

# Package ‘gscreend’

November 21, 2024

**Type** Package

**Title** Analysis of pooled genetic screens

**Version** 1.21.0

**Description** Package for the analysis of pooled genetic screens (e.g. CRISPR-KO). The analysis of such screens is based on the comparison of gRNA abundances before and after a cell proliferation phase. The gscreend packages takes gRNA counts as input and allows detection of genes whose knockout decreases or increases cell proliferation.

**License** GPL-3

**Encoding** UTF-8

**LazyData** false

**Depends** R (>= 3.6)

**Imports** SummarizedExperiment, nloptr, fGarch, methods, BiocParallel, graphics

**Suggests** knitr, testthat, rmarkdown, BiocStyle

**VignetteBuilder** knitr

**RoxygenNote** 7.2.3

**biocViews** Software, StatisticalMethod, PooledScreens, CRISPR

**URL** <https://github.com/imkeller/gspreend>

**BugReports** <https://github.com/imkeller/gspreend/issues>

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

**git\_branch** devel

**git\_last\_commit** 12b0796

**git\_last\_commit\_date** 2024-10-29

**Repository** Bioconductor 3.21

**Date/Publication** 2024-11-20

**Author** Katharina Imkeller [cre, aut],  
Wolfgang Huber [aut]

**Maintainer** Katharina Imkeller <k.imkeller@dkfz.de>

## Contents

assignGeneData . . . . .	2
calculateIntervalFits . . . . .	3
calculateLFC . . . . .	3
calculatePValues . . . . .	4
createPoolScreenExp . . . . .	4
createPoolScreenExpFromSE . . . . .	5
defineFittingIntervals . . . . .	6
fit_least_quantile . . . . .	6
GeneData . . . . .	7
GeneData,PoolScreenExp-method . . . . .	7
normalizePoolScreenExp . . . . .	8
plotModelParameters . . . . .	8
plotReplicateCorrelation . . . . .	9
PoolScreenExp-class . . . . .	10
ResultsTable . . . . .	10
RunGscreen . . . . .	11
sgRNAData . . . . .	12
sgRNAData,PoolScreenExp-method . . . . .	13
<b>Index</b>	<b>14</b>

---

assignGeneData	<i>Calculate gene rank</i>
----------------	----------------------------

---

### Description

Calculate gene rank

### Usage

```
assignGeneData(object, alpha_cutoff)
```

### Arguments

object	PoolScreenExp object
alpha_cutoff	alpha cutoff for alpha-RRA (default: 0.05)

### Value

object

---

calculateIntervalFits *Calculate fit parameters for every subset of data*

---

**Description**

Calculate fit parameters for every subset of data

**Usage**

```
calculateIntervalFits(object, quant1, quant2)
```

**Arguments**

object	PoolScreenExp object
quant1	lower quantile for least squares regression (default: 0.1)
quant2	upper quantile for least squares regression (default: 0.9)

**Value**

object

---

calculateLFC *Calculate log fold changes*

---

**Description**

Calculate log fold changes

**Usage**

```
calculateLFC(object)
```

**Arguments**

object	PoolScreenExp object
--------	----------------------

**Value**

object

---

calculatePValues      *Calculate p-values*

---

**Description**

Calculate p-values

**Usage**

```
calculatePValues(object)
```

**Arguments**

object      PoolScreenExp object

**Value**

object

---

createPoolScreenExp      *Create PoolScreenExp Experiment*

---

**Description**

Create PoolScreenExp Experiment

**Usage**

```
createPoolScreenExp(data)
```

**Arguments**

data      Input data object containing gRNA level data (SummarizedExperiment)

**Value**

object PoolScreenExp object

**Examples**

```
raw_counts <- read.table(  
  system.file('extdata', 'simulated_counts.txt',  
  package = 'gscreend'),  
  header=TRUE)  
  
counts_matrix <- cbind(raw_counts$library0, raw_counts$R0_0, raw_counts$R1_0)  
  
rowData <- data.frame(sgRNA_id = raw_counts$sgrna_id,  
  gene = raw_counts$Gene)  
  
colData <- data.frame(samplename = c('library', 'R1', 'R2'),  
  timepoint = c('T0', 'T1', 'T1'))  
  
library(SummarizedExperiment)  
se <- SummarizedExperiment(assays=list(counts=counts_matrix),  
  rowData=rowData, colData=colData)  
  
# create a PoolScreenExp experiment  
pse <- createPoolScreenExp(se)
```

---

```
createPoolScreenExpFromSE
```

*Create PoolScreenExp Experiment from summarized experiment*

---

**Description**

Create PoolScreenExp Experiment from summarized experiment

**Usage**

```
createPoolScreenExpFromSE(data)
```

**Arguments**

data                    SummarizedExperiment object containing gRNA level data

**Value**

object

---

`defineFittingIntervals`*Calculate interval limits for splitting data into subsets*

---

**Description**

Calculate interval limits for splitting data into subsets

**Usage**

```
defineFittingIntervals(object)
```

**Arguments**

object            PoolScreenExp object

**Value**

object

---

`fit_least_quantile`    *Fit paramters for skew normal distribution*

---

**Description**

Fit paramters for skew normal distribution

**Usage**

```
fit_least_quantile(LFC, quant1, quant2)
```

**Arguments**

LFC                logarithmic fold changes of gRNA counts  
quant1             lower quantile for least quantile of squares regression (default: 0.1)  
quant2             upper quantile for least quantile of squares regression (default: 0.9)

**Value**

fit\_skewnorm

---

GeneData

*GeneData: set and retrieve GeneData of PoolScreenExp*

---

### Description

GeneData: set and retrieve GeneData of PoolScreenExp

### Usage

```
GeneData(x)
```

### Arguments

x                    PoolScreenExp object

### Value

Gene slot of the object

### Examples

```
# import a PoolScreenExp object that has been generated using
# RunGscreen()
pse_an <- readRDS(
  system.file('extdata', 'gscreen_analysed_experiment.RData',
  package = 'gscreen'))

GeneData(pse_an)
```

---

GeneData, PoolScreenExp-method

*Accessor function for the Gene slot of the PoolScreenExp class*

---

### Description

Accessor function for the Gene slot of the PoolScreenExp class

### Usage

```
## S4 method for signature 'PoolScreenExp'
GeneData(x)
```

### Arguments

x                    PoolScreenExp object

**Value**

Gene slot of the object

**Examples**

```
# import a PoolScreenExp object that has been generated using
# RunGscreen()
pse_an <- readRDS(
  system.file('extdata', 'gscreen_analysed_experiment.RData',
    package = 'gscreen'))
```

```
GeneData(pse_an)
```

---

normalizePoolScreenExp

*Normalize raw count*

---

**Description**

Normalize raw count

**Usage**

```
normalizePoolScreenExp(object)
```

**Arguments**

object            PoolScreenExp object

**Value**

object

---

plotModelParameters

*Plot model parameters from the fitting*

---

**Description**

Plot model parameters from the fitting

**Usage**

```
plotModelParameters(object)
```



**Arguments**

object            PoolScreenExp object

**Value**

plot

**Examples**

```
# import a PoolScreenExp object that has been generated using
# RunGscreen()
pse_an <- readRDS(
  system.file('extdata', 'gscreen_analysed_experiment.RData',
  package = 'gscreen'))
plotModelParameters(pse_an)
```

---

plotReplicateCorrelation

*Plot replicate correlation*

---

**Description**

Plot replicate correlation

**Usage**

```
plotReplicateCorrelation(object, rep1 = "R1", rep2 = "R2")
```

**Arguments**

object            PoolScreenExp object  
rep1              Name of replicate 1  
rep2              Name of replicate 2

**Value**

replicate\_plot

**Examples**

```
# import a PoolScreenExp object that has been generated using RunGscreen()
pse_an <- readRDS(
  system.file('extdata', 'gscreen_analysed_experiment.RData',
  package = 'gscreen'))
plotReplicateCorrelation(pse_an, rep1 = 'R1', rep2 = 'R2')
```

---

PoolScreenExp-class     *Class to store pooled CRISPR screening experiment*

---

### Description

The poolScreenExp class is an S4 class used to store sgRNA and gene related data as well as parameters necessary for statistical model.

### Slots

sgRNAData A SummarizedExperiment containing the data related to sgRNAs.

FittingIntervals A vector defining the limits of the intervals used for fitting of null model.

LFCModelParameters A vector of parameters estimated when fitting the null model.

GeneData SummarizedExperiment containing the data related to genes.

FittingOptions A named list with options for fitting: IntervalFraction - fraction of sgRNAs used in every fitting interval (default 0.1), alphaCutoff - alpha cutoff for alpha RRA algorithm (default: 0.05).

---

ResultsTable     *Extract a results table*

---

### Description

Extract a results table

### Usage

```
ResultsTable(object, direction = "negative")
```

### Arguments

object     PoolScreenExp object

direction     Whether the table should contain information on positive or negative fold changes  
['negative'|'positive']

### Value

plot

**Examples**

```
# import a PoolScreenExp object that has been generated using
# RunGscreen()
pse_an <- readRDS(
  system.file('extdata', 'gscreen_analysed_experiment.RData',
    package = 'gscreen'))
ResultsTable(pse_an, direction = 'negative')
```

---

RunGscreen	<i>run gscreen</i>
------------	--------------------

---

**Description**

run gscreen

**Usage**

```
RunGscreen(object, quant1 = 0.1, quant2 = 0.9, alphacutoff = 0.05)
```

**Arguments**

object	PoolScreenExp object
quant1	lower quantile for least quantile of squares regression (default: 0.1)
quant2	upper quantile for least quantile of squares regression (default: 0.9)
alphacutoff	alpha cutoff for alpha-RRA (default: 0.05)

**Value**

object

**Examples**

```
raw_counts <- read.table(
  system.file('extdata', 'simulated_counts.txt',
    package = 'gscreen'),
  header=TRUE)

# Create the PoolScreenExp to be analyzed
counts_matrix <- cbind(raw_counts$library0, raw_counts$R0_0, raw_counts$R1_0)

rowData <- data.frame(sgRNA_id = raw_counts$sgrna_id,
  gene = raw_counts$Gene)

colData <- data.frame(samplename = c('library', 'R1', 'R2'),
  timepoint = c('T0', 'T1', 'T1'))

library(SummarizedExperiment)
```

```
se <- SummarizedExperiment(assays=list(counts=counts_matrix),
rowData=rowData, colData=colData)

pse <- createPoolScreenExp(se)

# Run Analysis
pse_an <- RunGscreen(pse)
```

---

sgRNAData

*sgRNAData: set and retrieve sgRNAData of PoolScreenExp*

---

## Description

sgRNAData: set and retrieve sgRNAData of PoolScreenExp

## Usage

```
sgRNAData(x)
```

## Arguments

x                    PoolScreenExp object

## Value

sgRNA slot of the object

## Examples

```
# import a PoolScreenExp object that has been generated using
# RunGscreen()
pse_an <- readRDS(
system.file('extdata', 'gscreen_analysed_experiment.RData',
package = 'gscreen'))

sgRNAData(pse_an)
```

---

`sgRNAData,PoolScreenExp-method`*Accessor function for the sgRNA slot of the PoolScreenExp class*

---

**Description**

Accessor function for the sgRNA slot of the PoolScreenExp class

**Usage**

```
## S4 method for signature 'PoolScreenExp'  
sgRNAData(x)
```

**Arguments**

x                    PoolScreenExp object

**Value**

sgRNA slot of the object

**Examples**

```
# import a PoolScreenExp object that has been generated using  
# RunGscreenend()  
pse_an <- readRDS(  
system.file('extdata', 'gscreenend_analysed_experiment.RData',  
package = 'gscreenend'))  
  
sgRNAData(pse_an)
```

# Index

## \* **internal**

- assignGeneData, [2](#)
- calculateIntervalFits, [3](#)
- calculateLFC, [3](#)
- calculatePValues, [4](#)
- createPoolScreenExpFromSE, [5](#)
- defineFittingIntervals, [6](#)
- fit\_least\_quantile, [6](#)
- normalizePoolScreenExp, [8](#)

assignGeneData, [2](#)

calculateIntervalFits, [3](#)  
calculateLFC, [3](#)  
calculatePValues, [4](#)  
createPoolScreenExp, [4](#)  
createPoolScreenExpFromSE, [5](#)

defineFittingIntervals, [6](#)

fit\_least\_quantile, [6](#)

GeneData, [7](#)  
GeneData, PoolScreenExp-method, [7](#)

normalizePoolScreenExp, [8](#)

plotModelParameters, [8](#)  
plotReplicateCorrelation, [9](#)  
PoolScreenExp-class, [10](#)

ResultsTable, [10](#)  
RunGscreend, [11](#)

sgRNAData, [12](#)  
sgRNAData, PoolScreenExp-method, [13](#)