'logcounts')
runFeatureSelection  Run Variable Feature Detection Methods

Description

Wrapper function to run all of the feature selection methods integrated within the singleCellTK package including three methods from Seurat ("vst", "mean.var.plot" or dispersion) and the Scran modelGeneVar method.

This function does not return the names of the variable features but only computes the metrics, which will be stored in the rowData slot. To set a HVG list for downstream use, users should call setTopHVG after computing the metrics. To get the names of the variable features, users should call getTopHVG function after computing the metrics.

Usage

runFeatureSelection(
  inSCE,
  useAssay,
  method = c("vst", "mean.var.plot", "dispersion", "modelGeneVar", "cell_ranger")
)

Arguments

inSCE Input SingleCellExperiment object.

useAssay Specify the name of the assay that should be used. Should use raw counts for "vst" method, or a normalized assay for other methods.

method Specify the method to use for variable gene selection. Options include "vst", "mean.var.plot" or "dispersion" from Seurat and "modelGeneVar" from Scran.

Value

The input SingleCellExperiment object that contains the computed statistics in the rowData slot

See Also

runModelGeneVar, runSeuratFindHVG, getTopHVG, plotTopHVG

Examples

data("mouseBrainSubsetSCE", package = "singleCellTK")
mouseBrainSubsetSCE <- runFeatureSelection(mouseBrainSubsetSCE, "logcounts", "modelGeneVar")
runFindMarker  
Find the marker gene set for each cluster

Description
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. \texttt{runFindMarker} will be deprecated in the future.

Usage

\begin{verbatim}
runFindMarker(
inSCE,
  useAssay = "logcounts",
  useReducedDim = NULL,
  method = c("wilcox", "MAST", "DESeq2", "Limma", "ANOVA"),
  cluster = "cluster",
  covariates = NULL,
  log2fcThreshold = NULL,
  fdrThreshold = 0.05,
  minClustExprPerc = NULL,
  maxCtrlExprPerc = NULL,
  minMeanExpr = NULL,
  detectThresh = 0
)
\end{verbatim}

\begin{verbatim}
findMarkerDiffExp(
inSCE,
  useAssay = "logcounts",
  useReducedDim = NULL,
  method = c("wilcox", "MAST", "DESeq2", "Limma", "ANOVA"),
  cluster = "cluster",
  covariates = NULL,
  log2fcThreshold = NULL,
  fdrThreshold = 0.05,
  minClustExprPerc = NULL,
  maxCtrlExprPerc = NULL,
  minMeanExpr = NULL,
  detectThresh = 0
)
\end{verbatim}

Arguments

\begin{itemize}
  \item \texttt{inSCE} \hspace{1cm} SingleCellExperiment inherited object.
  \item \texttt{useAssay} \hspace{1cm} character. A string specifying which assay to use for the MAST calculations. Default "logcounts".
\end{itemize}


**useReducedDim** character. A string specifying which reducedDim to use for MAST calculations. Set useAssay to NULL when using. Required.

**method** A single character for specific differential expression analysis method. Choose from 'wilcox', 'MAST', 'DESeq2', 'Limma', and 'ANOVA'. Default "wilcox".

**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 NULL

**minClustExprPerc** A numeric scalar. The minimum cutoff of the percentage of cells in the cluster of interests that expressed the marker gene. From 0 to 1. Default NULL.

**maxCtrlExprPerc** A numeric scalar. The maximum cutoff of the percentage of cells out of the cluster (control group) that expressed the marker gene. From 0 to 1. Default NULL.

**minMeanExpr** A numeric scalar. The minimum cutoff of the mean expression value of the marker in the cluster of interests. Default NULL.

**detectThresh** A numeric scalar, above which a matrix value will be treated as expressed when calculating cluster/control expression percentage. Default 0.

**Details**

The returned marker table, in the metadata slot, consists of 8 columns: "Gene", "Log2_FC", "Pvalue", "FDR", "cluster", "clusterExprPerc", "ControlExprPerc" and "clusterAveExpr".

"clusterExprPerc" is the fraction of cells, that has marker value (e.g. gene expression counts) larger than detectThresh, in the cell population of the cluster. As for each cluster, we set all cells out of this cluster as control. Similarly, "ControlExprPerc" is the fraction of cells with marker value larger than detectThresh in the control cell group.

**Value**

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

**See Also**

runDEAnalysis, getFindMarkerTopTable, plotFindMarkerHeatmap
Examples

```r
data("mouseBrainSubsetSCE", package = "singleCellTK")
mouseBrainSubsetSCE <- runFindMarker(mouseBrainSubsetSCE,
  useAssay = "logcounts",
  cluster = "level1class")
```

runGSVA

*Run GSVA analysis on a SingleCellExperiment object*

Description

Run GSVA analysis on a SingleCellExperiment object

Usage

```r
runGSVA(
  inSCE, 
  useAssay = "logcounts", 
  resultNamePrefix = NULL, 
  geneSetCollectionName, 
  ... 
)
```

Arguments

- **inSCE**: Input SingleCellExperiment object.
- **useAssay**: Indicate which assay to use. The default is "logcounts"
- **resultNamePrefix**: Character. Prefix to the name the GSVA results which will be stored in the reducedDim slot of inSCE. The names of the output matrix will be `resultNamePrefix_Scores`. If this parameter is set to NULL, then "GSVA_geneSetCollectionName_" will be used. Default NULL.
- **geneSetCollectionName**: Character. The name of the gene set collection to use.
- **...**: Parameters to pass to gsva()

Value

A SingleCellExperiment object with pathway activity scores from GSVA stored in reducedDim as GSVA_geneSetCollectionName_Scores.
Examples

data(scExample, package = "singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
sce <- scaterlogNormCounts(sce, assayName = "logcounts")
gs1 <- rownames(sce)[seq(10)]
gs2 <- rownames(sce)[seq(11,20)]
gs <- list("geneset1" = gs1, "geneset2" = gs2)

sce <- importGeneSetsFromList(inSCE = sce, geneSetList = gs,
by = "rownames")
sce <- runGSAV(inSCE = sce,
geneSetCollectionName = "GeneSetCollection",
useAssay = "logcounts")

runHarmony

Apply Harmony batch effect correction method to SingleCellExperiment object

Description

Harmony is an algorithm that projects cells into a shared embedding in which cells group by cell type rather than dataset-specific conditions.

Usage

runHarmony(
  inSCE,
  useAssay = NULL,
  useReducedDim = NULL,
  batch = "batch",
  reducedDimName = "HARMONY",
  nComponents = 50,
  lambda = 0.1,
  theta = 5,
  sigma = 0.1,
  nIter = 10,
  seed = 12345,
  verbose = TRUE,
  ...
)

Arguments

inSCE Input SingleCellExperiment object
useAssay A single character indicating the name of the assay requiring batch correction. Default NULL. It is recommended to use a reducedDim such as PCA through the ‘useReducedDim’ parameter of this function.
useReducedDim  A single character indicating the name of the reducedDim to be used. It is recommended to use a reducedDim instead of a full assay as using an assay might cause the algorithm to not converge and throw error. Specifying this will ignore useAssay. Default NULL.

batch  A single character indicating a field in colData that annotates the batches of each cell; or a vector/factor with the same length as the number of cells. Default "batch".

reducedDimName  A single character. The name for the corrected low-dimensional representation. Will be saved to reducedDim(inSCE). Default "HARMONY".

nComponents  An integer. The number of PCs to use and generate. Default 50L.

lambda  A Numeric scalar. Ridge regression penalty parameter. Must be strictly positive. Smaller values result in more aggressive correction. Default 0.1.

theta  A Numeric scalar. Diversity clustering penalty parameter. Larger values of theta result in more diverse clusters. theta=0 does not encourage any diversity. Default 5.

sigma  A Numeric scalar. Width of soft kmeans clusters. Larger values of sigma result in cells assigned to more clusters. Smaller values of sigma make soft kmeans cluster approach hard clustering. Default 0.1.

nIter  An integer. The max number of iterations to perform. Default 10L.

seed  Set seed for reproducibility. Default is 12345.

verbose  Whether to print progress messages. Default TRUE.

...  Other arguments passed to HarmonyMatrix. See details.

Details

Since some of the arguments of HarmonyMatrix is controlled by this wrapper function. The additional arguments users can work with only include: nclust, tau, block.size, max.iter.cluster, epsilon.cluster, epsilon.harmony, plot_convergence, reference_values and cluster_prior.

Value

The input SingleCellExperiment object with reducedDim(inSCE, reducedDimName) updated.

References

Ilya Korsunsky, et al., 2019

Examples

data('sceBatches', package = 'singleCellTK')
logcounts(sceBatches) <- log1p(counts(sceBatches))
## Not run:
if (require("harmony"))
  sceCorr <- runHarmony(sceBatches)

## End(Not run)
**runKMeans**

**Get clustering with KMeans**

### Description

Perform KMeans clustering on a **SingleCellExperiment** object, with **kmeans**.

### Usage

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

### Arguments

- `inSCE` A **SingleCellExperiment** object.
- `nCenters` An integer, the number of centroids (clusters).
- `useReducedDim` A single character, specifying which low-dimension representation to perform the clustering algorithm on. Default "PCA".
- `clusterName` A single character, specifying the name to store the cluster label in **colData**. Default "KMeans_cluster".
- `nComp` An integer. The number of components to use for K-Means. Default 10. See Detail.
- `nIter` An integer, the maximum number of iterations allowed. Default 10.
- `nStart` An integer, the number of random sets to choose. Default 1.
- `seed` An integer. The seed for the random number generator. Default 12345.
- `algorithm` 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

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

Apply Limma’s batch effect correction method to SingleCellExperiment object

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 Input SingleCellExperiment object
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 assay. Default "LIMMA".
batch A single character indicating a field in colData that annotates the batches of each cell; or a vector/factor with the same length as the number of cells. Default "batch".

Value

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

References

Gordon K Smyth, et al., 2003

data('sceBatches', package = 'singleCellTK')
logcounts(sceBatches) <- log1p(counts(sceBatches))
sceCorr <- runLimmaBC(sceBatches)
runMNNCorrect

Apply the mutual nearest neighbors (MNN) batch effect correction method to SingleCellExperiment object

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,
  propK = NULL,
  sigma = 0.1,
  cosNormIn = TRUE,
  cosNormOut = TRUE,
  varAdj = TRUE,
  BPPARAM = BiocParallel::SerialParam()
)

Arguments

inSCE Input SingleCellExperiment object
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 of each cell; or a vector/factor with the same length as the number of cells. Default "batch".
assayName A single character. The name for the corrected assay. Will be saved to assay. Default "MNN".
k An integer scalar specifying the number of nearest neighbors to consider when identifying MNNs. See "See Also". Default 20.
propK A numeric scalar in (0, 1) specifying the proportion of cells in each dataset to use for mutual nearest neighbor searching. See "See Also". Default NULL.
sigma A numeric scalar specifying the bandwidth of the Gaussian smoothing kernel used to compute the correction vector for each cell. See "See Also". Default 0.1.
### runModelGeneVar

**Description**

Generates and stores variability data in the input `SingleCellExperiment` object, using `modelGeneVar` method.

Also selects a specified number of top HVGs and store the logical selection in `rowData`.

**Usage**

```r
rungModelGeneVar(inSCE, useAssay = "logcounts")
```

**Arguments**

- **inSCE**
  - A `SingleCellExperiment` object

- **useAssay**
  - A character string to specify an assay to compute variable features from. Default "logcounts".

### cosNormIn

A logical scalar indicating whether cosine normalization should be performed on the input data prior to calculating distances between cells. See "See Also". Default TRUE.

### cosNormOut

A logical scalar indicating whether cosine normalization should be performed prior to computing corrected expression values. See "See Also". Default TRUE.

### varAdj

A logical scalar indicating whether variance adjustment should be performed on the correction vectors. See "See Also". Default TRUE.

### BPPARAM

A `BiocParallelParam` object specifying whether the PCA and nearest-neighbor searches should be parallelized.

**Value**

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

**References**

Haghverdi L, Lun ATL, et. al., 2018

**See Also**

- `mnnCorrect`

**Examples**

```r
data('sceBatches', package = 'singleCellTK')
logcounts(sceBatches) <- log1p(counts(sceBatches))
sceCorr <- runMNNCorrect(sceBatches)
```
runNormalization

Description

Wrapper function to run any of the integrated normalization/transformation methods in the single-CellITK. The available methods include 'LogNormalize', 'CLR', 'RC' and 'SCTransform' from Seurat, 'logNormCounts and 'CPM' from Scater. Additionally, users can 'scale' using Z.Score, 'transform' using log, log1p and sqrt, add 'pseudocounts' and trim the final matrices between a range of values.

Usage

runNormalization(
  inSCE,
  useAssay = "counts",
  outAssayName = "logcounts",
  normalizationMethod = "logNormCounts",
  scale = FALSE,
  seuratScaleFactor = 10000,
  transformation = NULL,
  pseudocountsBeforeNorm = NULL,
  pseudocountsBeforeTransform = NULL,
  trim = NULL,
  verbose = TRUE
)
runNormalization

Arguments

- **inSCE**: Input SingleCellExperiment object.
- **useAssay**: Specify the name of the assay that should be used.
- **outAssayName**: Specify the name of the new output assay.
- **normalizationMethod**: Specify a normalization method from ‘LogNormalize’, ‘CLR’, ‘RC’ and ‘SC-Transform’ from Seurat or ‘logNormCounts’ and ‘CPM’ from scater packages. Default NULL is set which will not run any normalization method.
- **scale**: Logical value indicating if the data should be scaled using Z.Score. Default FALSE.
- **seuratScaleFactor**: Specify the ‘scaleFactor’ argument if a Seurat normalization method is selected. Default is 10000. This parameter will not be used if methods other than seurat are selected.
- **transformation**: Specify the transformation options to run on the selected assay. Options include ‘log2’ (base 2 log transformation), ‘log1p’ (natural log + 1 transformation) and ‘sqrt’ (square root). Default value is NULL, which will not run any transformation.
- **pseudocountsBeforeNorm**: Specify a numeric pseudo value that should be added to the assay before normalization is performed. Default is NULL, which will not add any value.
- **pseudocountsBeforeTransform**: Specify a numeric pseudo value that should be added to the assay before transformation is run. Default is NULL, which will not add any value.
- **trim**: Specify a vector of two numeric values that should be used as the upper and lower trim values to trim the assay between these two values. For example, c(10,-10) will trim the values between 10 and -10. Default is NULL, which will not trim the data assay.
- **verbose**: Logical value indicating if progress messages should be displayed to the user. Default is TRUE.

Value

Output SCE object with new normalized/transformed assay stored.

Examples

```r
data(sce_chcl, package = "scds")
sce_chcl <- runNormalization(
inSCE = sce_chcl,
normalizationMethod = "LogNormalize",
useAssay = "counts",
outAssayName = "logcounts")```
runPerCellQC  

Wrapper for calculating QC metrics with scater.

Description

A wrapper function for addPerCellQC. Calculate general quality control metrics for each cell in the count matrix.

Usage

```r
runPerCellQC(
  inSCE,
  useAssay = "counts",
  mitoGeneLocation = "rownames",
  mitoRef = c(NULL, "human", "mouse"),
  mitoIDType = c("ensembl", "symbol", "entrez", "ensemblTranscriptID"),
  mitoPrefix = "MT-",
  mitoID = NULL,
  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**: A `SingleCellExperiment` object.
- **useAssay**: A string specifying which assay in the SCE to use. Default "counts".
- **mitoGeneLocation**: Character. Describes the location within `inSCE` where the gene identifiers in the mitochondrial gene sets should be located. 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 identifies will be mapped to that column in the `rowData` of `inSCE`. See `featureIndex` for more information. If this parameter is set to NULL, then no mitochondrial metrics will be calculated. Default "rownames".
- **mitoRef**: Character. The species used to extract mitochondrial genes ID from build-in mitochondrial geneset in SCTK. Available species options are "human" and "mouse". Default is "human".
- **mitoIDType**: Character. Types of mitochondrial gene id. SCTK supports "symbol", "entrez", "ensembl" and "ensemblTranscriptID". It is used with `mitoRef` to extract mitochondrial genes from build-in mitochondrial geneset in SCTK. Default NULL.
mitoPrefix Character. The prefix used to get mitochondrial gene from either rownames(inSCE) or columns of rowData(inSCE) specified by mitoGeneLocation. This parameter is usually used to extract mitochondrial genes from the gene symbol. For example, mitoPrefix = "^MT-" can be used to detect mito gene symbols like "MT-ND4". Note that case is ignored so "mt-" will still match "MT-ND4". Default "^MT-".

mitoID Character. A vector of mitochondrial genes to be quantified.

collectionName Character. Name of a GeneSetCollection obtained by using one of the importGeneSet* 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.

geneSetListLocation Character or numeric vector. If set to 'rownames', then the genes in geneSetList will be looked up in rownames(inSCE). If another character is supplied, then genes will be looked up in the column names of rowData(inSCE). A character vector with the same length as geneSetList can be supplied if the IDs for different gene sets are found in different places, including a mixture of 'rownames' and rowData(inSCE). An integer or integer vector can be supplied to denote the column index in rowData(inSCE). Default 'rownames'.

geneSetCollection Class of GeneSetCollection from package GSEABase. The location of the gene IDs in inSCE should be in the description slot of each gene set and should follow the same notation as geneSetListLocation. The function getGmt 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 inSCE. These gene sets will be included with those from geneSetList if both parameters are provided.

percent_top 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. Default c(50, 100, 200, 500).

use_altexps Logical scalar indicating whether QC statistics should be computed for alternative Experiments in inSCE (altExps(inSCE)). If TRUE, statistics are computed for all alternative experiments. Alternatively, an integer or character vector specifying the alternative Experiments to use to compute QC statistics. Alternatively NULL, in which case alternative experiments are not used. Default FALSE.

flatten Logical scalar indicating whether the nested DataFrame-class in the output should be flattened. Default TRUE.

detectionLimit A numeric scalar specifying the lower detection limit for expression. Default 0

BPPARAM A BiocParallelParam object specifying whether the QC calculations should be parallelized. Default BiocParallel::SerialParam().

Details

This function allows multiple ways to import mitochondrial genes and quantify their expression in cells. mitoGeneLocation is required for all methods to point to the location within inSCE object that stores the mitochondrial gene IDs or Symbols. The various ways mito genes can be specified are:
• A combination of mitoRef and mitoIDType parameters can be used to load pre-built mitochondrial gene sets stored in the SCTK package. These parameters are used in the importMitoGeneSet function.

• The mitoPrefix parameter can be used to search for features matching a particular pattern. The default pattern is an "MT-" at the beginning of the ID.

• The mitoID parameter can be used to directly supply a vector of mitochondrial gene IDs or names. Only features that exactly match items in this vector will be included in the mitochondrial gene set.

Value

A SingleCellExperiment object with cell QC metrics added to the colData slot.

See Also

addPerCellQC, link{plotRunPerCellQCRresults}, runCellQC

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)

runSCANORAMA

Apply the mutual nearest neighbors (MNN) batch effect correction method to SingleCellExperiment object

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",
assayName = "SCANORAMA",
SIGMA = 15,
ALPHA = 0.1,
KNN = 20,
approx = TRUE
)
Arguments

inSCE  
Input `SingleCellExperiment` object

useAssay  
A single character indicating the name of the assay requiring batch correction. Scanorama requires a transformed normalized expression assay. Default "logcounts".

batch  
A single character indicating a field in colData that annotates the batches of each cell; or a vector/factor with the same length as the number of cells. Default "batch".

assayName  
A single character. The name for the corrected assay. Will be saved to `assay`. Default "SCANORAMA".

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  

approx  
Boolean. Use approximate nearest neighbors, greatly speeds up matching runtime. Default TRUE.

Value

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

References

Brian Hie et al, 2019

Examples

```r
## Not run:
data('sceBatches', package = 'singleCellTK')
logcounts(sceBatches) <- log1p(counts(sceBatches))
sceCorr <- runSCANORAMA(sceBatches, "ScaterLogNormCounts")
## End(Not run)
```

runScanpyFindClusters

runScanpyFindClusters Computes the clusters from the input sce object and stores them back in sce object

Description

runScanpyFindClusters Computes the clusters from the input sce object and stores them back in sce object
runScanpyFindClusters

Usage

runScanpyFindClusters(
  inSCE,
  useAssay = "scanpyScaledData",
  useReducedDim = "scanpyPCA",
  nNeighbors = 10,
  dims = 40,
  method = c("leiden", "louvain"),
  colDataName = NULL,
  resolution = 1,
  niterations = -1,
  flavor = "vtraag",
  use_weights = FALSE,
  cor_method = "pearson",
  inplace = TRUE,
  externalReduction = NULL,
  seed = 12345
)

Arguments

inSCE (sce) object from which clusters should be computed and stored in
useAssay Assay containing scaled counts to use for clustering.
useReducedDim Reduction method to use for computing clusters. Default "scanpyPCA".
nNeighbors The size of local neighborhood (in terms of number of neighboring data points) 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 10.
dims numeric value of how many components to use for computing clusters. Default 40.
method selected method to compute clusters. One of "louvain", and "leiden". Default louvain.
colDataName Specify the name to give to this clustering result. Default is NULL that will generate a meaningful name automatically.
resolution A parameter value controlling the coarseness of the clustering. Higher values lead to more clusters Default 1.
niterations How many iterations of the Leiden clustering method to perform. Positive values above 2 define the total number of iterations to perform, -1 has the method run until it reaches its optimal clustering. Default -1.
flavor Choose between to packages for computing the clustering. Default vtraag
use_weights Boolean. Use weights from knn graph. Default FALSE
inplace If True, adds dendrogram information to annData object, else this function returns the information. Default TRUE
runScanpyFindHVG

runScanpyFindHVG Find highly variable genes and store in the input sce object

Description

runScanpyFindHVG Find highly variable genes and store in the input sce object

Usage

runScanpyFindHVG(
  inSCE,
  useAssay = "scanpyNormData",
  method = c("seurat", "cell_ranger", "seurat_v3"),
  altExpName = "featureSubset",
  altExp = FALSE,
  hvgNumber = 2000,
  minMean = 0.0125,
  maxMean = 3,
  minDisp = 0.5,
  maxDisp = Inf
)

Arguments

inSCE (sce) object to compute highly variable genes from and to store back to it
useAssay Specify the name of the assay to use for computation of variable genes. It is recommended to use log normalized data, except when flavor='seurat_v3', in which counts data is expected.
runScanpyFindMarkers

Description

runScanpyFindMarkers

Usage

runScanpyFindMarkers(
  inSCE,
  nGenes = NULL,
  useAssay = "scanpyNormData",
  colDataName,
  group1 = "all",
  method = 'seurat',
  altExpName = character(),
  altExp = FALSE,
  hvgNumber = 2000,
  minMean = 0.0125,
  maxMean = 3,
  minDisp = 0.5,
  maxDisp = Inf
)

Value

Updated SingleCellExperiment object with highly variable genes computation stored getTopHVG, plotTopHVG

Examples

data(scExample, package = "singleCellTK")
  # Not run:
  sce <- runScanpyNormalizeData(sce, useAssay = "counts")
  sce <- runScanpyFindHVG(sce, useAssay = "scanpyNormData", method = "seurat")
  g <- getTopHVG(sce, method = "seurat", hvgNumber = 500)
  # End(Not run)
group2 = "rest",
test = c("wilcoxon", "t-test", "t-test_overestim_var", "logreg"),
corr_method = c("benjamini-hochberg", "bonferroni")
)

Arguments

inSCE Input SingleCellExperiment object.
nGenes The number of genes that appear in the returned tables. Defaults to all genes.
useAssay Specify the name of the assay to use for computation of marker genes. It is recommended to use log normalized assay.
colDataName colData to use as the key of the observations grouping to consider.
group1 Name of group1. Subset of groups, to which comparison shall be restricted, or 'all' (default), for all groups.
group2 Name of group2. If 'rest', compare each group to the union of the rest of the group. If a group identifier, compare with respect to this group. Default is 'rest'
test Test to use for DE. Default "t-test".
corr_method p-value correction method. Used only for 't-test', 't-test_overestim_var', and 'wilcoxon'.

Value

A SingleCellExperiment object that contains marker genes populated in a data.frame stored inside metadata slot.

Examples

data(scExample, package = "singleCellTK")
## Not run:
sce <- runScanpyNormalizeData(sce, useAssay = "counts")
sce <- runScanpyFindHVG(sce, useAssay = "scanpyNormData", method = "seurat")
sce <- runScanpyScaleData(sce, useAssay = "scanpyNormData")
sce <- runScanpyPCA(sce, useAssay = "scanpyScaledData")
sce <- runScanpyFindClusters(sce, useReducedDim = "scanpyPCA")
sce <- runScanpyFindMarkers(sce, colDataName = "Scanpy_louvain_1" )
## End(Not run)
runScanpyPCA

Description

runScanpyNormalizeData Wrapper for NormalizeData() function from scanpy library. Normalizes the sce object according to the input parameters.

Usage

runScanpyNormalizeData(
  inSCE,
  useAssay,
  targetSum = 10000,
  maxFraction = 0.05,
  normAssayName = "scanpyNormData"
)

Arguments

inSCE (sce) object to normalize
useAssay Assay containing raw counts to use for normalization.
targetSum If NULL, after normalization, each observation (cell) has a total count equal to the median of total counts for observations (cells) before normalization. Default 1e4
maxFraction Include cells that have more counts than max_fraction of the original total counts in at least one cell. Default 0.05
normAssayName Name of new assay containing normalized data. Default scanpyNormData.

Value

Normalized SingleCellExperiment object

Examples

data(scExample, package = "singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
rownames(sce) <- rowData(sce)$feature_name
## Not run:
sce <- runScanpyNormalizeData(sce, useAssay = "counts")
## End(Not run)

runScanpyPCA

runScanpyPCA Computes PCA on the input sce object and stores the calculated principal components within the sce object.

Description

runScanpyPCA Computes PCA on the input sce object and stores the calculated principal components within the sce object.
runScanpyScaleData

Usage

runScanpyPCA(
    inSCE,
    useAssay = "scanpyScaledData",
    reducedDimName = "scanpyPCA",
    nPCs = 50,
    method = c("arpack", "randomized", "auto", "lobpcg"),
    use_highly_variable = TRUE,
    seed = 12345
)

Arguments

inSCE (sce) object on which to compute PCA
useAssay Assay containing scaled counts to use in PCA. Default "scanpyScaledData".
reducedDimName Name of new reducedDims object containing Scanpy PCA. Default scanpyPCA.
nPCs numeric value of how many components to compute. Default 50.
method selected method to use for computation of pca. One of 'arpack', 'randomized', 'auto' or 'lobpcg'. Default "arpack".
use_highly_variable boolean value of whether to use highly variable genes only. By default uses them if they have been determined beforehand.
seed Specify numeric value to set as a seed. Default 12345.

Value

Updated SingleCellExperiment object which now contains the computed principal components

Examples

data(scExample, package = "singleCellTK")
## Not run:
sce <- runScanpyNormalizeData(sce, useAssay = "counts")
sce <- runScanpyFindHVG(sce, useAssay = "scanpyNormData", method = "seurat")
sce <- runScanpyScaleData(sce, useAssay = "scanpyNormData")
sce <- runScanpyPCA(sce, useAssay = "scanpyScaledData")
## End(Not run)

runScanpyScaleData runScanpyScaleData Scales the input sce object according to the input parameters

Description

runScanpyScaleData Scales the input sce object according to the input parameters
runScanpyTSNE

Usage

runScanpyTSNE(
    inSCE,
    useAssay = NULL,
    useReducedDim = "scanpyPCA",
    reducedDimName = "scanpyTSNE",
    dims = 40,
    perplexity = 30,
    externalReduction = NULL,
    seed = 12345
)

Description

runScanpyTSNE Computes tSNE from the given sce object and stores the tSNE computations back into the sce object.

Usage

runScanpyScaleData(
    inSCE,
    useAssay = "scanpyNormData",
    scaledAssayName = "scanpyScaledData"
)

Arguments

inSCE (sce) object to scale
useAssay Assay containing normalized counts to scale.
scaledAssayName Name of new assay containing scaled data. Default scanpyScaledData.

Value

Scaled SingleCellExperiment object

Examples

data(scExample, package = "singleCellTK")
## Not run:
sce <- runScanpyNormalizeData(sce, useAssay = "counts")
sce <- runScanpyScaleData(sce, useAssay = "scanpyNormData")
## End(Not run)
runScanpyUMAP

runScanpyUMAP Computes UMAP from the given sce object and stores the UMAP computations back into the sce object

Description

runScanpyUMAP Computes UMAP from the given sce object and stores the UMAP computations back into the sce object

Usage

runScanpyUMAP(
  inSCE,
  useAssay = NULL,
  useReducedDim = "scanpyPCA",
  reducedDimName = "scanpyUMAP",
  ...)
runScanpyUMAP

```r
dims = 40,
minDist = 0.5,
nNeighbors = 10,
spread = 1,
alpha = 1,
gamma = 1,
externalReduction = NULL,
seed = 12345
)
```

**Arguments**

- `inSCE` (sce) object on which to compute the UMAP
- `useAssay` Specify name of assay to use. Default is NULL, so useReducedDim param will be used instead.
- `useReducedDim` Reduction to use for computing UMAP. Default is "scanpyPCA".
- `reducedDimName` Name of new reducedDims object containing Scanpy UMAP. Default scanpyUMAP.
- `dims` Numerical value of how many reduction components to use for UMAP computation. Default 40.
- `minDist` Sets the "min_dist" parameter to the underlying UMAP call. Default 0.5.
- `nNeighbors` Sets the "n_neighbors" parameter to the underlying UMAP call. Default 10.
- `spread` Sets the "spread" parameter to the underlying UMAP call. Default 1.
- `alpha` Sets the "alpha" parameter to the underlying UMAP call. Default 1.
- `gamma` Sets the "gamma" parameter to the underlying UMAP call. Default 1.
- `externalReduction` Pass DimReduce object if PCA computed through other libraries. Default NULL.
- `seed` Specify numeric value to set as a seed. Default 12345.

**Value**

Updated sce object with UMAP computations stored

**Examples**

```r
data(scExample, package = "singleCellTK")
# Not run:
sce <- runScanpyNormalizeData(sce, useAssay = "counts")
sce <- runScanpyFindHVG(sce, useAssay = "scanpyNormData", method = "seurat")
sce <- runScanpyScaleData(sce, useAssay = "scanpyNormData")
sce <- runScanpyPCA(sce, useAssay = "scanpyScaledData")
sce <- runScanpyFindClusters(sce, useReducedDim = "scanpyPCA")
sce <- runScanpyUMAP(sce, useReducedDim = "scanpyPCA")
# End(Not run)
```
runScDblFinder

Detect doublet cells using scDblFinder.

Description

A wrapper function for scDblFinder. Identify potential doublet cells based on simulations of putative doublet expression profiles. Generate a doublet score for each cell.

Usage

```r
runScDblFinder(
  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 or colData variable name. Indicates which sample each cell belongs to. Default NULL.
- `useAssay`: A string specifying which assay in the SCE to use. Default "counts".
- `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, can be set to NULL. Default 12345.
- `BPPARAM`: A BiocParallelParam-class object specifying whether the neighbour searches should be parallelized. Default BiocParallel::SerialParam().

Details

This function is a wrapper function for scDblFinder. runScDblFinder runs scDblFinder 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 scDblFinder QC outputs added to the colData slot.

References

runSCMerge

Apply scMerge batch effect correction method to SingleCellExperiment object

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",
  hvgExprs = "counts",
  seg = NULL,
  kmeansK = NULL,
  cellType = NULL,
  BPPARAM = BiocParallel::SerialParam()
)

Arguments

inSCE  Input SingleCellExperiment object

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".

hvgExprs A single character. The assay that to be used for highly variable genes identification. Default "counts".

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 with segList or segList_ensemblGeneID. Default NULL, and this value will be auto-detected by default with scSEGIndex.
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,
  useReducedDim = NULL,
  useAssay = NULL,
  useAltExp = NULL,
  altExpAssay = "counts",
  altExpRedDim = NULL,
  clusterName = "scranSNN_cluster",
  k = 8,
  nComp = 10,
)

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 codekmeansK needs to be the same as the number of batches. Default NULL, and this value will be auto-detected by default, depending on cellType.

cellType  A single character. A string indicating a field in colData(inSCE) that defines different cell types. Default 'cell_type'.

BPPARAM  A BiocParallelParam object specifying whether should be parallelized. Default BiocParallel::SerialParam().

Value

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

References

Hoa, et al., 2020

Examples

data('sceBatches', package = 'singleCellTK')
  ## Not run:
  logcounts(sceBatches) <- log1p(counts(sceBatches))
  sceCorr <- runSCMerge(sceBatches)
  
  ## End(Not run)
weightType = c("rank", "number", "jaccard"),
algorithm = c("louvain", "leiden", "walktrap", "infomap", "fastGreedy", "labelProp", "leadingEigen"),
BPPARAM = BiocParallel::SerialParam(),
seed = 12345,
)

Arguments

inSCE A SingleCellExperiment object.
useReducedDim A single character, specifying which low-dimension representation (reducedDim) to perform the clustering algorithm on. Default NULL.
useAssay A single character, specifying which assay 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".
altExpRedDim A single character, specifying which reducedDim within the altExp specified by useAltExp to use. Only used when useAltExp is set. Default NULL.
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 8.
nComp An integer. The number of components to use for graph construction. Default 10. See Detail.
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 "leiden", "louvain", "walktrap", "infomap", "fastGreedy", "labelProp", "leadingEigen". Default "louvain". See Detail.
BPPARAM A BiocParallelParam object to use for processing the SNN graph generation step in parallel.
seed Random seed for reproducibility of results. Default NULL will use global seed in use by the R environment.
... Other optional parameters passed to the igraph clustering functions. See Details.

Details

Different graph based clustering algorithms have diverse sets of parameters that users can tweak. The help information can be found here:
The Scran SNN building method can work on specified nComp components. When users specify input matrix by useAssay or useAltExp + altExpAssay, the method will generate nComp components and use them all. When specifying useReducedDim or useAltExp + altExpRedDim, this function will subset the top nComp components and pass them to the method.

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")

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,
runScrublet

Arguments

inSCE  A SingleCellExperiment object.
sample  Character vector or colData variable name. Indicates which sample each cell belongs to. Default NULL.
useAssay  A string specifying which assay in the SCE to use. Default "counts".
simDoubletRatio  Numeric. Number of doublets to simulate relative to the number of observed transcriptomes. Default 2.0.
nNeighbors  Integer. Number of neighbors used to construct the KNN graph of observed transcriptomes and simulated doublets. If NULL, this is set to round(0.5 * sqrt(n_cells)). Default NULL.
minDist  Float Determines how tightly UMAP packs points together. If NULL, this is set to 0.1. Default NULL.
expectedDoubletRate  The estimated doublet rate for the experiment. Default 0.1.
stdevDoubletRate  Uncertainty in the expected doublet rate. Default 0.02.
syntheticDoubletUmiSubsampling  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.
useApproxNeighbors  Boolean. Use approximate nearest neighbor method (annoy) for the KNN classifier. Default TRUE.
**distanceMetric** Character. Distance metric used when finding nearest neighbors. See detail. Default "euclidean".

**getDoubletNeighborParents**

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.

**minCounts** Numeric. Used for gene filtering prior to PCA. Genes expressed at fewer than minCounts in fewer than minCells are excluded. Default 3.

**minCells** Integer. Used for gene filtering prior to PCA. Genes expressed at fewer than minCounts in fewer than minCells 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 (Klein et al., Cell 2015). Default 85.

**logTransform** Boolean. If TRUE, log-transform the counts matrix (log1p(TPM)). sklearn.decomposition.TruncatedSVD 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. sklearn.decomposition.PCA will be used for dimensionality reduction. Default TRUE.

**normalizeVariance**

Boolean. If TRUE, normalize the data such that each gene has a variance of 1. sklearn.decomposition.TruncatedSVD 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 NULL, it is set to 0.5. Default NULL.

**tsnePerplexity** Integer. The number of nearest neighbors that is used in other manifold learning algorithms. If NULL, it is set to 30. Default NULL.

**verbose**

Boolean. If TRUE, print progress updates. Default TRUE.

**seed**

Seed for the random number generator, can be set to NULL. Default 12345.

**Details**

For the list of valid values for distanceMetric, see the documentation for annoy (if useApproxNeighbors is TRUE) or sklearn.neighbors.NearestNeighbors (if useApproxNeighbors is FALSE).

**Value**

A SingleCellExperiment object with scrub_doublets output appended to the colData slot. The columns include scrublet_score and scrublet_call.

**See Also**

plotScrubletResults, runCellQC
runSeuratFindClusters

runSeuratFindClusters Computes the clusters from the input sce object and stores them back in sce object

Description

runSeuratFindClusters Computes the clusters from the input sce object and stores them back in sce object

Usage

runSeuratFindClusters(
  inSCE, 
  useAssay = "seuratNormData", 
  useReduction = c("pca", "ica"), 
  dims = 10, 
  algorithm = c("louvain", "multilevel", "SLM"), 
  groupSingletons = TRUE, 
  resolution = 0.8, 
  seed = 12345, 
  externalReduction = NULL, 
  verbose = TRUE 
)

Arguments

inSCE (sce) object from which clusters should be computed and stored in
useAssay Assay containing scaled counts to use for clustering.
useReduction Reduction method to use for computing clusters. One of "pca" or "ica". Default "pca".
dims numeric value of how many components to use for computing clusters. Default 10.
algorithm selected algorithm to compute clusters. One of "louvain", "multilevel", or "SLM". Use louvain for "original Louvain algorithm" and multilevel for "Louvain algorithm with multilevel refinement". Default louvain.
groupSingletons boolean if singletons should be grouped together or not. Default TRUE.
RunSeuratFindHVG

runSeuratFindHVG

Find highly variable genes and store in the input sce object

Description

runSeuratFindHVG Find highly variable genes and store in the input sce object

Usage

runSeuratFindHVG(
  inSCE,
  useAssay = "counts",
  method = c("vst", "dispersion", "mean.var.plot"),
  hvgNumber = 2000,
  createFeatureSubset = "hvf",
  altExp = FALSE,
  verbose = TRUE
)
Arguments

inSCE (sce) object to compute highly variable genes from and to store back to it
useAssay Specify the name of the assay to use for computation of variable genes. It is recommended to use a raw counts assay with the "vst" method and normalized assay with all other methods. Default is "counts".
method selected method to use for computation of highly variable genes. One of 'vst', 'dispersion', or 'mean.var.plot'. Default "vst" which uses the raw counts. All other methods use normalized counts.
hvgNumber numeric value of how many genes to select as highly variable. Default 2000
createFeatureSubset Specify a name of the subset to create for the identified variable features. Default is "hvf". Leave it NULL if you do not want to create a subset of variable features.
altExp Logical value indicating if the input object is an altExperiment. Default FALSE.
verbose Logical value indicating if informative messages should be displayed. Default is TRUE.

Value

Updated SingleCellExperiment object with highly variable genes computation stored

See Also

runFeatureSelection, runModelGeneVar, getTopHVG, plotTopHVG

Examples

data(scExample, package = "singleCellTK")
sce <- runSeuratFindHVG(sce)

Description

runSeuratFindMarkers

Usage

runSeuratFindMarkers(
inSCE,
cells1 = NULL,
cells2 = NULL,
group1 = NULL,
group2 = NULL,
allGroup = NULL,
conserved = FALSE,
test = "wilcox",
onlyPos = FALSE,
minPCT = 0.1,
threshUse = 0.25,
verbose = TRUE
)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inSCE</td>
<td>Input SingleCellExperiment object.</td>
</tr>
<tr>
<td>cells1</td>
<td>A list of sample names included in group1.</td>
</tr>
<tr>
<td>cells2</td>
<td>A list of sample names included in group2.</td>
</tr>
<tr>
<td>group1</td>
<td>Name of group1.</td>
</tr>
<tr>
<td>group2</td>
<td>Name of group2.</td>
</tr>
<tr>
<td>allGroup</td>
<td>Name of all groups.</td>
</tr>
<tr>
<td>conserved</td>
<td>Logical value indicating if markers conserved between two groups should be identified. Default is FALSE.</td>
</tr>
<tr>
<td>test</td>
<td>Test to use for DE. Default &quot;wilcox&quot;.</td>
</tr>
<tr>
<td>onlyPos</td>
<td>Logical value indicating if only positive markers should be returned.</td>
</tr>
<tr>
<td>minPCT</td>
<td>Numeric value indicating the minimum fraction of min.pct cells in which genes are detected. Default is 0.1.</td>
</tr>
<tr>
<td>threshUse</td>
<td>Numeric value indicating the logFC threshold value on which on average, at least X-fold difference (log-scale) between the two groups of cells exists. Default is 0.25.</td>
</tr>
<tr>
<td>verbose</td>
<td>Logical value indicating if informative messages should be displayed. Default is TRUE.</td>
</tr>
</tbody>
</table>

Value

A SingleCellExperiment object that contains marker genes populated in a data.frame stored inside metadata slot.

**runSeuratHeatmap**

**runSeuratHeatmap** Computes the heatmap plot object from the pca slot in the input sce object

Description

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

Usage

runSeuratHeatmap(
  inSCE,
  useAssay,
  useReduction = c("pca", "ica"),
  dims = NULL,
  nfeatures = 30,
  cells = NULL,
  ncol = NULL,
  balanced = TRUE,
  fast = TRUE,
  combine = TRUE,
  raster = TRUE,
  externalReduction = NULL
)

Arguments

inSCE (sce) object from which to compute heatmap (pca should be computed)
useAssay Specify name of the assay that will be scaled by this function. The output scaled
  assay will be used for computation of the 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 Number of features to include in the heatmap. Default 30.
cells Numeric value indicating the number of top cells to plot. Default is NULL which
  indicates all cells.
col Numeric value indicating the number of columns to use for plot. Default is NULL which
  will automatically compute accordingly.
balanced Plot equal number of genes with positive and negative scores. Default is TRUE.
fast See DimHeatmap for more information. Default TRUE.
combine See DimHeatmap for more information. Default TRUE.
raster See DimHeatmap for more information. Default TRUE.
externalReduction Pass DimReduce object if PCA/ICA computed through other libraries. Default
  NULL.

Value

plot object
Examples

data(scExample, package = "singleCellTK")
## Not run:
sce <- runSeuratNormalizeData(sce, useAssay = "counts")
sce <- runSeuratFindHVG(sce, useAssay = "counts")
sce <- runSeuratScaleData(sce, useAssay = "counts")
sce <- runSeuratPCA(sce, useAssay = "counts")
heatmap <- runSeuratHeatmap(sce, useAssay = "counts")
plotSeuratHeatmap(heatmap)
## End(Not run)

runSeuratICA

runSeuratICA Computes ICA on the input sce object and stores the calculated independent components within the sce object

Description

runSeuratICA Computes ICA on the input sce object and stores the calculated independent components within the sce object

Usage

runSeuratICA(
  inSCE,  # (sce) object on which to compute ICA
  useAssay = "seuratScaledData",  # Assay containing scaled counts to use in ICA.
  useFeatureSubset = NULL,  # Subset of feature to use for dimension reduction. A character string indicating a rowData variable that stores the logical vector of HVG selection, or a vector that can subset the rows of inSCE. Default NULL.
  scale = TRUE,  # Logical scalar, whether to standardize the expression values using ScaleData. Default TRUE.
  reducedDimName = "seuratICA",  # Name of new reducedDims object containing Seurat ICA Default seuratICA.
  nics = 20,  # Number of independent components to compute. Default 20.
  seed = 12345,  # Seed for random number generator. Default 12345.
  verbose = FALSE  # Whether to print information during computation. Default FALSE.
)

Arguments

inSCE  # (sce) object on which to compute ICA
useAssay  # Assay containing scaled counts to use in ICA.
useFeatureSubset  # Subset of feature to use for dimension reduction. A character string indicating a rowData variable that stores the logical vector of HVG selection, or a vector that can subset the rows of inSCE. Default NULL.
scale  # Logical scalar, whether to standardize the expression values using ScaleData. Default TRUE.
reducedDimName  # Name of new reducedDims object containing Seurat ICA Default seuratICA.
nics  # Number of independent components to compute. Default 20.
runSeuratIntegration

A wrapper function to Seurat Batch-Correction/Integration workflow.

Description

runSeuratIntegration A wrapper function to Seurat Batch-Correction/Integration workflow.

Usage

runSeuratIntegration(
  inSCE, 
  useAssay = "counts", 
  batch, 
  newAssayName = "SeuratIntegratedAssay", 
  kAnchor, 
  kFilter, 
  kWeight, 
  ndims = 10
)

seed Random seed for reproducibility of results. Default NULL will use global seed in use by the R environment.

verbose Logical value indicating if informative messages should be displayed. Default is TRUE.

Details

For features used for computation, it can be controlled by features or useFeatureSubset. When features is specified, the scaling and dimensionality reduction will only be processed with these features. When features is NULL but useFeatureSubset is specified, will use the features that the HVG list points to. If both parameters are NULL, the function will see if any Seurat’s variable feature detection has been ever performed, and use them if found. Otherwise, all features are used.

Value

Updated SingleCellExperiment object which now contains the computed independent components

Examples

data(scExample, package = "singleCellTK")
## Not run:
sce <- runSeuratNormalizeData(sce, useAssay = "counts")
sce <- runSeuratFindHVG(sce, useAssay = "counts")
sce <- runSeuratScaleData(sce, useAssay = "counts")
sce <- runSeuratICA(sce, useAssay = "counts")
## End(Not run)
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 weighting 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.

---

**runSeuratJackStraw**

`runSeuratJackStraw` Compute jackstraw plot and store the computations in the input sce object

Description

`runSeuratJackStraw` Compute jackstraw plot and store the computations in the input sce object.

Usage

```r
runSeuratJackStraw(
  inSCE,
  useAssay,
  dims = NULL,
  numReplicate = 100,
  propFreq = 0.025,
  externalReduction = NULL
)
```

Arguments

- **inSCE**: (sce) object on which to compute and store jackstraw plot.
- **useAssay**: Specify name of the assay to use for scaling. Assay name provided against this parameter is scaled by the function and used for the computation of JackStraw scores along with the reduced dimensions specified by the `dims` parameter.
runSeuratNormalizeData

runSeuratNormalizeData Wrapper for NormalizeData() function from seurat library Normalizes the sce object according to the input parameters

Description

runSeuratNormalizeData Wrapper for NormalizeData() function from seurat library Normalizes the sce object according to the input parameters

Usage

runSeuratNormalizeData(
  inSCE,  
  useAssay,  
  normAssayName = "seuratNormData",  
  normalizationMethod = "LogNormalize",  
  scaleFactor = 10000,  
  verbose = TRUE
)
runSeuratPCA

Computes PCA on the input sce object and stores the calculated principal components within the sce object

Usage

runSeuratPCA(
  inSCE,
  useAssay = "seuratNormData",
  useFeatureSubset = "hvf",
  scale = TRUE,
  reducedDimName = "seuratPCA",
  nPCs = 20,
  seed = 12345,
  verbose = TRUE
)

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.
verbose Logical value indicating if informative messages should be displayed. Default is TRUE.

Value

Normalized SingleCellExperiment object

Examples

data(scExample, package = "singleCellTK")
## Not run:
sce <- runSeuratNormalizeData(sce, useAssay = "counts")
## End(Not run)
Arguments

inSCE (sce) object on which to compute PCA
useAssay Assay containing scaled counts to use in PCA. Default "seuratNormData".
useFeatureSubset Subset of feature to use for dimension reduction. A character string indicating a rowData variable that stores the logical vector of HVG selection, or a vector that can subset the rows of inSCE. Default "hvf".
scale Logical scalar, whether to standardize the expression values using ScaleData. Default TRUE.
reducedDimName Name of new reducedDims object containing Seurat PCA. Default seuratPCA.
nPCs numeric value of how many components to compute. Default 20.
seed Random seed for reproducibility of results. Default NULL will use global seed in use by the R environment.
verbose Logical value indicating if informative messages should be displayed. Default is TRUE.

Details

For features used for computation, it can be controlled by features or useFeatureSubset. When features is specified, the scaling and dimensionality reduction will only be processed with these features. When features is NULL but useFeatureSubset is specified, will use the features that the HVG list points to. If both parameters are NULL, the function will see if any Seurat’s variable feature detection has been ever performed, and use them if found. Otherwise, all features are used.

Value

Updated SingleCellExperiment object which now contains the computed principal components

Examples

data(scExample, package = "singleCellTK")
## Not run:
sce <- runSeuratNormalizeData(sce, useAssay = "counts")
sce <- runSeuratFindHVG(sce, useAssay = "counts")
sce <- setTopHVG(sce, method = "vst", featureSubsetName = "hvf")
sce <- runSeuratScaleData(sce, useAssay = "counts")
sce <- runSeuratPCA(sce, useAssay = "counts")

## End(Not run)
runSeuratScaleData  

runSeuratScaleData Scales the input sce object according to the input parameters

Description

runSeuratScaleData Scales the input sce object according to the input parameters

Usage

runSeuratScaleData(
  inSCE,
  useAssay = "seuratNormData",
  scaledAssayName = "seuratScaledData",
  model = "linear",
  scale = TRUE,
  center = TRUE,
  scaleMax = 10,
  verbose = TRUE
)

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.
verbose Logical value indicating if informative messages should be displayed. Default is TRUE.

Value

Scaled SingleCellExperiment object

Examples

data(scExample, package = "singleCellTK")
## Not run:
sce <- runSeuratNormalizeData(sce, useAssay = "counts")
sce <- runSeuratFindHVG(sce, useAssay = "counts")
sce <- runSeuratScaleData(sce, useAssay = "counts")

## End(Not run)
runSeuratSCTransform

runSeuratSCTransform

Runs the SCTransform function to transform/normalize the input data

Description

runSeuratSCTransform Runs the SCTransform function to transform/normalize the input data

Usage

runSeuratSCTransform(
  inSCE,
  normAssayName = "SCTCounts",
  useAssay = "counts",
  verbose = TRUE
)

Arguments

inSCE Input SingleCellExperiment object
normAssayName Name for the output data assay. Default "SCTCounts".
useAssay Name for the input data assay. Default "counts".
verbose Logical value indicating if informative messages should be displayed. Default is TRUE.

Value

Updated SingleCellExperiment object containing the transformed data

Examples

data("mouseBrainSubsetSCE", package = "singleCellTK")
mouseBrainSubsetSCE <- runSeuratSCTransform(mouseBrainSubsetSCE)

runSeuratTSNE

runSeuratTSNE

Computes tSNE from the given sce object and stores the tSNE computations back into the sce object

Description

runSeuratTSNE Computes tSNE from the given sce object and stores the tSNE computations back into the sce object
Usage

```r	nrunSeuratTSNE(
  inSCE,
  useReduction = c("pca", "ica"),
  reducedDimName = "seuratTSNE",
  dims = 10,
  perplexity = 30,
  externalReduction = NULL,
  seed = 1
)
```

Arguments

- `inSCE` (sce) object on which to compute the tSNE
- `useReduction` selected reduction algorithm to use for computing tSNE. One of "pca" or "ica". Default "pca".
- `reducedDimName` Name of new reducedDims object containing Seurat tSNE. Default seuratTSNE.
- `dims` Number of reduction components to use for tSNE computation. Default 10.
- `perplexity` Adjust the perplexity tuneable parameter for the underlying tSNE call. Default 30.
- `externalReduction` Pass DimRedc object if PCA/ICA computed through other libraries. Default NULL.
- `seed` Random seed for reproducibility of results. Default 1.

Value

Updated sce object with tSNE computations stored

Description

runSeuratUMAP Computes UMAP from the given sce object and stores the UMAP computations back into the sce object

Usage

```r	nrunSeuratUMAP(
  inSCE,
  useReduction = c("pca", "ica"),
  reducedDimName = "seuratUMAP",
  dims = 10,
)```
runSeuratUMAP

minDist = 0.3,
    nNeighbors = 30L,
    spread = 1,
    externalReduction = NULL,
    seed = 42,
    verbose = TRUE
)

Arguments

inSCE (sce) object on which to compute the UMAP

useReduction Reduction to use for computing UMAP. One of "pca" or "ica". Default is "pca".

reducedDimName Name of new reducedDims object containing Seurat UMAP Default seuratUMAP.

dims Numerical value of how many reduction components to use for UMAP computation. Default 10.

minDist Sets the "min.dist" parameter to the underlying UMAP call. See RunUMAP for more information. Default 0.3.

nNeighbors Sets the "n.neighbors" parameter to the underlying UMAP call. See RunUMAP for more information. Default 30L.

spread Sets the "spread" parameter to the underlying UMAP call. See RunUMAP for more information. Default 1.

externalReduction Pass DimReduc object if PCA/ICA computed through other libraries. Default NULL.

seed Random seed for reproducibility of results. Default 42.

verbose Logical value indicating if informative messages should be displayed. Default is TRUE.

Value

Updated sce object with UMAP computations stored

Examples

data(scExample, package = "singleCellTK")
## Not run:
sce <- runSeuratNormalizeData(sce, useAssay = "counts")
sce <- runSeuratFindHVG(sce, useAssay = "counts")
sce <- runSeuratScaleData(sce, useAssay = "counts")
sce <- runSeuratPCA(sce, useAssay = "counts")
sce <- runSeuratFindClusters(sce, useAssay = "counts")
sce <- runSeuratUMAP(sce, useReduction = "pca")

## End(Not run)
runSingleR  

Label cell types with SingleR

Description

SingleR works with a reference dataset where the cell type labeling is given. Given a reference dataset of samples (single-cell or bulk) with known labels, it assigns those labels to new cells from a test dataset based on similarities in their expression profiles.

Usage

runSingleR(
inSCE,
useAssay = "logcounts",
useSCERef = NULL,
labelColName = NULL,
useBltinRef = c("hpca", "bpe", "mp", "dice", "immgen", "mouse", "zeisel"),
level = c("main", "fine", "ont"),
featureType = c("symbol", "ensembl"),
labelByCluster = NULL
)

Arguments

inSCE  
SingleCellExperiment inherited object. Required.

useAssay  
character. A string specifying which assay to use for expression profile identification. Required.

useSCERef  
SingleCellExperiment inherited object. An optional customized reference dataset. Default NULL.

labelColName  
A single character. A string specifying the column in colData(useSCERef) that stores the cell type labeling. Default NULL.

useBltinRef  
A single character. A string that specifies a reference provided by SingleR. Choose from "hpca", "bpe", "mp", "dice", "immgen", "mouse", "zeisel". See detail. Default "hpca".

level  
A string for cell type labeling level. Used only when using some of the SingleR built-in references. Choose from "main", "fine", "ont". Default "main".

featureType  
A string for whether to use gene symbols or Ensembl IDs when using a SingleR built-in reference. Should be set based on the type of rownames of inSCE. Choose from "symbol", "ensembl". Default "symbol".

labelByCluster  
A single character. A string specifying the column name in colData(inSCE) that stores clustering labels. Use this when users want to only label cells on cluster level, instead of performing calculation on each cell. Default NULL.

Value

Input SCE object with cell type labeling updated in colData(inSCE), together with scoring metrics.
runSoupX

Examples

data("sceBatches")
logcounts(sceBatches) <- log1p(counts(sceBatches))
#sceBatches <- runSingleR(sceBatches, useBltinRef = "mp")

runSoupX Detecting and correct contamination with SoupX

Description

A wrapper function for autoEstCont and adjustCounts. Identify potential contamination from experimental factors such as ambient RNA. Visit their vignette for better understanding.

Usage

runSoupX(
inSCE,
sample = NULL,
useAssay = "counts",
background = NULL,
bgAssayName = NULL,
bgBatch = NULL,
assayName = ifelse(is.null(background), "SoupX", "SoupX_bg"),
cluster = NULL,
reducedDimName = ifelse(is.null(background), "SoupX_UMAP_", "SoupX_bg_UMAP_"),
tfidfMin = 1,
soupQuantile = 0.9,
maxMarkers = 100,
contaminationRange = c(0.01, 0.8),
rhoMaxFDR = 0.2,
priorRho = 0.05,
priorRhoStdDev = 0.1,
forceAccept = FALSE,
adjustMethod = c("subtraction", "soupOnly", "multinomial"),
roundToInt = FALSE,
tol = 0.001,
pCut = 0.01
)

Arguments

inSCE

A SingleCellExperiment object.

sample

A single character specifying a name that can be found in \texttt{colData}(inSCE) to directly use the cell annotation; or a character vector with as many elements as cells to indicates which sample each cell belongs to. SoupX will be run on cells from each sample separately. Default NULL.
runSoupX

useAssay  A single character string specifying which assay in inSCE to use. Default 'counts'.

background  A numeric matrix of counts or a SingleCellExperiment object with the matrix in assay slot. It should have the same structure as inSCE except it contains the matrix including empty droplets. Default NULL.

bgAssayName  A single character string specifying which assay in background to use when background is a SingleCellExperiment object. If NULL, the function will use the same value as useAssay. Default NULL.

bgBatch  The same thing as sample but for background. Can be a single character only when background is a SingleCellExperiment object. Default NULL.

assayName  A single character string of the output corrected matrix. Default "SoupX" when not using a background, otherwise, "SoupX_bg".

cluster  Prior knowledge of clustering labels on cells. A single character string for specifying clustering label stored in colData(inSCE), or a character vector with as many elements as cells. When not supplied, quickCluster method will be applied.

reducedDimName  A single character string of the prefix of output corrected embedding matrix for each sample. Default "SoupX_UMAP_" when not using a background, otherwise, "SoupX_bg_UMAP_".


soupQuantile  Numeric. Only use genes that are at or above this expression quantile in the soup. This prevents inaccurate estimates due to using genes with poorly constrained contribution to the background. Default 0.9. See ?SoupX::autoEstCont.

maxMarkers  Integer. If we have heaps of good markers, keep only the best maxMarkers of them. Default 100. See ?SoupX::autoEstCont.

contaminationRange  Numeric vector of two elements. This constrains the contamination fraction to lie within this range. Must be between 0 and 1. The high end of this range is passed to estimateNonExpressingCells as maximumContamination. Default c(0.01, 0.8). See ?SoupX::autoEstCont.

rhoMaxFDR  Numeric. False discovery rate passed to estimateNonExpressingCells, to test if rho is less than maximumContamination. Default 0.2. See ?SoupX::autoEstCont.

priorRho  Numeric. Mode of gamma distribution prior on contamination fraction. Default 0.05. See ?SoupX::autoEstCont.


forceAccept  Logical. Should we allow very high contamination fractions to be used. Passed to setContaminationFraction. Default FALSE. See ?SoupX::autoEstCont.

adjustMethod  Character. Method to use for correction. One of 'subtraction', 'soupOnly', or 'multinomial'. Default 'subtraction'. See ?SoupX::adjustCounts.

roundToInt  Logical. Should the resulting matrix be rounded to integers? Default FALSE. See ?SoupX::adjustCounts.

tol  Numeric. Allowed deviation from expected number of soup counts. Don’t change this. Default 0.001. See ?SoupX::adjustCounts.
Run TSCAN to obtain pseudotime values for cells

Description

Wrapper for obtaining a pseudotime ordering of the cells by projecting them onto the minimum spanning tree (MST)

Usage

```r
runTSCAN(
  inSCE,
  useReducedDim = "PCA",
  cluster = NULL,
  starter = NULL,
  seed = 12345
)
```

Examples

```r
## Not run:
# SoupX does not work for toy example,
sce <- importExampleData("pbmc3k")
sce <- runSoupX(sce, sample = "sample")
plotSoupXResults(sce, sample = "sample")
```

Value

The input inSCE object with `soupX_numIs`, `soupX_clusters`, `soupX_contamination` appended to `colData` slot; `soupX_{sample}_est` and `soupX_{sample}_counts` for each sample appended to `rowData` slot; and other computational metrics at `getSoupX(inSCE)`. Replace "soupX" to "soupX_bg" when background is used.

Author(s)

Yichen Wang

See Also

plotSoupXResults

pCut

Numeric. The p-value cut-off used when method = 'soupOnly'. Default 0.01. See ?SoupX::adjustCounts.
Run TSCAN Cluster DE Analysis

**Arguments**

- **inSCE**: Input `SingleCellExperiment` object.
- **useReducedDim**: Character. A low-dimension representation in `reducedDims`, will be used for both clustering if `cluster` not specified and MST construction. Default "PCA".
- **cluster**: Grouping for each cell in `inSCE`. A vector with equal length to the number of the cells in `inSCE`, or a single character for retrieving `colData` variable. Default `NULL`, will run `runScranSNN` to obtain.
- **starter**: Character. Specifies the starting node from which to compute the pseudotime. Default `NULL`, will select an arbitrary node.
- **seed**: An integer. Random seed for clustering if `cluster` is not specified. Default 12345.

**Value**

The input `inSCE` object with pseudotime ordering of the cells along the paths and the cluster label stored in `colData`, and other unstructured information in `metadata`.

**Author(s)**

Nida Pervaiz

**Examples**

```r
data("mouseBrainSubsetSCE", package = "singleCellTK")
mouseBrainSubsetSCE <- runTSCAN(inSCE = mouseBrainSubsetSCE,
                              useReducedDim = "PCA_logcounts")
```

**Description**

This function finds all paths that root from a given cluster `useCluster`, and performs tests to identify significant features for each path, and are not significant and/or changing in the opposite direction in the other paths. Using a branching cluster (i.e. a node with degree > 2) may highlight features which are responsible for the branching event. MST has to be pre-calculated with `runTSCAN`.

**Usage**

```r
runTSCANClusterDEAnalysis(
    inSCE,  
    useCluster,  
    useAssay = "logcounts",  
    fdrThreshold = 0.05  
)
```
**runTSCANDEG**

Test gene expression changes along a TSCAN trajectory path

**Description**

Wrapper for identifying genes with significant changes with respect to one of the TSCAN pseudo-time paths

**Usage**

```r
runTSCANDEG(inSCE, pathIndex, useAssay = "logcounts", discardCluster = NULL)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inSCE</td>
<td>Input <code>SingleCellExperiment</code> object.</td>
</tr>
<tr>
<td>pathIndex</td>
<td>Path index for which the pseudotime values should be used. This corresponds to the terminal node of specific path from the root node to the terminal node. Run listTSCANTerminalNodes(inSCE) for available options.</td>
</tr>
<tr>
<td>useAssay</td>
<td>Character. The name of the assay to use for testing the expression change. Should be log-normalized. Default &quot;logcounts&quot;</td>
</tr>
<tr>
<td>discardCluster</td>
<td>Cluster(s) which are not of use or masks other interesting effects can be discarded. Default NULL.</td>
</tr>
</tbody>
</table>

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inSCE</td>
<td>Input <code>SingleCellExperiment</code> object.</td>
</tr>
<tr>
<td>useCluster</td>
<td>The cluster to be regarded as the root, has to existing in colData(inSCE)$TSCAN_clusters.</td>
</tr>
<tr>
<td>useAssay</td>
<td>Character. The name of the assay to use. This assay should contain log normalized counts. Default &quot;logcounts&quot;.</td>
</tr>
<tr>
<td>fdrThreshold</td>
<td>Only output DEGs with FDR value smaller than this value. Default 0.05.</td>
</tr>
</tbody>
</table>

**Value**

The input inSCE with results updated in metadata.

**Author(s)**

Nida Pervaiz

**Examples**

```r
data("mouseBrainSubsetSCE", package = "singleCellTK")
mouseBrainSubsetSCE <- runTSCAN(inSCE = mouseBrainSubsetSCE, useReducedDim = "PCA_logcounts")
mouseBrainSubsetSCE <- runTSCANClusterDEAnalysis(inSCE = mouseBrainSubsetSCE, useCluster = 1)
```
runTSNE

Description

T-Stochastic Neighbour Embedding (t-SNE) algorithm is commonly for 2D visualization of single-cell data. This function wraps the Rtsne function.

With this function, users can create tSNE embedding directly from raw count matrix, with necessary preprocessing including normalization, scaling, dimension reduction all automated. Yet we still recommend having the PCA as input, so that the result can match with the clustering based on the same input PCA, and will be much faster.

Usage

```r
runtSNE(
  inSCE,
  useReducedDim = "PCA",
  useAssay = NULL,
  useAltExp = NULL,
  reducedDimName = "TSNE",
  logNorm = TRUE,
  useFeatureSubset = NULL,
  nTop = 2000,
  center = TRUE,
  scale = TRUE,
  pca = TRUE,
  partialPCA = FALSE,
  initialDims = 25,
  theta = 0.5,
  perplexity = 30,
  nIterations = 1000,
)```

Run t-SNE embedding with Rtsne method

Examples

data("mouseBrainSubsetSCE", package = "singleCellTK")
mouseBrainSubsetSCE <- runTSCAN(inSCE = mouseBrainSubsetSCE,
  useReducedDim = "PCA_logcounts")
terminalNodes <- listTSCANTerminalNodes(mouseBrainSubsetSCE)
mouseBrainSubsetSCE <- runTSCANDEG(inSCE = mouseBrainSubsetSCE,
  pathIndex = terminalNodes[1])

Value

The input `inSCE` with results updated in metadata.

Author(s)

Nida Pervaiz
runTSNE

numThreads = 1,
seed = 12345
)

runQuickTSNE(inSCE, useAssay = "counts", ...)

getTSNE(
inSCE,
useReducedDim = "PCA",
useAssay = NULL,
useAltExp = NULL,
reducedDimName = "TSNE",
logNorm = TRUE,
useFeatureSubset = NULL,
nTop = 2000,
center = TRUE,
scale = TRUE,
pca = TRUE,
partialPCA = FALSE,
initialDims = 25,
theta = 0.5,
perplexity = 30,
nIterations = 1000,
numThreads = 1,
seed = 12345
)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inSCE</td>
<td>Input SingleCellExperiment object.</td>
</tr>
<tr>
<td>useReducedDim</td>
<td>The low dimension representation to use for UMAP computation. Default &quot;PCA&quot;.</td>
</tr>
<tr>
<td>useAssay</td>
<td>Assay to use for tSNE computation. If useAltExp is specified, useAssay has</td>
</tr>
<tr>
<td></td>
<td>to exist in assays(altExp(inSCE, useAltExp)). Default NULL.</td>
</tr>
<tr>
<td>useAltExp</td>
<td>The subset to use for tSNE computation, usually for the selected.variable</td>
</tr>
<tr>
<td></td>
<td>features. Default NULL.</td>
</tr>
<tr>
<td>reducedDimName</td>
<td>a name to store the results of the dimension reductions. Default &quot;TSNE&quot;.</td>
</tr>
<tr>
<td>logNorm</td>
<td>Whether the counts will need to be log-normalized prior to generating the</td>
</tr>
<tr>
<td></td>
<td>tSNE via scaterLogNormCounts. Ignored when using useReducedDim. Default TRUE.</td>
</tr>
<tr>
<td>useFeatureSubset</td>
<td>Subset of feature to use for dimension reduction. A character string</td>
</tr>
<tr>
<td></td>
<td>indicating a rowData variable that stores the logical vector of HVG selection,</td>
</tr>
<tr>
<td></td>
<td>or a vector that can subset the rows of inSCE. Default NULL.</td>
</tr>
<tr>
<td>nTop</td>
<td>Automatically detect this number of variable features to use for dimension</td>
</tr>
<tr>
<td></td>
<td>reduction. Ignored when using useReducedDim or using useFeatureSubset.</td>
</tr>
<tr>
<td>center</td>
<td>Whether data should be centered before PCA is applied. Ignored when using</td>
</tr>
<tr>
<td></td>
<td>useReducedDim. Default TRUE.</td>
</tr>
</tbody>
</table>
scale: Whether data should be scaled before PCA is applied. Ignored when using `useReducedDim`. Default `TRUE`.

pca: Whether an initial PCA step should be performed. Ignored when using `useReducedDim`. Default `TRUE`.

partialPCA: Whether truncated PCA should be used to calculate principal components (requires the irlba package). This is faster for large input matrices. Ignored when using `useReducedDim`. Default `FALSE`.

initialDims: Number of dimensions from PCA to use as input in tSNE. Default 25.

theta: Numeric value for speed/accuracy trade-off (increase for less accuracy), set to 0.0 for exact TSNE. Default 0.5.

perplexity: perplexity parameter. Should not be bigger than $3 \times \text{perplexity} < \text{ncol(inSCE)} - 1$. Default 30. See Rtsne details for interpretation.

nIterations: maximum iterations. Default 1000.

numThreads: Integer, number of threads to use using OpenMP. Default 1. 0 corresponds to using all available cores.

seed: Random seed for reproducibility of tSNE results. Default `NULL` will use global seed in use by the R environment.

...: Other parameters to be passed to runTSNE

Value

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

Examples

data(scExample, package = "singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
# Run from raw counts
sce <- runQuickTSNE(sce)
## Not run:
# Run from PCA
sce <- scaterlogNormCounts(sce, "logcounts")
sce <- runModelGeneVar(sce)
sce <- setTopHVG(sce, method = "modelGeneVar", hvgNumber = 2000,
  featureSubsetName = "HVG_modelGeneVar2000")
sce <- scaterPCA(sce, useAssay = "logcounts",
  useFeatureSubset = "HVG_modelGeneVar2000", scale = TRUE)
sce <- runTSNE(sce, useReducedDim = "PCA")

## End(Not run)
runUMAP

Run UMAP embedding with scater method

Description

Uniform Manifold Approximation and Projection (UMAP) algorithm is commonly for 2D visualization of single-cell data. These functions wrap the scater `calculateUMAP` function.

Users can use `runQuickUMAP` to directly create UMAP embedding from raw count matrix, with necessary preprocessing including normalization, variable feature selection, scaling, dimension reduction all automated. Therefore, `useReducedDim` is disabled for `runQuickUMAP`.

In a complete analysis, we still recommend having dimension reduction such as PCA created beforehand and select proper numbers of dimensions for using `runUMAP`, so that the result can match with the clustering based on the same input PCA.

Usage

```r
runUMAP(
  inSCE,
  useReducedDim = "PCA",
  useAssay = NULL,
  useAltExp = NULL,
  sample = NULL,
  reducedDimName = "UMAP",
  logNorm = TRUE,
  useFeatureSubset = NULL,
  nTop = 2000,
  scale = TRUE,
  pca = TRUE,
  initialDims = 25,
  nNeighbors = 30,
  nIterations = 200,
  alpha = 1,
  minDist = 0.01,
  spread = 1,
  seed = 12345,
  verbose = TRUE,
  BPPARAM = SerialParam()
)
```

```r
runQuickUMAP(inSCE, useAssay = "counts", sample = "sample", ...)
```

```r
getUMAP(
  inSCE,
  useReducedDim = "PCA",
  useAssay = NULL,
  useAltExp = NULL,
)```
sample = NULL,  
reducedDimName = "UMAP",  
logNorm = TRUE,  
useFeatureSubset = NULL,  
nTop = 2000,  
scale = TRUE,  
pca = TRUE,  
initialDims = 25,  
nNeighbors = 30,  
nIterations = 200,  
alpha = 1,  
minDist = 0.01,  
spread = 1,  
seed = 12345,  
BPPARAM = SerialParam()
)

Arguments

Arguments

inSCE  
Input SingleCellExperiment object.

useReducedDim  
The low dimension representation to use for UMAP computation. If useAltExp is specified, useReducedDim has to exist in reducedDims(altExp(inSCE, useAltExp)). Default "PCA".

useAssay  
Assay to use for UMAP computation. If useAltExp is specified, useAssay has to exist in assays(altExp(inSCE, useAltExp)). Ignored when using useReducedDim. Default NULL.

useAltExp  
The subset to use for UMAP computation, usually for the selected variable features. Default NULL.

sample  
Character vector. Indicates which sample each cell belongs to. If given a single character, will take the annotation from colData. Default NULL.

reducedDimName  
A name to store the results of the UMAP embedding coordinates obtained from this method. Default "UMAP".

logNorm  
Whether the counts will need to be log-normalized prior to generating the UMAP via scaterLogNormCounts. Ignored when using useReducedDim. Default TRUE.

useFeatureSubset  
Subset of feature to use for dimension reduction. A character string indicating a rowData variable that stores the logical vector of HVG selection, or a vector that can subset the rows of inSCE. Default NULL.

nTop  
Automatically detect this number of variable features to use for dimension reduction. Ignored when using useReducedDim or using useFeatureSubset. Default 2000.

tscale  
Whether useAssay matrix will need to be standardized. Default TRUE.

pca  
Logical. Whether to perform dimension reduction with PCA before UMAP. Ignored when using useReducedDim. Default TRUE.

initialDims  
Number of dimensions from PCA to use as input in UMAP. Default 25.
runVAM

nNeighbors  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 `calculateUMAP` for more information.

nIterations  The number of iterations performed during layout optimization. Default is 200.

alpha  The initial value of "learning rate" of layout optimization. Default is 1.

minDist  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 `calculateUMAP` for more information.

spread  The effective scale of embedded points. In combination with minDist, this determines how clustered/clumped the embedded points are. Default 1. See `calculateUMAP` for more information.

seed  Random seed for reproducibility of UMAP results. Default NULL will use global seed in use by the R environment.

verbose  Logical. Whether to print log messages. Default TRUE.

BPPARAM  A `BiocParallelParam` object specifying whether the PCA should be parallelized.

...  Parameters passed to `runUMAP`

Value

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

Examples

data(scExample, package = "singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
# Run from raw counts
sce <- runQuickUMAP(sce)
plotDimRed(sce, "UMAP")

---

runVAM  

Run VAM to score gene sets in single cell data

Description

Wrapper for the Variance-adjusted Mahalanobis (VAM), which is a fast and accurate method for cell-specific gene set scoring of single cell data. This algorithm computes distance statistics and one-sided p-values for all cells in the specified single cell gene expression matrix. Gene sets should already be imported and stored in the meta data using functions such as `importGeneSetsFromList` or `importGeneSetsFromMSigDB`
runVAM

Usage

runVAM(
  inSCE,
  geneSetCollectionName,
  useAssay,
  resultNamePrefix = NULL,
  center = FALSE,
  gamma = TRUE
)

Arguments

inSCE Input SingleCellExperiment object.

geneSetCollectionName Character. The name of the gene set collection to use.

useAssay Character. The name of the assay to use. This assay should contain log normalized counts.

resultNamePrefix Character. Prefix to the name the VAM results which will be stored in the reducedDim slot of inSCE. The names of the output matrices will be resultNamePrefix_Distance and resultNamePrefix_CDF. If this parameter is set to NULL, then "VAM_geneSetCollectionName_" will be used. Default NULL.

center Boolean. If TRUE, values will be mean centered when computing the Mahalanobis statistic. Default FALSE.

gamma Boolean. If TRUE, a gamma distribution will be fit to the non-zero squared Mahalanobis distances computed from a row-permuted version of the gene expression matrix. The estimated gamma distribution will be used to compute a one-sided p-value for each cell. If FALSE, the p-value will be computed using the standard chi-square approximation for the squared Mahalanobis distance (or non-central if center = FALSE). Default TRUE.

Value

A SingleCellExperiment object with VAM metrics stored in reducedDim as VAM_NameOfTheGeneset_Distance and VAM_NameOfTheGeneset_CDF.

Author(s)

Nida Pervaiz

See Also

importGeneSetsFromList, importGeneSetsFromMSigDB, importGeneSetsFromGMT, importGeneSetsFromCollection for importing gene sets. sctkListGeneSetCollections, getPathwayResultNames and getGenesetNamesFromCollection for available related information in inSCE.
**Examples**

```r
data(scExample, package = "singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
sce <- scaterlogNormCounts(sce, assayName = "logcounts")
gs1 <- rownames(sce)[seq(10)]
gs2 <- rownames(sce)[seq(11, 20)]
gs <- list("geneset1" = gs1, "geneset2" = gs2)
sce <- importGeneSetsFromList(inSCE = sce, geneSetList = gs,
                             by = "rownames")
sce <- runVAM(inSCE = sce,
             geneSetCollectionName = "GeneSetCollection",
             useAssay = "logcounts")
```

---

**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**

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

**Arguments**

- **inSCE**: Input `SingleCellExperiment` object
- **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".
sampleSummaryStats

Generate table of SCTK QC outputs.

Description

Creates a table of QC metrics generated from QC algorithms, which is stored within the metadata slot of the input SingleCellExperiment object.

Usage

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

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.
statsName Character. The name of the slot that will store the QC stat table. Default "qc_table".

Value


Examples

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

scaterCPM Uses CPM from scater library to compute counts-per-million.

Description

scaterCPM Uses CPM from scater library to compute counts-per-million.

Usage

scaterCPM(inSCE, assayName = "ScaterCPMCounts", useAssay = "counts")

Arguments

inSCE Input SingleCellExperiment object
assayName New assay name for cpm data.
useAssay Input assay

Value

inSCE Updated SingleCellExperiment object

Author(s)

Irzam Sarfraz
Examples

```r
data(sce_chcl, package = "scds")
sce_chcl <- scaterCPM(sce_chcl, "countsCPM", "counts")
```

```
sccaterlogNormCounts <- scaterlogNormCounts(sce_chcl, "logcounts", "counts")
```

Description

`sccaterlogNormCounts` Uses `logNormCounts` to log normalize input data

Usage

```r
scaterlogNormCounts(
  inSCE,
  assayName = "ScaterLogNormCounts",
  useAssay = "counts"
)
```

Arguments

- `inSCE` Input SingleCellExperiment object
- `assayName` 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

```r
data(sce_chcl, package = "scds")
sce_chcl <- scaterlogNormCounts(sce_chcl, "logcounts", "counts")
```
scaterPCA

Perform scater PCA on a SingleCellExperiment Object

Description

A wrapper to runPCA function to compute principal component analysis (PCA) from a given SingleCellExperiment object.

Usage

scaterPCA(
  inSCE,
  useAssay = "logcounts",
  useFeatureSubset = NULL,
  scale = TRUE,
  reducedDimName = "PCA",
  nComponents = 50,
  ntop = 2000,
  useAltExp = NULL,
  seed = 12345,
  BPPARAM = BiocParallel::SerialParam()
)

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"

useFeatureSubset Subset of feature to use for dimension reduction. A character string indicating a rowData variable that stores the logical vector of HVG selection, or a vector that can subset the rows of inSCE. Default NULL.

scale Logical scalar, whether to standardize the expression values. Default TRUE.

reducedDimName Name to use for the reduced output assay. Default "PCA".

nComponents Number of principal components to obtain from the PCA computation. Default 50.

ntop Automatically detect this number of variable features to use for dimension reduction. Ignored when using useReducedDim or using useFeatureSubset. Default 2000.

useAltExp The subset to use for PCA computation, usually for the selected.variable features. Default NULL.

seed Integer, random seed for reproducibility of PCA results. Default NULL.

BPPARAM A BiocParallelParam object specifying whether the PCA should be parallelized.
Value

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

Examples

data(scExample, package = "singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
sce <- scaterlogNormCounts(sce, "logcounts")
sce <- scaterPCA(sce, "logcounts", scale = TRUE)

Description

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

data("scExample")

Format

A SingleCellExperiment object.

Value

Example Single Cell RNA-Seq data in SingleCellExperiment Object, subset of 10x public dataset

Examples

data("scExample")
### sceBatches

**Example Single Cell RNA-Seq data in SingleCellExperiment object, with different batches annotated**

#### 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.

#### Usage

```r
data('sceBatches')
```

#### Format

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

#### Value

Example Single Cell RNA-Seq data in SingleCellExperiment object, with different batches annotated

---

### sctkListGeneSetCollections

**Lists imported GeneSetCollections**

#### Description

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

#### Usage

```r
sctkListGeneSetCollections(inSCE)
```

#### Arguments

- `inSCE` A `SingleCellExperiment` object.

#### Value

Character vector.
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", "louvain", "leidenalg", "bbknn", "scanorama", "anndata"),
    selectConda = TRUE,
    forge = FALSE,
pipIgnoreInstalled = TRUE,
pythonVersion = NULL,
...
)

Arguments

envname Character. Name of the conda environment to create.
conda Character. Path to conda executable. Use "auto" to find conda using the PATH and other conventional install locations. Default 'auto'.
packages Character Vector. List of packages to install from Conda.
pipPackages Character Vector. List of packages to install into the Conda environment using 'pip'.
selectConda Boolean. Run selectSCTKConda after installing all packages to select the Conda environment. Default TRUE.
forge Boolean. Include the Conda Forge repository.
pipIgnoreInstalled 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).
pythonVersion Passed to python_version variable in conda_install. Default NULL.
...
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**

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

**Arguments**

- `envname` Character. Name of the virtual environment to create.
- `packages` Character Vector. List of packages to install.
- `selectEnvironment` Boolean. Run `selectSCTKVirtualEnvironment` after installing all packages to select the virtual environment. Default TRUE.
- `python` 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

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

SEG

Stably Expressed Gene (SEG) list object, with SEG sets for human and mouse.

Description

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

Usage

data('SEG')

Format

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

Value

Stably Expressed Gene (SEG) list object, with SEG sets for human and mouse.

Source

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

Examples

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

selectSCTKConda

Selects a Conda environment

Description

Selects a Conda environment with Python packages used in singleCellTK.

Usage

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

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

```r
## 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

```r
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

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

## End(Not run)
```

### Description

Users can set rownames of an SCE object with either a character vector where the length equals to `nrow(x)`, or a single character specifying a column in `rowData(x)`. Also applicable to matrix like object where `rownames<-` method works, but only allows full size name vector. Users can set `dedup = TRUE` to remove duplicated entries in the specification, by adding `-1, -2, ..., -i` suffix to the duplication of the same identifier.

### Usage

```r
setRowNames(x, rowNames, dedup = TRUE)
```

### Arguments

- **x**
  - Input object where the rownames will be modified.
- **rowNames**
  - Character vector of the rownames. If `x` is an `SingleCellExperiment` object, a single character specifying a column in `rowData(x)`.
- **dedup**
  - Logical. Whether to deduplicate the specified `rowNames`. Default `TRUE`

### Value

The input SCE object with rownames updated.

### Examples

```r
data("scExample", package = "singleCellTK")
head(rownames(sce))
sce <- setRowNames(sce, "feature_name")
head(rownames(sce))
```
setSCTKDisplayRow  Indicates which rowData to use for visualization

Description
This function is to be used to specify which

Usage
setSCTKDisplayRow(inSCE, featureDisplayRow)

Arguments
inSCE  Input SingleCellExperiment object with saved dimension reduction components or a variable with saved results. Required.
featureDisplayRow  Indicates which column name of rowData to be used for plots.

Value
A SingleCellExperiment object with the specific column name of rowData to be used for plotting stored in metadata.

Examples
data(scExample, package="singleCellTK")
sce <- subsetSCECols(sce, colData = "type != 'EmptyDroplet'")
sce <- setSCTKDisplayRow(inSCE = sce, featureDisplayRow = "feature_name")
plotSCEViolinAssayData(inSCE = sce, feature = "ENSG00000019582")

description
Run the single cell analysis app

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

```r
## Not run:
# Upload data through the app
singleCellTK()

# Load the app with a SingleCellExperiment object
data("mouseBrainSubsetSCE")
singleCellTK(mouseBrainSubsetSCE)

## End(Not run)
```

---

**subDiffEx**  
*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.*

---

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

```r
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)][seq(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)][seq(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,]
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

```r
subsetSCERows(
  inSCE,
  index = NULL,
  bool = NULL,
  rowData = NULL,
  returnAsAltExp = TRUE,
  altExpName = "subset",
  prependAltExpName = TRUE
)
```

Arguments

- **inSCE**  
  Input SingleCellExperiment object.

- **index**  
  Integer vector. Vector of indicies indicating which rows to keep. If `NULL`, this will not be used for subsetting. Default `NULL`.

- **bool**  
  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 `inSCE`. If `NULL`, this will not be used for subsetting. Default `NULL`.

- **rowData**  
  Character. An expression that will identify a subset of rows using variables found in the `rowData` of `inSCE`. For example, if `x` is a numeric vector in `rowData`, then "x < 5" will return all rows with `x` less than 5. Single quotes should be used for character strings. For example, "y == 'yes'" will return all rows 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`.

- **returnAsAltExp**  
  Boolean. If `TRUE`, the subsetted SingleCellExperiment object will be returned in the `altExp` slot of `inSCE`. If `FALSE`, the subsetted SingleCellExperiment object will be directly returned.

- **altExpName**  
  Character. Name of the alternative experiment object to add if `returnAsAltExp` = `TRUE`. Default `subset`.

- **prependAltExpName**  
  Boolean. If `TRUE`, `altExpName` will be added to the beginning of the assay names in the `altExp` object. This is only utilized if `returnAsAltExp` = `TRUE`. Default `TRUE`.
summarizeSCE

Value

A SingleCellExperiment object that has been subselected 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)
**trimCounts**

**Trim Counts**

**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**

\[
\text{trimCounts}(\text{counts}, \text{trimValue} = c(10, -10))
\]

**Arguments**

- `counts`: matrix

**Value**

trimmed counts matrix

**Examples**

```r
data(sce_chcl, package = "scds")
assay(sce_chcl, "countsTrimmed") <- trimCounts(assay(sce_chcl, "counts"), c(10, -10))
```
Index

* datasets
  MitoGenes, 79
  mouseBrainSubsetSCE, 79
  msigdb_table, 80
  sce, 300
  sceBatches, 301
  SEG, 305
  addPerCellQC, 247, 249
  adjustCounts, 283
  altExp, 263, 312
  assay, 43, 217, 242, 243, 250, 261, 263, 297
  autoEstCont, 283
  barcodeRanks, 210
  bcds, 212, 213
  BiocParallelParam, 234, 244, 248, 262, 263, 293, 296, 299
  buildSNNGraph, 262
  calcEffectSizes, 8
  calculateUMAP, 291, 293
  cluster_fast_greedy, 264
  cluster_infomap, 264
  cluster_label_prop, 264
  cluster_leading_eigen, 264
  cluster_leiden, 264
  cluster_louvain, 264
  cluster_walktrap, 264
  colorRamp2, 135
  combineSCE, 8
  computeHeatmap, 9
  computeZScore, 10
  conda_create, 303
  conda_install, 302, 303
  constructSCE, 11
  convertSCEToSeurat, 11
  convertSeuratToSCE, 13
cdxs, 217, 218
  cdxs_bcds_hybrid, 219
data.table, 11
  DataFrame-class, 248
  dbscan, 225, 226
decontX, 15, 224, 225
dedupRowNames, 14
  DelayedArray, 49, 50, 55, 57–60, 62, 71–73, 75, 187
  DelayedArray-class, 52
detectCellOutlier, 14
diffAbundanceFET, 16, 38
  DimHeatmap, 271
discreteColorPalette, 17
distinctColors, 17, 17
downSampleCells, 18
downSampleDepth, 20
  emptyDrops, 72, 230, 231
  estimateNonExpressingCells, 284
  expData, 21
  expData, ANY, character-method, 22
  expData<-, 22
  expData<-, ANY, character, CharacterOrNullOrMissing, logical-method, 23
  expDataNames, 24
  expDataNames, ANY-method, 24
  expDeleteDataTag, 25
  exportSCE, 26
  exportSCEtoAnnData, 27
  exportSCEtoFlatFile, 28
  exportSCEToSeurat, 29
  expSetDataTag, 30
  expTaggedData, 30
  fastMNN, 234
  featureIndex, 31, 63, 65, 67, 68, 70, 247
FindIntegrationAnchors, 274
findMarkerDiffExp (runFindMarker), 236
findMarkerTopTable
  (getFindMarkerTopTable), 39
fit_dirichlet, 226
generateHTANMeta, 33
generateMeta, 33
generateSimulatedData, 34
GeneSetCollection, 63–67, 69, 70, 302
getBiomarker, 35
generateDEGTopTable, 36
generateDiffAbundanceResults
  (getDiffAbundanceResults), 37
generateDiffAbundanceResults, SingleCellExperiment-method
  (getDiffAbundanceResults), 37
generateDiffAbundanceResults<- (getDiffAbundanceResults), 37
generateDiffAbundanceResults<-, SingleCellExperiment-method
  (getDiffAbundanceResults), 37
generateEnrichRResult, 233
generateEnrichRResult (getEnrichRResult), 38
generateEnrichRResult<-(getEnrichRResult<-), 38
generateEnrichRResult<-, SingleCellExperiment-method
  (getEnrichRResult<-), 38
generateEnrichRResult<-, SingleCellExperiment-method
  (getEnrichRResult<-), 38
generateFindMarkerTopTable, 39, 111, 237
genGenesetNamesFromCollection, 40, 294
genGmt, 65, 248
genMSigDBTable, 41
genPathwayResultNames, 42, 294
genSampleSummaryStatsTable, 42
genSampleSummaryStatsTable, SingleCellExperiment-method
  (getSampleSummaryStatsTable), 42
geneSeuratVariableFeatures, 44
geneSoupX (geneSoupX<-), 45
geneSoupX, SingleCellExperiment-method
  (geneSoupX<-), 45
geneSoupX<-, 45
geneSoupX<-, SingleCellExperiment-method
  (geneSoupX<-), 45
geneTopHVG, 46, 176, 235, 245, 253, 269
geneTSCANResults<-, SingleCellExperiment-method
  (geneTSCANResults), 48
geneTSCANResults (geneTSCANResults), 48
geneTSCANResults<-(geneTSCANResults), 48
geneTSNE, 228
geneUMAP, 288
geneTSNE (runTSNE), 288
geneUMAP (runUMAP), 288
geneClustering, 89, 90, 98, 155
geneClustering (grep), 32, 209
geneGSEABase, 63, 65, 67, 69, 70
geneHarmonyMatrix, 240
geneHeatmap, 111, 155
geneigraph, 263
geneimportAlevin, 49
  method (geneimportAnnData), 50
geneimportBUStools, 51
geneimportCellRanger, 53
geneimportCellRangerV2 (importCellRanger),
geneimportCellRangerV2Sample, 56
geneimportCellRangerV3 (importCellRanger), 53
geneimportCellRangerV3Sample, 57
geneimportDropEst, 58
geneimportExampleData, 60
geneimportFromFiles, 61
geneimportGeneSetsFromCollection, 63, 65, 67, 294
geneimportGeneSetsFromGmt, 64, 64, 67, 69, 70, 294, 302
geneimportGeneSetsFromList, 64, 65, 66, 69, 70, 293, 294, 302
geneimportGeneSetsFromMSigDB, 41, 64, 65, 67, 68, 80, 293, 294, 302
geneimportMultipleSources, 71
geneimportOptimus, 71
geneimportSEQC, 73
geneimportSTARsolo, 74
IntegrateData, 274
geneisOutlier, 14
geneiterateSimulations, 76
genekmeans, 241

gene list.dirs, 54
gene listSampleSummaryStatsTables, 77
gene listSampleSummaryStatsTables, SingleCellExperiment-method
  (listSampleSummaryStatsTables), 77
<table>
<thead>
<tr>
<th>Function Call</th>
<th>Page Numbers</th>
</tr>
</thead>
<tbody>
<tr>
<td>listTSCANResults</td>
<td>48</td>
</tr>
<tr>
<td>listTSCANResults, SingleCellExperiment-method</td>
<td></td>
</tr>
<tr>
<td>listTSCANTerminalNodes</td>
<td>48</td>
</tr>
<tr>
<td>listTSCANTerminalNodes, SingleCellExperiment-method</td>
<td></td>
</tr>
<tr>
<td>logNormCounts</td>
<td>298</td>
</tr>
<tr>
<td>Matrix</td>
<td>60</td>
</tr>
<tr>
<td>matrix</td>
<td>49, 50, 52, 55, 57–59, 62, 72, 73, 75</td>
</tr>
<tr>
<td>mergeSCEColData</td>
<td>78</td>
</tr>
<tr>
<td>metadata</td>
<td>64, 65, 67, 69, 70, 77</td>
</tr>
<tr>
<td>MitoGenes</td>
<td>79</td>
</tr>
<tr>
<td>mnnCorrect</td>
<td>244</td>
</tr>
<tr>
<td>modelGeneVar</td>
<td>226, 244</td>
</tr>
<tr>
<td>mouseBrainSubsetSCE</td>
<td>79</td>
</tr>
<tr>
<td>msigdb_table</td>
<td>80</td>
</tr>
<tr>
<td>msigdb_show_species</td>
<td>68</td>
</tr>
<tr>
<td>multiBatchPCA</td>
<td>234</td>
</tr>
<tr>
<td>NestorowaHSCData</td>
<td>60</td>
</tr>
<tr>
<td>plotBarcodeRankDropsResults</td>
<td>80, 83, 210</td>
</tr>
<tr>
<td>plotBarcodeRankScatter</td>
<td>81</td>
</tr>
<tr>
<td>plotBatchCorrCompare</td>
<td>83</td>
</tr>
<tr>
<td>plotBatchVariance</td>
<td>84</td>
</tr>
<tr>
<td>plotBcdsResults</td>
<td>85, 213</td>
</tr>
<tr>
<td>plotBubble</td>
<td>88</td>
</tr>
<tr>
<td>plotClusterAbundance</td>
<td>89</td>
</tr>
<tr>
<td>plotCdxsResults</td>
<td>90, 218</td>
</tr>
<tr>
<td>plotDecontXResults</td>
<td>93</td>
</tr>
<tr>
<td>plotDEGHeatmap</td>
<td>96, 101, 224</td>
</tr>
<tr>
<td>plotDEGRegression</td>
<td>98, 224</td>
</tr>
<tr>
<td>plotDEGViolin</td>
<td>99, 224</td>
</tr>
<tr>
<td>plotDEGVolcano</td>
<td>101, 224</td>
</tr>
<tr>
<td>plotDimRed</td>
<td>102</td>
</tr>
<tr>
<td>plotDoubletFinderResults</td>
<td>103, 229</td>
</tr>
<tr>
<td>plotEmptyDropsResults</td>
<td>105, 109, 231</td>
</tr>
<tr>
<td>plotEmptyDropsScatter</td>
<td>107, 107, 231</td>
</tr>
<tr>
<td>plotFindMarkerHeatmap</td>
<td>40, 109, 237</td>
</tr>
<tr>
<td>plotMarkerDiffExp</td>
<td></td>
</tr>
<tr>
<td>(plotFindMarkerHeatmap)</td>
<td>109</td>
</tr>
<tr>
<td>plotMASTThresholdGenes</td>
<td>112</td>
</tr>
<tr>
<td>plotPathway</td>
<td>113</td>
</tr>
<tr>
<td>plotPCA</td>
<td>114</td>
</tr>
<tr>
<td>plotRunPerCellQCResults</td>
<td>115</td>
</tr>
<tr>
<td>plotScanpyDotPlot</td>
<td>117</td>
</tr>
<tr>
<td>plotScanpyEmbedding</td>
<td>119</td>
</tr>
<tr>
<td>plotScanpyHeatmap</td>
<td>120</td>
</tr>
<tr>
<td>plotScanpyHVG</td>
<td>121</td>
</tr>
<tr>
<td>plotScanpyMarkerGenes</td>
<td>122</td>
</tr>
<tr>
<td>plotScanpyMarkerGenesDotPlot</td>
<td>123</td>
</tr>
<tr>
<td>plotScanpyMarkerGenesHeatmap</td>
<td>124</td>
</tr>
<tr>
<td>plotScanpyMarkerGenesMatrixPlot</td>
<td>125</td>
</tr>
<tr>
<td>plotScanpyMarkerGenesViolin</td>
<td>127</td>
</tr>
<tr>
<td>plotScanpyMatrixPlot</td>
<td>128</td>
</tr>
<tr>
<td>plotScanpyPCA</td>
<td>129</td>
</tr>
<tr>
<td>plotScanpyPCAGeneRanking</td>
<td>130</td>
</tr>
<tr>
<td>plotScanpyPCAVariance</td>
<td>131</td>
</tr>
<tr>
<td>plotScanpyViolin</td>
<td>131</td>
</tr>
<tr>
<td>plotScDb1FinderResults</td>
<td>132, 261</td>
</tr>
<tr>
<td>plotScdsHybridResults</td>
<td>135</td>
</tr>
<tr>
<td>plotSCEBarAssayData</td>
<td>138</td>
</tr>
<tr>
<td>plotSCEBarColData</td>
<td>140</td>
</tr>
<tr>
<td>plotSCEBatchFeatureMean</td>
<td>141</td>
</tr>
<tr>
<td>plotSCEDensity</td>
<td>142</td>
</tr>
<tr>
<td>plotSCEDensityAssayData</td>
<td>144</td>
</tr>
<tr>
<td>plotSCEDensityColData</td>
<td>145</td>
</tr>
<tr>
<td>plotSCEDimReduceColData</td>
<td>147, 175</td>
</tr>
<tr>
<td>plotSCEDimReduceFeatures</td>
<td>150</td>
</tr>
<tr>
<td>plotSCEHeatmap</td>
<td>97, 98, 111, 152</td>
</tr>
<tr>
<td>plotSCEScatter</td>
<td>155</td>
</tr>
<tr>
<td>plotSCEvioin</td>
<td>158</td>
</tr>
<tr>
<td>plotSCEvioinAssayData</td>
<td>160</td>
</tr>
<tr>
<td>plotSCEvioinColData</td>
<td>163</td>
</tr>
<tr>
<td>plotScrubletResults</td>
<td>165, 266</td>
</tr>
<tr>
<td>plotSeuratElbow</td>
<td>168</td>
</tr>
<tr>
<td>plotSeuratGenes</td>
<td>169</td>
</tr>
<tr>
<td>plotSeuratHeatmap</td>
<td>170</td>
</tr>
<tr>
<td>plotSeuratHVG</td>
<td>171</td>
</tr>
<tr>
<td>plotSeuratJackStraw</td>
<td>171</td>
</tr>
<tr>
<td>plotSeuratReduction</td>
<td>172</td>
</tr>
<tr>
<td>plotSoupXResults</td>
<td>173</td>
</tr>
<tr>
<td>plotTopHVG</td>
<td>47, 175, 235, 245, 253, 269</td>
</tr>
<tr>
<td>plotTSCANClusterDEG</td>
<td>177</td>
</tr>
<tr>
<td>plotTSCANClusterPseudo</td>
<td>178</td>
</tr>
<tr>
<td>plotTSCANDimReduceFeatures</td>
<td>179</td>
</tr>
<tr>
<td>plotTSCANPseudotimeGenes</td>
<td>180</td>
</tr>
<tr>
<td>plotTSCANPseudotimeHeatmap</td>
<td>182</td>
</tr>
<tr>
<td>plotTSCANResults</td>
<td>183</td>
</tr>
<tr>
<td>plotTSNE</td>
<td>184</td>
</tr>
<tr>
<td>plotUMAP</td>
<td>185</td>
</tr>
<tr>
<td>qcInputProcess</td>
<td>186</td>
</tr>
<tr>
<td>quickCluster</td>
<td>284</td>
</tr>
<tr>
<td>Function</td>
<td>References</td>
</tr>
<tr>
<td>----------</td>
<td>------------</td>
</tr>
<tr>
<td>readMM</td>
<td>49, 50, 52, 55, 57–59, 62, 72, 73, 75</td>
</tr>
<tr>
<td>readSingleCellMatrix</td>
<td>187</td>
</tr>
<tr>
<td>reducedDim</td>
<td>263</td>
</tr>
<tr>
<td>reducedMNN</td>
<td>234</td>
</tr>
<tr>
<td>reportCellQC</td>
<td>188</td>
</tr>
<tr>
<td>reportClusterAbundance</td>
<td>189</td>
</tr>
<tr>
<td>reportDiffAbundanceFET</td>
<td>190</td>
</tr>
<tr>
<td>reportDiffExp</td>
<td>191</td>
</tr>
<tr>
<td>reportDropletQC</td>
<td>192</td>
</tr>
<tr>
<td>reportFindMarker</td>
<td>193</td>
</tr>
<tr>
<td>reportQCTool</td>
<td>193</td>
</tr>
<tr>
<td>reportSeurat</td>
<td>194</td>
</tr>
<tr>
<td>reportSeuratClustering</td>
<td>197</td>
</tr>
<tr>
<td>reportSeuratDimRed</td>
<td>198</td>
</tr>
<tr>
<td>reportSeuratFeatureSelection</td>
<td>200</td>
</tr>
<tr>
<td>reportSeuratMarkerSelection</td>
<td>201</td>
</tr>
<tr>
<td>reportSeuratNormalization</td>
<td>202</td>
</tr>
<tr>
<td>reportSeuratResults</td>
<td>203</td>
</tr>
<tr>
<td>reportSeuratRun</td>
<td>205</td>
</tr>
<tr>
<td>reportSeuratScaling</td>
<td>208</td>
</tr>
<tr>
<td>ReprocessedAllenData</td>
<td>60</td>
</tr>
<tr>
<td>ReprocessedFluidigmData</td>
<td>60</td>
</tr>
<tr>
<td>retrieveFeatureIndex</td>
<td>209</td>
</tr>
<tr>
<td>retrieveFeatureInfo</td>
<td>32</td>
</tr>
<tr>
<td>retrieveSCEIndex</td>
<td>209</td>
</tr>
<tr>
<td>Rtsne</td>
<td>288, 290</td>
</tr>
<tr>
<td>runANOVA</td>
<td>(runDEAnalysis), 220</td>
</tr>
<tr>
<td>runBarcodeRankDrops</td>
<td>81–83, 210</td>
</tr>
<tr>
<td>runBBKNN</td>
<td>211</td>
</tr>
<tr>
<td>runBcds</td>
<td>85, 86, 88, 212</td>
</tr>
<tr>
<td>runCellQC</td>
<td>64, 65, 67, 69, 70, 213, 213, 213, 218, 229, 249, 261, 266</td>
</tr>
<tr>
<td>runClusterSummaryMetrics</td>
<td>215</td>
</tr>
<tr>
<td>runComBatSeq</td>
<td>216</td>
</tr>
<tr>
<td>runCxds</td>
<td>90–92, 166, 217</td>
</tr>
<tr>
<td>runCxdsBcdsHybrid</td>
<td>136, 219</td>
</tr>
<tr>
<td>runDEAnalysis</td>
<td>101, 191, 220, 237</td>
</tr>
<tr>
<td>runDecontX</td>
<td>94, 224</td>
</tr>
<tr>
<td>runDESeq2</td>
<td>(runDEAnalysis), 220</td>
</tr>
<tr>
<td>runDimReduce</td>
<td>227</td>
</tr>
<tr>
<td>runDoubletFinder</td>
<td>104, 105, 228</td>
</tr>
<tr>
<td>runDropletQC</td>
<td>210, 229, 231</td>
</tr>
<tr>
<td>runEmptyDrops</td>
<td>105–109, 230</td>
</tr>
<tr>
<td>runEnrichR</td>
<td>39, 232</td>
</tr>
<tr>
<td>runFastMNN</td>
<td>233</td>
</tr>
<tr>
<td>runFeatureSelection</td>
<td>47, 176, 235, 245, 269</td>
</tr>
<tr>
<td>runFindMarker</td>
<td>40, 109, 111, 193, 236, 236</td>
</tr>
<tr>
<td>runGSVA</td>
<td>238</td>
</tr>
<tr>
<td>runHarmony</td>
<td>239</td>
</tr>
<tr>
<td>runKMeans</td>
<td>241</td>
</tr>
<tr>
<td>runLimmaBC</td>
<td>242</td>
</tr>
<tr>
<td>runLimmaDE</td>
<td>(runDEAnalysis), 220</td>
</tr>
<tr>
<td>runMAST</td>
<td>(runDEAnalysis), 220</td>
</tr>
<tr>
<td>runMNNCorrect</td>
<td>233, 243</td>
</tr>
<tr>
<td>runModelGeneVar</td>
<td>47, 176, 235, 244, 269</td>
</tr>
<tr>
<td>runNormalization</td>
<td>245</td>
</tr>
<tr>
<td>runPCA</td>
<td>299</td>
</tr>
<tr>
<td>runPerCellQC</td>
<td>116, 117, 214, 247</td>
</tr>
<tr>
<td>runQuickTSNE</td>
<td>(runTSNE), 288</td>
</tr>
<tr>
<td>runQuickUMAP</td>
<td>(runUMAP), 291</td>
</tr>
<tr>
<td>runSCANORAMA</td>
<td>249</td>
</tr>
<tr>
<td>runScanpyFindClusters</td>
<td>250</td>
</tr>
<tr>
<td>runScanpyFindHVG</td>
<td>252</td>
</tr>
<tr>
<td>runScanpyFindMarkers</td>
<td>253</td>
</tr>
<tr>
<td>runScanpyNormalizeData</td>
<td>254</td>
</tr>
<tr>
<td>runScanpyPCA</td>
<td>255</td>
</tr>
<tr>
<td>runScanpyScaleData</td>
<td>256</td>
</tr>
<tr>
<td>runScanpyTSNE</td>
<td>257</td>
</tr>
<tr>
<td>runScanpyUMAP</td>
<td>258</td>
</tr>
<tr>
<td>runScDbFinder</td>
<td>133, 135, 260</td>
</tr>
<tr>
<td>runSCMerge</td>
<td>261</td>
</tr>
<tr>
<td>runScratch</td>
<td>167, 264</td>
</tr>
<tr>
<td>runSeuratFindClusters</td>
<td>267</td>
</tr>
<tr>
<td>runSeuratFindHVG</td>
<td>47, 176, 235, 245, 268</td>
</tr>
<tr>
<td>runSeuratFindMarkers</td>
<td>269</td>
</tr>
<tr>
<td>runSeuratHeatmap</td>
<td>270</td>
</tr>
<tr>
<td>runSeuratICA</td>
<td>228, 272</td>
</tr>
<tr>
<td>runSeuratIntegration</td>
<td>273</td>
</tr>
<tr>
<td>runSeuratJackStraw</td>
<td>274</td>
</tr>
<tr>
<td>runSeuratNormalizeData</td>
<td>275</td>
</tr>
<tr>
<td>runSeuratPCA</td>
<td>228, 276</td>
</tr>
<tr>
<td>runSeuratScaleData</td>
<td>278</td>
</tr>
<tr>
<td>runSeuratSCTransform</td>
<td>279</td>
</tr>
<tr>
<td>runSeuratTSNE</td>
<td>228, 279</td>
</tr>
<tr>
<td>runSeuratUMAP</td>
<td>228, 280</td>
</tr>
<tr>
<td>runSingleR</td>
<td>282</td>
</tr>
<tr>
<td>runSoupX</td>
<td>45, 174, 283</td>
</tr>
<tr>
<td>runTSCAN</td>
<td>48, 178, 183, 285, 286</td>
</tr>
<tr>
<td>runTSCANClusterDEAnalysis</td>
<td>48, 177, 178, 286</td>
</tr>
<tr>
<td>runTSCANDEG</td>
<td>48, 180–182, 287</td>
</tr>
<tr>
<td>runTSNE</td>
<td>228, 288</td>
</tr>
<tr>
<td>RunUMAP</td>
<td>281</td>
</tr>
</tbody>
</table>
INDEX

runUMAP, 228, 291
runVAM, 293
runWilcox (runDEAnalysis), 220
runZINBWaVE, 295

sampleSummaryStats, 296
ScaleData, 272, 277
scaterCPM, 297
scaterlogNormCounts, 289, 292, 298
scaterPCA, 228, 299
scDBlFinder, 260, 261
sce, 300
sceBatches, 301
scRNAseq, 60
scSEGIndex, 261
sctkListGeneSetCollections, 294, 301
sctkPythonInstallConda, 302, 306
sctkPythonInstallVirtualEnv, 304, 307
SCTransform, 279
SEG, 305
segList, 261
segList_ensemblGeneID, 261
selectSCTKConda, 303, 305
selectSCTKVirtualEnvironment, 304, 306
setContaminationFraction, 284
setRowNames, 307
setSampleSummaryStatsTable<-
  (getSampleSummaryStatsTable), 42
setSampleSummaryStatsTable<-, SingleCellExperiment-method
  (getSampleSummaryStatsTable), 42
setSCTKDisplayRow, 36, 47, 97, 101, 176,
  177, 180–182, 308
setTopHVG, 235
setTopHVG (getTopHVG), 46
SingleCellExperiment, 9, 11, 15, 16, 19, 20,
  26–29, 31–36, 38, 40–46, 48, 50, 51,
  53, 60–74, 77, 80–83, 85, 86, 89–91,
  94, 96, 99–101, 104–108, 110, 113,
  115, 116, 133, 136, 138, 140,
  142–144, 146, 148, 150, 151, 153,
  156, 158, 161, 163, 166, 174,
  177–179, 181–185, 197–214,
  216–219, 222–232, 234–244, 247,
  249, 250, 260–266, 282–284, 286,
  287, 289, 290, 292–297, 299–301,
  307, 308, 311–313
singleCellTK, 63, 64, 66, 68, 69, 302,
  304–306, 308
subDiffEx, 309
subDiffExANOVA (subDiffEx), 309
subDiffExTest (subDiffEx), 309
subsetSCECols, 311
subsetSCERows, 312
SummarizedExperiment, 31, 32
summarizeSCE, 313
TENxPBMCData, 60, 61
thresholdSCRNACountMatrix, 112
trimCounts, 314
umap, 225
unit, 154, 155
virtualenv_create, 304
virtualenv_install, 304
with_seed, 226