

# Package ‘scfind’

October 16, 2019

**Type** Package

**Title** A search tool for single cell RNA-seq data by gene lists

**Version** 1.6.0

**Author** Vladimir Kiselev

**Maintainer** Vladimir Kiselev <vladimir.yu.kiselev@gmail.com>

**Description** Recently a very large collection of single-cell RNA-seq (scRNA-seq) datasets has been generated and publicly released. For the collection to be useful, the information must be organized in a way that supports queries that are relevant to researchers. `scfind` builds an index from scRNA-seq datasets which organizes the information in a suitable and compact manner so that the datasets can be very efficiently searched for either cells or cell types in which a given list of genes is expressed.

**License** GPL-3

**Imports** SingleCellExperiment, SummarizedExperiment, methods, stats, bit, dplyr, hash, reshape2, Rcpp(>= 0.12.12)

**LinkingTo** Rcpp

**Depends** R(>= 3.4)

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 6.0.1

**Suggests** knitr, rmarkdown, testthat

**VignetteBuilder** knitr

**biocViews** ImmunoOncology, SingleCell, Software, RNASeq, Transcriptomics, DataRepresentation, Transcription, Sequencing, GeneExpression

**NeedsCompilation** no

**URL** <https://github.com/hemberg-lab/scfind>

**BugReports** <https://support.bioconductor.org/t/scfind/>

**git\_url** <https://git.bioconductor.org/packages/scfind>

**git\_branch** RELEASE\_3\_9

**git\_last\_commit** 99f610b

**git\_last\_commit\_date** 2019-05-02

**Date/Publication** 2019-10-15

**R topics documented:**

ann . . . . .	2
buildCellIndex . . . . .	2
buildCellTypeIndex . . . . .	3
findCell . . . . .	4
findCellType . . . . .	5
yan . . . . .	6

<b>Index</b>	<b>7</b>
--------------	----------

---

ann	<i>Cell type annotations for data extracted from a publication by Yan et al.</i>
-----	--

---

**Description**

Cell type annotations for data extracted from a publication by Yan et al.

**Usage**

```
ann
```

**Format**

An object of class `data.frame` with 90 rows and 1 columns.

**Source**

<http://dx.doi.org/10.1038/nsmb.2660>

Each row corresponds to a single cell from 'yan' dataset

---

buildCellIndex	<i>Build a cell Index</i>
----------------	---------------------------

---

**Description**

Creates a compressed cell Index

**Usage**

```
buildCellIndex(object = NULL, cell_type_column = "cell_type1")
```

```
buildCellIndex.SCESet(object, cell_type_column)
```

```
## S4 method for signature 'SingleCellExperiment'
buildCellIndex(object = NULL,
  cell_type_column = "cell_type1")
```

**Arguments**

`object` object of `SingleCellExperiment` class containing the cell classification information

`cell_type_column` column name in the `colData` slot of the object `SingleCellExperiment` containing the cell classification information

**Value**

a 'data.frame' containing calculated gene index

**Examples**

```
library(SingleCellExperiment)
sce <- SingleCellExperiment(assays = list(normcounts = as.matrix(yan)), colData = ann)
# this is needed to calculate dropout rate for feature selection
# important: normcounts have the same zeros as raw counts (fpkm)
counts(sce) <- normcounts(sce)
logcounts(sce) <- log2(normcounts(sce) + 1)
# use gene names as feature symbols
rowData(sce)$feature_symbol <- rownames(sce)
isSpike(sce, 'ERCC') <- grepl('^ERCC-', rownames(sce))
# remove features with duplicated names
sce <- sce[!duplicated(rownames(sce)), ]
index <- buildCellIndex(sce)
```

---

`buildCellTypeIndex`     *Build a cell type Index*

---

**Description**

Calculates a fraction of expressed cells per gene per cell type

**Usage**

```
buildCellTypeIndex(object = NULL, cell_type_column = "cell_type1")

buildCellTypeIndex.SCESet(object, cell_type_column)

## S4 method for signature 'SingleCellExperiment'
buildCellTypeIndex(object = NULL,
  cell_type_column = "cell_type1")
```

**Arguments**

`object` object of `SingleCellExperiment` class

`cell_type_column` column name in the `colData` slot of the object `SingleCellExperiment` containing the cell classification information

**Value**

a 'data.frame' containing calculated gene index

**Examples**

```
library(SingleCellExperiment)
sce <- SingleCellExperiment(assays = list(normcounts = as.matrix(yan)), colData = ann)
# this is needed to calculate dropout rate for feature selection
# important: normcounts have the same zeros as raw counts (fpkm)
counts(sce) <- normcounts(sce)
logcounts(sce) <- log2(normcounts(sce) + 1)
# use gene names as feature symbols
rowData(sce)$feature_symbol <- rownames(sce)
isSpike(sce, 'ERCC') <- grepl('^ERCC-', rownames(sce))
# remove features with duplicated names
sce <- sce[!duplicated(rownames(sce)), ]
index <- buildCellTypeIndex(sce)
```

---

findCell

*Find cells associated with a given gene list*

---

**Description**

Calculates p-values of a log-likelihood of a list of genes to be associated with each cell type. Log-likelihood is based on gene expression values.

**Usage**

```
findCell(input = NULL, genelist = NULL)
```

```
findCell.SCESet(input, genelist)
```

```
## S4 method for signature 'list'
```

```
findCell(input = NULL, genelist = NULL)
```

**Arguments**

**input** object of SingleCellExperiment class

**genelist** column name in the colData slot of the object SingleCellExperiment containing the cell classification information

**Value**

a 'list' containing calculated gene index

**Examples**

```

library(SingleCellExperiment)
sce <- SingleCellExperiment(assays = list(normcounts = as.matrix(yan)), colData = ann)
# this is needed to calculate dropout rate for feature selection
# important: normcounts have the same zeros as raw counts (fpkm)
counts(sce) <- normcounts(sce)
logcounts(sce) <- log2(normcounts(sce) + 1)
# use gene names as feature symbols
rowData(sce)$feature_symbol <- rownames(sce)
isSpike(sce, 'ERCC') <- grepl('^ERCC-', rownames(sce))
# remove features with duplicated names
sce <- sce[!duplicated(rownames(sce)), ]
index <- buildCellIndex(sce)
res <- findCell(index, genelist = c('SOX6', 'SNAI3'))

```

---

findCellType

*Find cell types associated with a given gene list*


---

**Description**

Calculates p-values of a log-likelihood of a list of genes to be associated with each cell type. Log-likelihood is based on gene expression values.

**Usage**

```

findCellType(gene_index = NULL, gene_list = NULL)

findCellType.data.frame(gene_index, gene_list)

## S4 method for signature 'data.frame'
findCellType(gene_index = NULL, gene_list = NULL)

```

**Arguments**

gene_index	a data.frame with cell types in columns and genes in rows
gene_list	genes that need to be searched in the gene_index

**Value**

a named numeric vector containing p-values

**Examples**

```

library(SingleCellExperiment)
sce <- SingleCellExperiment(assays = list(normcounts = as.matrix(yan)), colData = ann)
# this is needed to calculate dropout rate for feature selection
# important: normcounts have the same zeros as raw counts (fpkm)
counts(sce) <- normcounts(sce)
logcounts(sce) <- log2(normcounts(sce) + 1)
# use gene names as feature symbols
rowData(sce)$feature_symbol <- rownames(sce)
isSpike(sce, 'ERCC') <- grepl('^ERCC-', rownames(sce))

```

```
# remove features with duplicated names
sce <- sce[!duplicated(rownames(sce)), ]
index <- buildCellTypeIndex(sce)
res <- findCellType(index, gene_list = c('SOX6', 'SNAI3'))
```

---

yan

*Single cell RNA-Seq data extracted from a publication by Yan et al.*

---

### **Description**

Single cell RNA-Seq data extracted from a publication by Yan et al.

### **Usage**

yan

### **Format**

An object of class `data.frame` with 20214 rows and 90 columns.

### **Source**

<http://dx.doi.org/10.1038/nsmb.2660>

Columns represent cells, rows represent genes expression values.

# Index

## \*Topic **datasets**

ann, [2](#)

yan, [6](#)

ann, [2](#)

buildCellIndex, [2](#)

buildCellIndex, SingleCellExperiment-method  
(buildCellIndex), [2](#)

buildCellIndex.SCESet (buildCellIndex),  
[2](#)

buildCellTypeIndex, [3](#)

buildCellTypeIndex, SingleCellExperiment-method  
(buildCellTypeIndex), [3](#)

buildCellTypeIndex.SCESet

(buildCellTypeIndex), [3](#)

findCell, [4](#)

findCell, list-method (findCell), [4](#)

findCell.SCESet (findCell), [4](#)

findCellType, [5](#)

findCellType, data.frame-method  
(findCellType), [5](#)

findCellType.data.frame (findCellType),  
[5](#)

yan, [6](#)