

# Package ‘chromVAR’

November 26, 2023

**Type** Package

**Title** Chromatin Variation Across Regions

**Version** 1.24.0

**Description** Determine variation in chromatin accessibility across sets of annotations or peaks. Designed primarily for single-cell or sparse chromatin accessibility data, e.g. from scATAC-seq or sparse bulk ATAC or DNase-seq experiments.

**License** MIT + file LICENSE

**Imports** IRanges, GenomeInfoDb, GenomicRanges, ggplot2, nabor, BiocParallel, BiocGenerics, Biostrings, TFBSTools, Rsamtools, S4Vectors, methods, Rcpp, grid, plotly, shiny, miniUI, stats, utils, graphics, DT, Rtsne, Matrix, SummarizedExperiment, RColorBrewer, BSgenome

**Depends** R (>= 3.4)

**Suggests** JASPAR2016, BSgenome.Hsapiens.UCSC.hg19, readr, testthat, knitr, rmarkdown, pheatmap, motifmatchr

**biocViews** SingleCell, Sequencing, GeneRegulation, ImmunoOncology

**LazyData** TRUE

**LinkingTo** Rcpp, RcppArmadillo

**SystemRequirements** C++11

**VignetteBuilder** knitr

**RoxygenNote** 6.0.1

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

**git\_branch** RELEASE\_3\_18

**git\_last\_commit** e2d61b0

**git\_last\_commit\_date** 2023-10-24

**Repository** Bioconductor 3.18

**Date/Publication** 2023-11-26

**Author** Alicia Schep [aut, cre],  
 Jason Buenrostro [ctb],  
 Caleb Lareau [ctb],  
 William Greenleaf [ths],  
 Stanford University [cph]

**Maintainer** Alicia Schep <aschep@gmail.com>

## R topics documented:

addGCBias	3
annotationMatches	4
assembleKmers	5
cbind,chromVARDeviations-method	6
chromVAR	6
chromVARDeviations-class	7
chromVAR_theme	7
computeDeviations	8
computeExpectations	10
computeVariability	11
counts,SummarizedExperiment-method	12
deviations	13
deviationScores	13
deviationsCovariability	14
deviationsTsne	15
differentialDeviations	16
differentialVariability	17
example_counts	17
filterPeaks	18
filterSamples	19
filterSamplesPlot	20
getAnnotationCorrelation	21
getAnnotations	22
getAnnotationSynergy	24
getBackgroundPeaks	26
getCisGroups	27
getCounts	28
getFragmentsPerPeak	29
getFragmentsPerSample	30
getJasparMotifs	31
getPeaks	32
getPermutedData	33
getSampleCorrelation	34
getSampleDepths	35
getSampleDistance	36
getTotalFragments	37
makeBiasBins	38
makePermutedSets	39

*addGCBias* 3

matchKmers . . . . .	41
mini_counts . . . . .	43
mini_dev . . . . .	43
mini_ix . . . . .	44
plotDeviationsTsne . . . . .	44
plotKmerMismatch . . . . .	45
plotVariability . . . . .	46
pwmDistance . . . . .	46
rbind,chromVARDeviations-method . . . . .	47
readNarrowpeaks . . . . .	48

**Index** 49

---

addGCBias	<i>addGCBias</i>
-----------	------------------

---

## Description

Computes GC content for peaks

## Usage

```
addGCBias(object, ...)
```

```
## S4 method for signature 'RangedSummarizedExperiment'
```

```
addGCBias(object,  
  genome = GenomeInfoDb::genome(object))
```

```
## S4 method for signature 'SummarizedExperiment'
```

```
addGCBias(object, peaks,  
  genome = GenomeInfoDb::genome(peaks))
```

## Arguments

object	(Ranged)SummarizedExperiment
...	additional arguments
genome	BSgenome object, by default hg19
peaks	GenomicRanges with peaks, needed if object is SummarizedExperiment and not RangedSummarizedExperiment

## Value

(Ranged)SummarizedExperiment object with new column in row metadata with the gc content of the peak in question

**Methods (by class)**

- RangedSummarizedExperiment: method for RangedSummarizedExperiment
- SummarizedExperiment: method for SummarizedExperiment

**Examples**

```
data(example_counts, package = "chromVAR")
# show example on small part of data
subset_counts <- example_counts[1:500,]
library(BSgenome.Hsapiens.UCSC.hg19)
example_counts <- addGCbias(subset_counts,
                             genome = BSgenome.Hsapiens.UCSC.hg19)
```

---

annotationMatches      *annotationMatches*

---

**Description**

annotationMatches

**Usage**

```
annotationMatches(object)

annotationMatches(object) <- value

## S4 method for signature 'SummarizedExperiment'
annotationMatches(object)

## S4 replacement method for signature 'SummarizedExperiment'
annotationMatches(object) <- value
```

**Arguments**

object            SummarizedExperiment with matches slot, see details  
value             logical Matrix with annotation matches

**Details**

Will extract matrix from the "matches", "annotationMatches", or "motif\_matches" assay of a SummarizedExperiment

**Value**

logical matrix of annotation matches

**Author(s)**

Alicia Schep

**Examples**

```
# load annotation matrix; result from matchMotifs
data(mini_ix, package = "chromVAR")
matches <- annotationMatches(mini_ix)
```

---

assembleKmers	<i>assembleKmers</i>
---------------	----------------------

---

**Description**

function to create de novo motifs from kmers based on deviations

**Usage**

```
assembleKmers(object, threshold = 1.5, p = 0.01, progress = TRUE)
```

**Arguments**

object	kmer chromVARDeviations object
threshold	variability threshold
p	p value threshold for inclusion of kmer
progress	show progress bar?

**Details**

function for assembling de novo kmers from kmer deviations

**Value**

list with (1) motifs: de novo motif matrices, (2) seed: seed kmer for de novo motif

---

`cbind, chromVARDeviations-method`  
*cbind method for chromVARDeviations*

---

### Description

`cbind` returns an error when applied to `chromVARDeviations` because results for all cells or samples should originate from same `computeDeviations` computation

### Usage

```
## S4 method for signature 'chromVARDeviations'  
cbind(..., deparse.level = 1)
```

### Arguments

`...` `chromVARDeviations` object to be combined  
`deparse.level` See `?base::rbind` for a description of this argument.

### Value

`chromVARDeviations` object

### Author(s)

Alicia Schep

### See Also

[chromVARDeviations-class](#)

---

`chromVAR` *chromVAR: A package for computing variability across sets of peaks.*

---

### Description

Determine variation in chromatin accessibility across sets of annotations or peaks. Designed primarily for single-cell or sparse chromatin accessibility, e.g. from scATAC-seq or sparse ATAC or DNase-seq experiments.

---

chromVARDeviations-class  
*chromVARDeviations*

---

### Description

Class for storing results from `computeDeviations` function.

### Details

This class inherits from `SummarizedExperiment`, and most methods for that class should work for objects of this class as well. Additionally, two accessor functions are defined for extracting bias corrected deviations (`deviations`) and deviation Z-scores (`deviationScores`)

---

chromVAR\_theme      *chromVAR\_theme*

---

### Description

theme for use with `ggplot2`, used by chromVAR plotting functions

### Usage

```
chromVAR_theme(base_size = 12, base_family = "Helvetica")
```

### Arguments

<code>base_size</code>	base font size
<code>base_family</code>	base font family

### Value

`ggplot2` theme

### Author(s)

Alicia Schep

### Examples

```
p <- ggplot2::qplot(1:3,1:3) + chromVAR_theme(18)
```

---

computeDeviations      *computeDeviations*

---

## Description

Computes deviations in chromatin accessibility across sets of annotations

## Usage

```
computeDeviations(object, annotations, ...)  
  
## S4 method for signature 'SummarizedExperiment,SummarizedExperiment'  
computeDeviations(object,  
  annotations, background_peaks = getBackgroundPeaks(object),  
  expectation = computeExpectations(object))  
  
## S4 method for signature 'SummarizedExperiment,MatrixOrmatrix'  
computeDeviations(object,  
  annotations, background_peaks = getBackgroundPeaks(object),  
  expectation = computeExpectations(object))  
  
## S4 method for signature 'SummarizedExperiment,list'  
computeDeviations(object, annotations,  
  background_peaks = getBackgroundPeaks(object),  
  expectation = computeExpectations(object))  
  
## S4 method for signature 'SummarizedExperiment,missingOrNULL'  
computeDeviations(object,  
  annotations, background_peaks = getBackgroundPeaks(object),  
  expectation = computeExpectations(object))  
  
## S4 method for signature 'MatrixOrmatrix,SummarizedExperiment'  
computeDeviations(object,  
  annotations, background_peaks, expectation = computeExpectations(object))  
  
## S4 method for signature 'MatrixOrmatrix,MatrixOrmatrix'  
computeDeviations(object, annotations,  
  background_peaks, expectation = computeExpectations(object))  
  
## S4 method for signature 'MatrixOrmatrix,list'  
computeDeviations(object, annotations,  
  background_peaks, expectation = computeExpectations(object))  
  
## S4 method for signature 'MatrixOrmatrix,missingOrNULL'  
computeDeviations(object, annotations,  
  background_peaks, expectation = computeExpectations(object))
```



**Arguments**

object	chromVARCounts object
annotations	chromVARAnnotations object
...	additional arguments
background_peaks	(optional) background peaks matrix computed using <a href="#">getBackgroundPeaks</a> , computed internally with default parameters if not provided
expectation	(optional) expectations computed using <a href="#">computeExpectations</a> , computed automatically if not provided

**Details**

multiprocessing using [bplapply](#)

**Value**

[chromVARDeviations-class](#), which inherits from `SummarizedExperiment`, and has two assays: `deviations` and `deviation scores`.

**Methods (by class)**

- object = `SummarizedExperiment`, annotations = `SummarizedExperiment`: object and annotations are `SummarizedExperiment`
- object = `SummarizedExperiment`, annotations = `MatrixOrmatrix`: object is `SummarizedExperiment`, annotations are `Matrix`
- object = `SummarizedExperiment`, annotations = `list`: object is `SummarizedExperiment`, annotations are `list`
- object = `SummarizedExperiment`, annotations = `missingOrNULL`: object is `SummarizedExperiment`, annotations are `missing`
- object = `MatrixOrmatrix`, annotations = `SummarizedExperiment`: object and annotations are `SummarizedExperiment`
- object = `MatrixOrmatrix`, annotations = `MatrixOrmatrix`: object is `SummarizedExperiment`, annotations are `Matrix`
- object = `MatrixOrmatrix`, annotations = `list`: object is `SummarizedExperiment`, annotations are `list`
- object = `MatrixOrmatrix`, annotations = `missingOrNULL`: object is `SummarizedExperiment`, annotations are `missing`

**Author(s)**

Alicia Schep

**See Also**

[computeVariability](#), [plotVariability](#)

**Examples**

```
# Register BiocParallel
BiocParallel::register(BiocParallel::SerialParam())
# Load very small example counts (already filtered)
data(mini_counts, package = "chromVAR")
# load annotation matrix; result from matchMotifs
data(mini_ix, package = "chromVAR")

# computing deviations
dev <- computeDeviations(object = mini_counts,
                          annotations = mini_ix)
```

---

```
computeExpectations  computeExpectations
```

---

**Description**

```
computeExpectations
```

**Usage**

```
computeExpectations(object, ...)

## S4 method for signature 'MatrixOrmatrix'
computeExpectations(object, norm = FALSE,
                    group = NULL)

## S4 method for signature 'SummarizedExperiment'
computeExpectations(object, norm = FALSE,
                    group = NULL)
```

**Arguments**

object	SummarizedExperiment
...	additional arguments
norm	weight all samples equally?
group	an group vector, optional

**Details**

By default, this function will compute the expected fraction of reads per peak as the the total fragments per peak across all samples divided by total reads in peaks in all samples. Optionally, norm can be set to TRUE and then the expectation will be the average fraction of reads in a peak across the cells. This is not recommended for single cell applications as cells with very few reads will have a large impact. Another option is to give a vector of groups, in which case the expectation will be the average fraction of reads per peak within each group. If group vector is provided and norm is set to TRUE then within each group the fraction of reads per peak is the average fraction of reads per

peak in each sample. Otherwise, the within group fraction of reads per peak is based on the reads per peak within the sample divided by the total reads within each sample. The group can also be given by a length 1 character vector representing the name of a column in the colData of the input object if the input is a SummarizedExperiment

**Value**

vector with expected fraction of reads per peak.

**Methods (by class)**

- MatrixOrmatrix: method for Matrix or matrix
- SummarizedExperiment: method for SummarizedExperiment with counts slot

**Author(s)**

Alicia Schep

**Examples**

```
# First get some data
data(mini_counts, package = "chromVAR")

# Compute expectations
expectations <- computeExpectations(mini_counts)
```

---

computeVariability      *computeVariability*

---

**Description**

function to compute overall variability of motif sets across samples

**Usage**

```
computeVariability(object, bootstrap_error = TRUE, bootstrap_samples = 1000,
  bootstrap_quantiles = c(0.025, 0.975), na.rm = TRUE)
```

**Arguments**

object	output from <a href="#">computeDeviations</a>
bootstrap_error	compute bootstrap confidence interval
bootstrap_samples	number of bootstrap samples to take
bootstrap_quantiles	quantiles for bootstrap
na.rm	remove NAs? default is true

**Value**

data.frame with columns for name, variability, bootstrap lower bound, bootstrap upper bound, raw p value, adjust p value.

**Examples**

```
# Load very small example results from computeDeviations
data(mini_dev, package = "chromVAR")
variability <- computeVariability(mini_dev)
```

---

counts, SummarizedExperiment-method

*Accessors for the 'counts' slot of a SummarizedExperiment*

---

**Description**

Accessors for the 'counts' slot of a SummarizedExperiment

**Usage**

```
## S4 method for signature 'SummarizedExperiment'
counts(object)

## S4 replacement method for signature 'SummarizedExperiment, MatrixOrmatrix'
counts(object) <- value
```

**Arguments**

object	SummarizedExperiment object
value	matrix of counts

**Value**

Matrix of counts

**Examples**

```
data(mini_counts, package = "chromVAR")
fragment_counts <- counts(mini_counts)
```

---

deviations	<i>deviations</i>
------------	-------------------

---

**Description**

Accessor for bias corrected deviations from [chromVARDeviations-class](#) object

**Usage**

```
deviations(object)  
  
## S4 method for signature 'chromVARDeviations'  
deviations(object)
```

**Arguments**

object            chromVARDeviations object

**Value**

matrix of bias corrected deviations

**Author(s)**

Alicia Schep

**Examples**

```
# Load very small example results from computeDeviations  
data(mini_dev, package = "chromVAR")  
bias_corrected_deviations <- deviations(mini_dev)
```

---

deviationScores	<i>deviationScores</i>
-----------------	------------------------

---

**Description**

Accessor for deviation Z-scores from [chromVARDeviations-class](#) object

**Usage**

```
deviationScores(object)  
  
## S4 method for signature 'chromVARDeviations'  
deviationScores(object)
```

**Arguments**

object            chromVARDeviations object

**Value**

The deviationScores and deviations accessors both return matrices.  
matrix of deviation Z-scores

**Author(s)**

Alicia Schep

**Examples**

```
# Load very small example results from computeDeviations
data(mini_dev, package = "chromVAR")
scores <- deviationScores(mini_dev)
```

---

deviationsCovariability  
*deviationsCovariability*

---

**Description**

deviationsCovariability

**Usage**

```
deviationsCovariability(object)
```

**Arguments**

object            deviations result

**Details**

Returns the 'covariability' between motifs/kmers/peaksets. Covariability' is defined as covariance between Z-scores divided by variance of Z-scores for one motif/kmer/peakset (the row).

**Value**

'covariability' matrix

**Examples**

```
# load very small example data
data(mini_counts, package = "chromVAR")
motifs <- getJasparMotifs()
library(motifmatchr)
motif_ix <- matchMotifs(motifs, mini_counts,
  genome = BSgenome.Hsapiens.UCSC.hg19::BSgenome.Hsapiens.UCSC.hg19)

# computing deviations
dev <- computeDeviations(object = mini_counts,
  annotations = motif_ix)

# get covariability for just first three motifs
devcov <- deviationsCovariability(dev[1:3,])
```

---

deviationsTsne

*deviationsTsne*


---

**Description**

Perform tsne using bias corrected deviations to visualize either cell/sample similarity or motif/kmer/annotation similarity

**Usage**

```
deviationsTsne(object, threshold = 1.5, perplexity = if (what == "samples")
  30 else 8, max_iter = 1000, theta = 0.5, what = c("samples",
  "annotations"), shiny = FALSE)
```

**Arguments**

object	deviations result
threshold	variability threshold – use only deviations with variability greater than threshold
perplexity	perplexity parameter for tsne
max_iter	max iterations parameter for tsne
theta	theta parameter for tsne
what	tsne for similarity of samples or annotations?
shiny	load a shiny widget that enable you to explore perplexity and variability threshold parameter?

**Value**

data.frame with two columns for the two dimensions of tSNE output

**Author(s)**

Alicia Schep

**Examples**

```
# Load very small example results from computeDeviations
data(mini_dev, package = "chromVAR")

tsne_res <- deviationsTsne(mini_dev, threshold = 0.8, shiny = FALSE)
# setting very low variability threshold because this is mini data set
# threshold should generally be above 1
# Use plotVariability to get a sense of an appropriate threshold
```

---

differentialDeviations

*differentialDeviations*


---

**Description**

Function to see whether deviations differ between groups

**Usage**

```
differentialDeviations(object, groups, alternative = c("two.sided", "less",
"greater"), parametric = TRUE)
```

**Arguments**

object	chromVARDeviations object
groups	either vector of groups or name of column in colData of object with group information
alternative	only used if there are two groups – two.sided or one sided test
parametric	use parametric test. alternatively will use kruskal wallace

**Value**

data.frame with p value and adjusted p value

**Author(s)**

Alicia Schep

**Examples**

```
# Load very small example results from computeDeviations
data(mini_dev, package = "chromVAR")
difdev <- differentialDeviations(mini_dev, "Cell_Type")
```



---

```
differentialVariability
      differentialVariability
```

---

**Description**

Function to determine whether groups differ in variability

**Usage**

```
differentialVariability(object, groups, parametric = TRUE)
```

**Arguments**

object	chromVARDeviations object
groups	either vector of groups or name of column in colData of object with group information
parametric	use parametric test. alternatively will use kruskal wallace

**Value**

data.frame with p value and adjusted p value

**Author(s)**

Alicia Schep

**Examples**

```
# Load very small example results from computeDeviations
data(mini_dev, package = "chromVAR")
difvar <- differentialVariability(mini_dev, "Cell_Type")
```

---

```
example_counts      example_counts
```

---

**Description**

Very small sample data set for trying out chromVAR

**Usage**

```
data(example_counts)
```

**Value**

[RangedSummarizedExperiment](#)

**Examples**

```
data(example_counts)
```

---

filterPeaks

*filterPeaks*

---

**Description**

function to get indices of peaks that pass filters

**Usage**

```
filterPeaks(object, min_fragments_per_peak = 1, non_overlapping = TRUE,  
            ix_return = FALSE)
```

**Arguments**

object	SummarizedExperiment with matrix of fragment counts per peak per sample, as computed by <a href="#">getCounts</a>
min_fragments_per_peak	minimum number of fragments in peaks across all samples
non_overlapping	reduce peak set to non-overlapping peaks, see details
ix_return	return indices of peaks to keep instead of subsetted counts object

**Details**

if `non_overlapping` is set to true, when peaks overlap the overlapping peak with lower counts is removed

**Value**

vector of indices, representing peaks that should be kept

**Author(s)**

Alicia Schep

**See Also**

[getPeaks](#), [filterSamples](#), [getCounts](#)



---

filterSamplesPlot     *filterSamplesPlot*

---

## Description

plot filtering of samples

## Usage

```
filterSamplesPlot(object, min_in_peaks = NULL, min_depth = NULL,  
  use_plotly = interactive())
```

## Arguments

object	SummarizedExperiment with matrix of fragment counts per peak per sample, as computed by <a href="#">getCounts</a>
min_in_peaks	minimum fraction of samples within peaks
min_depth	minimum library size
use_plotly	make interactive plot?

## Details

If unspecified, `min_in_peaks` and `min_depth` cutoffs will be estimated based on data. `min_in_peaks` is set to 0.5 times the median proportion of fragments in peaks. `min_depth` is set to the maximum of 500 or 10 median library size.

## Value

indices of samples to keep

## See Also

[getCounts](#), [getPeaks](#), [filterPeaks](#)

## Examples

```
data(example_counts, package = "chromVAR")  
  
counts_filtered <- filterSamples(example_counts, min_depth = 1500,  
  min_in_peaks = 0.15, shiny = FALSE)  
counts_filtered_plot <- filterSamplesPlot(counts_filtered,  
  min_in_peaks = 0.15,  
  min_depth = 1500,  
  use_plotly = FALSE)
```

---

```
getAnnotationCorrelation
      getAnnotationCorrelation
```

---

## Description

getAnnotationCorrelation

## Usage

```
getAnnotationCorrelation(object, annotations, ...)
```

```
## S4 method for signature 'SummarizedExperiment,SummarizedExperiment'
getAnnotationCorrelation(object,
  annotations, background_peaks = getBackgroundPeaks(object),
  expectation = computeExpectations(object), variabilities = NULL)
```

```
## S4 method for signature 'SummarizedExperiment,MatrixOrmatrix'
getAnnotationCorrelation(object,
  annotations, background_peaks = getBackgroundPeaks(object),
  expectation = computeExpectations(object), variabilities = NULL)
```

```
## S4 method for signature 'SummarizedExperiment,list'
getAnnotationCorrelation(object,
  annotations, background_peaks = getBackgroundPeaks(object),
  expectation = computeExpectations(object), variabilities = NULL)
```

```
## S4 method for signature 'MatrixOrmatrix,SummarizedExperiment'
getAnnotationCorrelation(object,
  annotations, background_peaks, expectation = computeExpectations(object),
  variabilities = NULL)
```

```
## S4 method for signature 'MatrixOrmatrix,MatrixOrmatrix'
getAnnotationCorrelation(object,
  annotations, background_peaks, expectation = computeExpectations(object),
  variabilities = NULL)
```

```
## S4 method for signature 'MatrixOrmatrix,list'
getAnnotationCorrelation(object, annotations,
  background_peaks, expectation = computeExpectations(object),
  variabilities = NULL)
```

**Arguments**

object	result from computeDeviations
annotations	SummarizedExperiment of annotation matches
...	additional arguments
background_peaks	optional, matrix of background peaks
expectation	optional, expected fraction of reads per peak, as computed by computeExpectations
variabilities	optional, variabilities computed from computeVariability

**Details**

should only be run on small number of motifs/kmers/peaksets (very slow!)

**Value**

correlation matrix

**Methods (by class)**

- object = SummarizedExperiment, annotations = SummarizedExperiment: object and annotations are SummarizedExperiment
- object = SummarizedExperiment, annotations = MatrixOrmatrix: object is SummarizedExperiment, annotations are Matrix
- object = SummarizedExperiment, annotations = list: object is SummarizedExperiment, annotations are list
- object = MatrixOrmatrix, annotations = SummarizedExperiment: object and annotations are SummarizedExperiment
- object = MatrixOrmatrix, annotations = MatrixOrmatrix: object is SummarizedExperiment, annotations are Matrix
- object = MatrixOrmatrix, annotations = list: object is SummarizedExperiment, annotations are list

---

getAnnotations

*getAnnotations*

---

**Description**

getAnnotations

**Usage**

```
getAnnotations(annotations, ...)  
  
## S4 method for signature 'GRangesList'  
getAnnotations(annotations, rowRanges, ...)  
  
## S4 method for signature 'MatrixOrmatrix'  
getAnnotations(annotations, ...)  
  
## S4 method for signature 'data.frame'  
getAnnotations(annotations, ...)  
  
## S4 method for signature 'list'  
getAnnotations(annotations, npeaks = NULL, ...)  
  
## S4 method for signature 'character'  
getAnnotations(annotations, rowRanges, column = NULL,  
...)
```

**Arguments**

annotations	matrix, Matrix, or data.frame of fragment counts, or SummarizedExperiment with counts assays, see details
...	additional arguments to pass to SummarizedExperiment
rowRanges	GenomicRanges or GenomicRangesList or RangedSummarizedExperiment
npeaks	number of peaks
column	column of bed file with annotation names, see details

**Value**

SummarizedExperiment object with 'matches' assay

**Methods (by class)**

- GRangesList: get annotation matrix from GRangesList
- MatrixOrmatrix: get annotation matrix from Matrix or matrix
- data.frame: get annotation matrix from data.frame
- list: get annotation matrix from list
- character: get annotations from bed files

**Author(s)**

Alicia Schep

**Examples**

```
# First get example counts
data(mini_counts, package = "chromVAR")

# Get annotations from genomic ranges list
library(GenomicRanges)
library(SummarizedExperiment)
my_annotation_granges <- GRangesList(GRanges("chr1",
                                             ranges = IRanges(start =
                                                             c(566763,805090), width = 8)),
                                     GRanges("chr1", ranges = IRanges(start =
                                                             c(566792,895798), width = 8)))

anno_ix <- getAnnotations(my_annotation_granges,
                          rowRanges = rowRanges(mini_counts))
```

---

```
getAnnotationSynergy  getAnnotationSynergy
```

---

**Description**

```
getAnnotationSynergy
```

**Usage**

```
getAnnotationSynergy(object, annotations, ...)
```

```
## S4 method for signature 'SummarizedExperiment,SummarizedExperiment'
getAnnotationSynergy(object,
  annotations, background_peaks = getBackgroundPeaks(object),
  expectation = computeExpectations(object), variabilities = NULL,
  nbg = 25)
```

```
## S4 method for signature 'SummarizedExperiment,MatrixOrmatrix'
getAnnotationSynergy(object,
  annotations, background_peaks = getBackgroundPeaks(object),
  expectation = computeExpectations(object), variabilities = NULL,
  nbg = 25)
```

```
## S4 method for signature 'SummarizedExperiment,list'
getAnnotationSynergy(object, annotations,
  background_peaks = getBackgroundPeaks(object),
  expectation = computeExpectations(object), variabilities = NULL,
  nbg = 25)
```

```
## S4 method for signature 'MatrixOrmatrix,SummarizedExperiment'
getAnnotationSynergy(object,
```



```

    annotations, background_peaks, expectation = computeExpectations(object),
    variabilities = NULL, nbg = 25)

## S4 method for signature 'MatrixOrmatrix,MatrixOrmatrix'
getAnnotationSynergy(object,
    annotations, background_peaks, expectation = computeExpectations(object),
    variabilities = NULL, nbg = 25)

## S4 method for signature 'MatrixOrmatrix,list'
getAnnotationSynergy(object, annotations,
    background_peaks, expectation = computeExpectations(object),
    variabilities = NULL, nbg = 25)

```

### Arguments

object	result from computeDeviations
annotations	SummarizedExperiment of annotation matches
...	additional arguments
background_peaks	optional, matrix of background peaks
expectation	optional, expected fraction of reads per peak, as computed by computeExpectations
variabilities	optional, variabilities computed from computeVariability
nbg	number of background iterations

### Details

should only be run on small number of motifs/kmers/peaksets (very slow!)

### Value

synergy matrix

### Methods (by class)

- object = SummarizedExperiment, annotations = SummarizedExperiment: object and annotations are SummarizedExperiment
- object = SummarizedExperiment, annotations = MatrixOrmatrix: object is SummarizedExperiment, annotations are Matrix
- object = SummarizedExperiment, annotations = list: object is SummarizedExperiment, annotations are list
- object = MatrixOrmatrix, annotations = SummarizedExperiment: object and annotations are SummarizedExperiment
- object = MatrixOrmatrix, annotations = MatrixOrmatrix: object is SummarizedExperiment, annotations are Matrix
- object = MatrixOrmatrix, annotations = list: object is SummarizedExperiment, annotations are list

---

```
getBackgroundPeaks    getBackgroundPeaks
```

---

### Description

Function to get a set of background peaks for each peak based on GC content and # of fragments across all samples

### Usage

```
getBackgroundPeaks(object, ...)

## S4 method for signature 'SummarizedExperiment'
getBackgroundPeaks(object,
  bias = rowData(object)$bias, niterations = 50, w = 0.1, bs = 50)

## S4 method for signature 'RangedSummarizedExperiment'
getBackgroundPeaks(object,
  bias = rowRanges(object)$bias, niterations = 50, w = 0.1, bs = 50)

## S4 method for signature 'MatrixOrmatrix'
getBackgroundPeaks(object, bias, niterations = 50,
  w = 0.1, bs = 50)
```

### Arguments

object	fragment counts as SummarizedExperiment, RangedSummarized, Matrix, or matrix
...	additional arguments
bias	vector of values for some bias signal for each row of object
niterations	number of background peaks to sample
w	parameter controlling similarity of background peaks
bs	bin size parameter

### Details

Background peaks are chosen by sampling peaks based on similarity in GC content and # of fragments across samples using the Mahalanobis distance. The w paramter controls how similar background peaks should be. The bs parameter controls the precision with which the similarity is computed; increasing bs will make the function run slower. Sensible default parameters are chosen for both.

### Value

matrix with one row per peak and one column per iteration. values in a row represent indices of background peaks for the peak with that index

**Methods (by class)**

- SummarizedExperiment: method for SummarizedExperiment
- RangedSummarizedExperiment: method for RangedSummarizedExperiment
- MatrixOrmatrix: method for Matrix or matrix

**Examples**

```
# Load very small example counts (already filtered)
data(mini_counts, package = "chromVAR")

# get background peaks
bgpeaks <- getBackgroundPeaks(mini_counts)
```

---

getCisGroups	<i>getCisGroups</i>
--------------	---------------------

---

**Description**

Function for grouping peaks based on proximity along chromosomes

**Usage**

```
getCisGroups(object, ...)
```

```
## S4 method for signature 'RangedSummarizedExperiment'
getCisGroups(object, grpsize = 25,
              stepsize = 10)
```

```
## S4 method for signature 'GenomicRanges'
getCisGroups(object, grpsize = 25, stepsize = 10)
```

**Arguments**

object	GenomicRanges or RangedSummarizedExperiment
...	additional arguments
grpsize	number of peaks to include in each group
stepsize	number of peaks between each new set of groups

**Value**

SummarizedExperiment with annotationMatches assay storing which peaks belong to which groups

**Methods (by class)**

- RangedSummarizedExperiment: method for RangedSummarizedExperiment
- GenomicRanges: method for GenomicRanges

**Author(s)**

Alicia Schep

**Examples**

```
# Load very small example counts (already filtered)
data(mini_counts, package = "chromVAR")
mini_counts <- sort(mini_counts)
cisg <- getCisGroups(mini_counts)
```

---

`getCounts`*getCounts*

---

**Description**

makes matrix of fragment counts in peaks using one or multiple bam or bed files

**Usage**

```
getCounts(alignment_files, peaks, paired, by_rg = FALSE, format = c("bam",
  "bed"), colData = NULL)
```

**Arguments**

<code>alignment_files</code>	filenames for bam or bed files with aligned reads
<code>peaks</code>	GRanges object with peaks
<code>paired</code>	paired end data?
<code>by_rg</code>	use RG tags in bam to separate groups?
<code>format</code>	bam or bed? default is bam
<code>colData</code>	sample annotation DataFrame

**Value**

[RangedSummarizedExperiment-class](#) object

**See Also**

[getSampleDepths](#), [getPeaks](#), [filterSamples](#)

**Examples**

```
# First we'll read in some peaks
peaks_file <- system.file("extdata", "test_bed.txt", package = "chromVAR")
test_peaks <- getPeaks(peaks_file, sort = TRUE)

# With single bam with RG tags (can also give multiple bams with RG)
test_rg <- system.file("extdata", "test_RG.bam", package = "chromVAR")
test_counts <- getCounts(test_rg, peaks = test_peaks, by_rg = TRUE,
                        paired = TRUE,
                        colData = S4Vectors::DataFrame(condition = "A"))

# Multiple bams without RG tags
test_bam1 <- system.file("extdata", "test_single1.bam", package = "chromVAR")
test_bam2 <- system.file("extdata", "test_single2.bam", package = "chromVAR")
test_bam3 <- system.file("extdata", "test_single3.bam", package = "chromVAR")
test_counts2 <- getCounts(c(test_bam1, test_bam2, test_bam3),
                        peaks = test_peaks, by_rg = FALSE,
                        paired = TRUE,
                        colData = S4Vectors::DataFrame(celltype =
                                                    c("A", "B", "C")))

# Bed file with reads (can give multiple bed files, here we will just read 1)
test_bed <- system.file("extdata", "test_reads.bed", package = "chromVAR")
test_counts3 <- getCounts(test_bed, test_peaks, by_rg = FALSE,
                        paired = FALSE,
                        format = "bed")
```

---

```
getFragmentsPerPeak     getFragmentsPerPeak
```

---

**Description**

```
getFragmentsPerPeak
```

**Usage**

```
getFragmentsPerPeak(object)

## S4 method for signature 'SummarizedExperiment'
getFragmentsPerPeak(object)

## S4 method for signature 'MatrixOrmatrix'
getFragmentsPerPeak(object)
```

**Arguments**

```
object                    SummarizedExperiment, matrix, or Matrix object
```

**Value**

vector with sum across rows of counts assay within chromVARCounts

**Methods (by class)**

- SummarizedExperiment: method for SummarizedExperiment object with counts assay
- MatrixOrmatrix: method for Matrix or matrix object

**See Also**

[getFragmentsPerSample](#), [getTotalFragments](#)

**Examples**

```
# Load very small example counts (already filtered)
data(mini_counts, package = "chromVAR")

frags_per_peak <- getFragmentsPerPeak(mini_counts)
```

---

`getFragmentsPerSample` *getFragmentsPerSample*

---

**Description**

`getFragmentsPerSample`

**Usage**

```
getFragmentsPerSample(object)

## S4 method for signature 'SummarizedExperiment'
getFragmentsPerSample(object)

## S4 method for signature 'MatrixOrmatrix'
getFragmentsPerSample(object)
```

**Arguments**

`object` SummarizedExperiment, matrix, or Matrix object

**Value**

vector with sum across columns of counts assay within chromVARCounts

**Methods (by class)**

- SummarizedExperiment: method for SummarizedExperiment object with counts assay
- MatrixOrmatrix: method for Matrix or matrix object

**See Also**[getFragmentsPerPeak](#), [getTotalFragments](#)**Examples**

```
# Load very small example counts (already filtered)
data(mini_counts, package = "chromVAR")
frags_per_sample <- getFragmentsPerSample(mini_counts)
```

---

<code>getJasparMotifs</code>	<i>getJasparMotifs</i>
------------------------------	------------------------

---

**Description**

Function to get motifs from JASPAR database

**Usage**

```
getJasparMotifs(species = "Homo sapiens", collection = "CORE", ...)
```

**Arguments**

<code>species</code>	Which species? use either jaspar code or latin name. default is 'Homo sapiens'
<code>collection</code>	Which collection to use? default is 'CORE'
<code>...</code>	additional arguments to opts for <a href="#">getMatrixSet</a>

**Details**

Simply a wrapper function for [getMatrixSet](#) that calls JASPAR2016 database using [JASPAR2016](#)

**Value**

[PFMatrixList](#)

**Examples**

```
motifs <- getJasparMotifs()
```

---

getPeaks	<i>getPeaks</i>
----------	-----------------

---

### Description

Read in peaks from a bed file.

### Usage

```
getPeaks(filename, extra_cols = c(), sort_peaks = FALSE)
```

### Arguments

filename	filename of bed file
extra_cols	extra columns to read in beyond first three
sort_peaks	sort the peaks?

### Details

As in standard definition of bed file, first column is assumed to be chromosome, second is assumed to be start of peak (0-based), and third is assumed to be end of peak (1-based). Note that in output `GenomicRanges` output, start and end indices are both 1-based. Extra columns can be added as metadata or strand information if provided, but the user must indicate column index and name using named vector for `extra_cols`.

### Value

[GenomicRanges](#) containing peaks from file

### See Also

[getCounts](#), [filterPeaks](#), [readNarrowpeaks](#)

### Examples

```
peaks_file <- system.file("extdata", "test_bed.txt", package = "chromVAR")
peaks <- getPeaks(peaks_file, sort = TRUE)
```



---

getPermutedData	<i>getPermutedData</i>
-----------------	------------------------

---

### Description

Function to get permuted data while maintaining biases

### Usage

```
getPermutedData(object, niterations = 10, w = 0.1, bs = 50)
```

### Arguments

object	SummarizedExperiment
niterations	number of background peaks to sample
w	parameter controlling similarity of background peaks
bs	bin size parameter

### Details

Replaces the counts at a given peak with the count from another peak with similar GC content and average accessibility

### Value

new SummarizedExperiment with addition assays representing permuted version of counts

### Examples

```
# Load very small example counts (already filtered)
data(mini_counts, package = "chromVAR")

# get background peaks
perm_counts <- getPermutedData(mini_counts, niterations = 2)
```

---

getSampleCorrelation *getSampleCorrelation*

---

### Description

Get correlation between samples based on bias corrected deviations

### Usage

```
getSampleCorrelation(object, threshold = 1.5)
```

### Arguments

object	deviations result
threshold	threshold for variability

### Details

This function will compute the correlation between samples based on the normalized deviations. It will first remove correlated motifs/peak sets. Then the pearson correlation coefficient will be computed and returned.

### Value

correlation matrix between samples

### Author(s)

Alicia Schep

### See Also

[getSampleDistance](#)

### Examples

```
# Load very small example results from computeDeviations
data(mini_dev, package = "chromVAR")
sample_cor <- getSampleCorrelation(mini_dev, threshold = 0.8)
# setting very low variability threshold because this is mini data set
# threshold should generally be above 1
# Use plotVariability to get a sense of an appropriate threshold
# As this is mini data set, results probably not meaningful!
```

---

getSampleDepths	<i>getSampleDepths</i>
-----------------	------------------------

---

**Description**

makes vector of read depths in bam files or RG groups within bam files

**Usage**

```
getSampleDepths(alignment_files, paired = TRUE, by_rg = FALSE,  
  format = c("bam", "bed"))
```

**Arguments**

alignment_files	filenames for bam or bed file(s) with aligned reads
paired	paired end data?
by_rg	use RG tags to separate groups?
format	bam or bed format? default is bam

**Value**

numeric vector

**See Also**

[getCounts](#), [filterSamples](#)

**Examples**

```
# With single bam with RG tags (can also give multiple bams with RG)  
test_rg <- system.file("extdata", "test_RG.bam", package = "chromVAR")  
test_counts <- getSampleDepths(test_rg, by_rg = TRUE,  
  paired = TRUE)
```

```
# Multiple bams without RG tags  
test_bam1 <- system.file("extdata", "test_single1.bam", package = "chromVAR")  
test_bam2 <- system.file("extdata", "test_single2.bam", package = "chromVAR")  
test_bam3 <- system.file("extdata", "test_single3.bam", package = "chromVAR")  
test_counts2 <- getSampleDepths(c(test_bam1, test_bam2, test_bam3),  
  by_rg = FALSE,  
  paired = TRUE)
```

---

getSampleDistance      *getSampleDistance*

---

### Description

Get distance between samples based on bias corrected deviations

### Usage

```
getSampleDistance(object, threshold = 1.5, initial_dims = 50,  
  distance_function = dist)
```

### Arguments

object	deviations result
threshold	threshold for variability
initial_dims	initial dimentions for preliminary dimensionality reduction via pca
distance_function	distance function to use

### Details

This function will compute the distance between samples based on the normalized deviations. It will first remove correlated motifs / peak sets. Then the dimensionality will be further reduced via PCA if the number of dimensions exceeds initial\_dims. Then the supplied distance\_function will be used.

### Value

dist object for distance between samples

### Author(s)

Alicia Schep

### See Also

[getSampleCorrelation](#)

### Examples

```
# Load very small example results from computeDeviations  
data(mini_dev, package = "chromVAR")  
sample_dist <- getSampleDistance(mini_dev, threshold = 0.8)  
# setting very low variability threshold because this is mini data set  
# threshold should generally be above 1  
# Use plotVariability to get a sense of an appropriate threshold  
# As this is mini data set, results not meaningful!
```

---

`getTotalFragments`      *getTotalFragments*

---

### **Description**

`getTotalFragments`

### **Usage**

```
getTotalFragments(object)

## S4 method for signature 'SummarizedExperiment'
getTotalFragments(object)

## S4 method for signature 'MatrixOrmatrix'
getTotalFragments(object)
```

### **Arguments**

`object`                  SummarizedExperiment, matrix, or Matrix object

### **Value**

sum of all counts within object

### **Methods (by class)**

- SummarizedExperiment: method for SummarizedExperiment object with counts assay
- MatrixOrmatrix: method for Matrix or matrix object

### **See Also**

[getFragmentsPerSample](#), [getFragmentsPerPeak](#)

### **Examples**

```
# Load very small example counts (already filtered)
data(mini_counts, package = "chromVAR")
total_fragments <- getTotalFragments(mini_counts)
```

---

makeBiasBins	<i>makeBiasBins</i>
--------------	---------------------

---

## Description

Makes bins based on fragment counts

## Usage

```
makeBiasBins(object, ...)

## S4 method for signature 'SummarizedExperiment'
makeBiasBins(object,
  bias = rowData(object)$bias, nbins = 25, frac = 0.3)

## S4 method for signature 'RangedSummarizedExperiment'
makeBiasBins(object,
  bias = rowRanges(object)$bias, nbins = 25, frac = 0.3)

## S4 method for signature 'MatrixOrmatrix'
makeBiasBins(object, bias, nbins = 25,
  frac = 0.3)
```

## Arguments

object	fragment counts stored as RangedSummarizedExperiment, SummarizedExperiment, matrix, or Matrix
...	additional arguments
bias	vector of some bias signal (usually gc content) for each row of object
nbins	number of bins for each category, see Details
frac	fraction of peaks within given bin to select randomly

## Details

Will create  $\text{nbins} * 3$  annotations based on sampling from peaks with a certain fragment count, fragment count, or fragment count & bias.

## Value

SummarizedExperiment storing bias bins annotation

## Methods (by class)

- SummarizedExperiment: method for SummarizedExperiment
- RangedSummarizedExperiment: method for RangedSummarizedExperiment
- MatrixOrmatrix: method for Matrix or matrix

**Author(s)**

Alicia Schep

**Examples**

```
# Load very small example counts (already filtered)
data(mini_counts, package = "chromVAR")
bb <- makeBiasBins(mini_counts)
```

---

```
makePermutedSets      makePermutedSets
```

---

**Description**

Makes annotations sets with similar bias to input sets

**Usage**

```
makePermutedSets(object, annotations, ...)

## S4 method for signature 'SummarizedExperiment,SummarizedExperiment'
makePermutedSets(object,
  annotations, bias = rowData(object)$bias, window = 10)

## S4 method for signature 'RangedSummarizedExperiment,SummarizedExperiment'
makePermutedSets(object,
  annotations, bias = rowRanges(object)$bias, window = 10)

## S4 method for signature 'MatrixOrmatrix,SummarizedExperiment'
makePermutedSets(object,
  annotations, bias, window = 10)

## S4 method for signature 'SummarizedExperiment,MatrixOrmatrix'
makePermutedSets(object,
  annotations, bias = rowData(object)$bias, window = 10)

## S4 method for signature 'RangedSummarizedExperiment,MatrixOrmatrix'
makePermutedSets(object,
  annotations, bias = rowRanges(object)$bias, window = 10)

## S4 method for signature 'MatrixOrmatrix,MatrixOrmatrix'
makePermutedSets(object, annotations,
  bias, window = 10)

## S4 method for signature 'SummarizedExperiment,list'
makePermutedSets(object, annotations,
```

```

bias = rowData(object)$bias, window = 10)

## S4 method for signature 'RangedSummarizedExperiment,list'
makePermutedSets(object,
  annotations, bias = rowRanges(object)$bias, window = 10)

## S4 method for signature 'MatrixOrmatrix,list'
makePermutedSets(object, annotations, bias,
  window = 10)

```

### Arguments

object	fragment counts stored as RangedSummarizedExperiment, SummarizedExperiment, matrix, or Matrix
annotations	annotations as SummarizedExperiment, matrix, or list
...	additional arguments
bias	vector of some bias signal (usually gc content) for each row of object
window	number of nearest neighbors to consider

### Details

Will create  $\text{nbins} * 3$  annotations based on sampling from peaks with a certain fragment count, fragment count, or fragment count & bias.

### Value

SummarizedExperiment storing bias bins annotation

### Methods (by class)

- object = SummarizedExperiment, annotations = SummarizedExperiment: method for SummarizedExperiment and SummarizedExperiment
- object = RangedSummarizedExperiment, annotations = SummarizedExperiment: method for RangedSummarizedExperiment and SummarizedExperiment
- object = MatrixOrmatrix, annotations = SummarizedExperiment: method for Matrix or matrix and SummarizedExperiment
- object = SummarizedExperiment, annotations = MatrixOrmatrix: method for SummarizedExperiment and MatrixOrmatrix
- object = RangedSummarizedExperiment, annotations = MatrixOrmatrix: method for RangedSummarizedExperiment and MatrixOrmatrix
- object = MatrixOrmatrix, annotations = MatrixOrmatrix: method for Matrix/matrix and Matrix/matrix
- object = SummarizedExperiment, annotations = list: method for SummarizedExperiment and list
- object = RangedSummarizedExperiment, annotations = list: method for RangedSummarizedExperiment and list
- object = MatrixOrmatrix, annotations = list: method for Matrix or matrix and list



**Author(s)**

Alicia Schep

**Examples**

```
# Load very small example counts (already filtered)
data(mini_counts, package = "chromVAR")
data(example_motifs, package = "motifmatchr")
library(motifmatchr)
library(BSgenome.Hsapiens.UCSC.hg19)
motif_ix <- matchMotifs(example_motifs, mini_counts,
                       genome = BSgenome.Hsapiens.UCSC.hg19)

perm_sets <- makePermutedSets(mini_counts, motif_ix)
```

---

 matchKmers

---

*matchKmers*


---

**Description**

Find kmer matches in the DNA string-based subject

**Usage**

```
matchKmers(k, subject, ...)

## S4 method for signature 'character,DNAStringSet'
matchKmers(k, subject, out = c("matches",
                              "positions"), ranges = NULL)

## S4 method for signature 'character,character'
matchKmers(k, subject, out = c("matches",
                              "positions"), ranges = NULL)

## S4 method for signature 'character,DNAString'
matchKmers(k, subject, out = c("matches",
                              "positions"), ranges = NULL)

## S4 method for signature 'character,GenomicRanges'
matchKmers(k, subject,
           genome = GenomeInfoDb::genome(subject), out = c("matches", "positions"))

## S4 method for signature 'character,RangedSummarizedExperiment'
matchKmers(k, subject, ...)

## S4 method for signature 'numeric,ANY'
matchKmers(k, subject, ...)
```

```
## S4 method for signature 'DNAStrngSet,ANY'
matchKmers(k, subject, ...)
```

```
## S4 method for signature 'DNAStrng,ANY'
matchKmers(k, subject, ...)
```

### Arguments

k	k
subject	either <a href="#">GenomicRanges</a> , <a href="#">DNAStrngSet</a> , <a href="#">DNAStrng</a> , or character vector
...	additional arguments
out	what to return? see details
ranges	if subject is not <a href="#">GenomicRanges</a> , ranges to use when out is positions
genome	<a href="#">BSgenome</a> object, only used if subject is <a href="#">GenomicRanges</a>

### Details

Can either return a [SummarizedExperiment](#) with just sparse matrix with values set to 1 for a match (if `return == 'matches'`), or a [GenomicRanges](#) object with all the positions of matches

### Value

[SummarizedExperiment](#) with matches assay storing which peaks contain which kmers

### Methods (by class)

- k = character, subject = [DNAStrngSet](#): For [DNAStrngSet](#) Objects
- k = character, subject = character: For character strings
- k = character, subject = [DNAStrng](#): For DNA String objects
- k = character, subject = [GenomicRanges](#): For [GenomicRanges](#)
- k = character, subject = [RangedSummarizedExperiment](#): For [RangedSummarizedExperiment](#) (containing [GRanges](#) in `rowRanges`)
- k = numeric, subject = ANY: Catch-all for other un-documented types
- k = [DNAStrngSet](#), subject = ANY: Catch-all for other un-documented types with [DNAStrngSet](#)
- k = [DNAStrng](#), subject = ANY: Catch-all for other un-documented types with [DNAStrng](#)

### See Also

[getAnnotations](#), [computeDeviations](#)

**Examples**

```
# Load very small example counts (already filtered)
data(mini_counts, package = "chromVAR")

# Get peak-kmer annotation matrix for 6mers
library(BSgenome.Hsapiens.UCSC.hg19)
kmer_ix <- matchKmers(6, mini_counts,
                      genome = BSgenome.Hsapiens.UCSC.hg19)
```

---

mini_counts	<i>mini_counts</i>
-------------	--------------------

---

**Description**

Tiny sample data set for chromVAR function examples

**Usage**

```
data(mini_counts)
```

**Value**

[RangedSummarizedExperiment](#)

**See Also**

[mini\\_dev](#), [mini\\_ix](#)

**Examples**

```
data(mini_counts)
```

---

mini_dev	<i>mini_dev</i>
----------	-----------------

---

**Description**

Tiny sample chromVARDeviations object resulting from computeDeviations Result from running computeDeviations(mini\_counts, mini\_ix) on mini\_ix and mini\_counts data from this package

**Usage**

```
data(mini_dev)
```

**Value**

[chromVARDeviations-class](#)

**See Also**

[computeDeviations](#), [mini\\_counts](#), [mini\\_ix](#)

**Examples**

```
data(mini_dev)
```

---

<code>mini_ix</code>	<i>mini_ix</i>
----------------------	----------------

---

**Description**

Tiny sample annotation object for use in chromVAR examples Result from running `matchMotifs(example_motifs,mini_counts,"hg19)` on `example_motifs` from `motifmatchr` package and `mini_counts` from this package

**Usage**

```
data(mini_ix)
```

**Value**

[RangedSummarizedExperiment](#)

**See Also**

[mini\\_counts](#), [mini\\_dev](#)

**Examples**

```
data(mini_ix)
```

---

<code>plotDeviationsTsne</code>	<i>plotDeviationsTsne</i>
---------------------------------	---------------------------

---

**Description**

plots sample similarity tsne

**Usage**

```
plotDeviationsTsne(object, tsne, var_df = NULL, sample_column = NULL,
  annotation_name = NULL, shiny = interactive())
```

**Arguments**

object	deviations result object
tsne	result from <a href="#">deviationsTsne</a>
var_df	variability result
sample_column	column name for sample data – colData(object) – to be used for coloring points
annotation_name	name of chromVAR annotation for coloring points
shiny	return shiny app? otherwise return static plots

**Value**

shiny app or plots

**Author(s)**

Alicia Schep

---

plotKmerMismatch      *plotKmerMismatch*

---

**Description**

plotKmerMismatch

**Usage**

```
plotKmerMismatch(kmer, cov_mat, pval = 0.01)
```

**Arguments**

kmer	kmer, e.g. 'AAAAAAA'
cov_mat	result from <a href="#">deviationsCovariability</a>
pval	p value threshold

**Value**

A plot

---

plotVariability      *plotVariability*

---

**Description**

plot variability of motifs/etc

**Usage**

```
plotVariability(variability, xlab = "Sorted TFs", n = 3,
  labels = variability$name, use_plotly = interactive())
```

**Arguments**

variability	output from <a href="#">computeVariability</a>
xlab	label for x-axis (default is 'Sorted TFs')
n	number of toppoints to label?
labels	names of sets. if not given, uses rownames of variability
use_plotly	make plot interactive (using plotly)

**Value**

ggplot or plotly object, depending on whether use\_plotly is TRUE

**Author(s)**

Alicia Schep

**Examples**

```
# Load very small example results from computeDeviations
data(mini_dev, package = "chromVAR")
variability <- computeVariability(mini_dev)
var_plot <- plotVariability(variability, use_plotly = FALSE)
```

---

pwmDistance      *pwmDistance*

---

**Description**

computes distance between every pwm in a list or between pwms in one list with pwms in another

**Usage**

```
pwmDistance(x, y = NULL, min_overlap = 5)
```

**Arguments**

x list of pwms or pfms, see Details  
y list of pwms or pfms, see Details  
min\_overlap minimum number of basepairs overlapping between motifs

**Details**

The format of x and y should be a [PWMMatrixList](#) or [PFMatrixList](#) or a list of matrices with rows corresponding to "A","C","G","T" and columns summing to 1.

**Value**

a list with three matrices- 'dist' has the distance between each pair of motifs, 'strand' has the strand of the motif for the match, and 'offset' has the offset between the motifs.

**Examples**

```
motifs <- getJasparMotifs()
library(TFBSTools)
pwm_dists <- pwmDistance(toPWM(motifs[[1]]), toPWM(motifs[[2]]))
```

---

rbind,chromVARDeviations-method

*rbind method chromVARDeviations*

---

**Description**

Concatenates chromVARDeviations results for different sets of annotations

**Usage**

```
## S4 method for signature 'chromVARDeviations'
rbind(..., deparse.level = 1)
```

**Arguments**

... chromVARDeviations object to be combined  
deparse.level See `?base::rbind` for a description of this argument.

**Value**

chromVARDeviations object

**Author(s)**

Alicia Schep

**See Also**[chromVARDeviations-class](#)**Examples**

```
# Load very small example results from computeDeviations
data(mini_dev, package = "chromVAR")
doubledev <- rbind(mini_dev, mini_dev) #concatenate two of the same tother
```

---

readNarrowpeaks	<i>readNarrowpeaks</i>
-----------------	------------------------

---

**Description**

Reads in peaks in narrowpeaks format, as output by macs2. Uses summit as center of peak, and makes peak the given 'width'. By default removes overlapping peaks to get set of peaks with no overlaps

**Usage**

```
readNarrowpeaks(filename, width = 500, non_overlapping = TRUE)
```

**Arguments**

filename	filename
width	desired width of peaks
non_overlapping	remove overlapping peaks

**Value**[GRanges-class](#)



# Index

- \* **datasets**
  - example\_counts, [17](#)
  - mini\_counts, [43](#)
  - mini\_dev, [43](#)
  - mini\_ix, [44](#)
- addGCBias, [3](#)
- addGCBias, RangedSummarizedExperiment-method
  - (addGCBias), [3](#)
- addGCBias, SummarizedExperiment-method
  - (addGCBias), [3](#)
- annoation\_matches<-, SummarizedExperiment-method
  - (annotationMatches), [4](#)
- annotationMatches, [4](#)
- annotationMatches, SummarizedExperiment-method
  - (annotationMatches), [4](#)
- annotationMatches<-
  - (annotationMatches), [4](#)
- annotationMatches<-, SummarizedExperiment-method
  - (annotationMatches), [4](#)
- assembleKmers, [5](#)
- bplapply, [9](#)
- cbind, chromVARDeviations-method, [6](#)
- chromVAR, [6](#)
- chromVAR-package (chromVAR), [6](#)
- chromVAR\_theme, [7](#)
- chromVARDeviations-class, [7](#)
- computeDeviations, [7](#), [8](#), [11](#), [42](#), [44](#)
- computeDeviations, MatrixOrmatrix, list-method
  - (computeDeviations), [8](#)
- computeDeviations, MatrixOrmatrix, MatrixOrmatrix-method
  - (computeDeviations), [8](#)
- computeDeviations, MatrixOrmatrix, missingOrNULL-method
  - (computeDeviations), [8](#)
- computeDeviations, MatrixOrmatrix, SummarizedExperiment-method
  - (computeDeviations), [8](#)
- computeDeviations, SummarizedExperiment, list-method
  - (computeDeviations), [8](#)
- computeDeviations, SummarizedExperiment, MatrixOrmatrix-method
  - (computeDeviations), [8](#)
- computeDeviations, SummarizedExperiment, missingOrNULL-method
  - (computeDeviations), [8](#)
- computeDeviations, SummarizedExperiment, SummarizedExperiment-method
  - (computeDeviations), [8](#)
- computeExpectations, [9](#), [10](#)
- computeExpectations, MatrixOrmatrix-method
  - (computeExpectations), [10](#)
- computeExpectations, SummarizedExperiment-method
  - (computeExpectations), [10](#)
- computeVariability, [9](#), [11](#), [46](#)
- counts, SummarizedExperiment-method, [12](#)
- counts<-, SummarizedExperiment, MatrixOrmatrix-method
  - (counts, SummarizedExperiment-method), [12](#)
- counts<-, SummarizedExperiment-method
  - (counts, SummarizedExperiment-method), [12](#)
- counts<-, SummarizedExperiment-method
  - (counts, SummarizedExperiment-method), [12](#)
- deviations, [7](#), [13](#)
- deviations, chromVARDeviations-method
  - (deviations), [13](#)
- deviationScores, [7](#), [13](#)
- deviationScores, chromVARDeviations-method
  - (deviationScores), [13](#)
- deviationsCovariability, [14](#), [45](#)
- deviationsTsne, [15](#), [45](#)
- differentialDeviations, [16](#)
- differentialVariability, [17](#)
- DNAStrng, [42](#)
- DNAStrngSet, [42](#)
- example\_counts, [17](#)
- filterPeaks, [18](#), [19](#), [20](#), [32](#)
- filterSamples, [18](#), [19](#), [28](#), [35](#)
- filterSamplesPlot, [20](#)
- GenomicRanges, [32](#), [42](#)

- getAnnotationCorrelation, 21
- getAnnotationCorrelation, MatrixOrmatrix, list-method (getAnnotationCorrelation), 21
- getAnnotationCorrelation, MatrixOrmatrix, MatrixOrmatrix-method (getAnnotationCorrelation), 21
- getAnnotationCorrelation, MatrixOrmatrix, SummarizedExperiment-method (getAnnotationCorrelation), 21
- getAnnotationCorrelation, SummarizedExperiment, list-method (getAnnotationCorrelation), 21
- getAnnotationCorrelation, SummarizedExperiment, MatrixOrmatrix-method (getAnnotationCorrelation), 21
- getAnnotationCorrelation, SummarizedExperiment, SummarizedExperiment-method (getAnnotationCorrelation), 21
- getAnnotations, 22, 42
- getAnnotations, character-method (getAnnotations), 22
- getAnnotations, data.frame-method (getAnnotations), 22
- getAnnotations, GRangesList-method (getAnnotations), 22
- getAnnotations, list-method (getAnnotations), 22
- getAnnotations, MatrixOrmatrix-method (getAnnotations), 22
- getAnnotationSynergy, 24
- getAnnotationSynergy, MatrixOrmatrix, list-method (getAnnotationSynergy), 24
- getAnnotationSynergy, MatrixOrmatrix, MatrixOrmatrix-method (getAnnotationSynergy), 24
- getAnnotationSynergy, MatrixOrmatrix, SummarizedExperiment-method (getAnnotationSynergy), 24
- getAnnotationSynergy, SummarizedExperiment, list-method (getAnnotationSynergy), 24
- getAnnotationSynergy, SummarizedExperiment, MatrixOrmatrix-method (getAnnotationSynergy), 24
- getAnnotationSynergy, SummarizedExperiment, SummarizedExperiment-method (getAnnotationSynergy), 24
- getBackgroundPeaks, 9, 26
- getBackgroundPeaks, MatrixOrmatrix-method (getBackgroundPeaks), 26
- getBackgroundPeaks, RangedSummarizedExperiment-method (getBackgroundPeaks), 26
- getBackgroundPeaks, SummarizedExperiment-method (getBackgroundPeaks), 26
- getCisGroups, 27
- getCisGroups, GenomicRanges-method (getCisGroups), 27
- getCisGroups, RangedSummarizedExperiment-method (getCisGroups), 27
- getCisGroups, 27
- getFragments, 18–20, 28, 32, 35
- getFragmentsPerPeak, 29, 31, 37
- getFragmentsPerPeak, MatrixOrmatrix-method (getFragmentsPerPeak), 29
- getFragmentsPerPeak, SummarizedExperiment-method (getFragmentsPerPeak), 29
- getFragmentsPerSample, 30, 30, 37
- getFragmentsPerSample, MatrixOrmatrix-method (getFragmentsPerSample), 30
- getFragmentsPerSample, SummarizedExperiment-method (getFragmentsPerSample), 30
- getJasparMotifs, 31
- getMatrixSet, 31
- getPeaks, 18–20, 28, 32
- getPermutedData, 33
- getSampleCorrelation, 34, 36
- getSampleDepths, 28, 35
- getSampleDistance, 34, 36
- getTotalFragments, 30, 31, 37
- getTotalFragments, MatrixOrmatrix-method (getTotalFragments), 37
- getTotalFragments, SummarizedExperiment-method (getTotalFragments), 37
- JASPAR2016, 31
- makeBiasBins, 38
- makeBiasBins, MatrixOrmatrix-method (makeBiasBins), 38
- makeBiasBins, RangedSummarizedExperiment-method (makeBiasBins), 38
- makeBiasBins, SummarizedExperiment-method (makeBiasBins), 38
- makePermutedSets, 39
- makePermutedSets, MatrixOrmatrix, list-method (makePermutedSets), 39
- makePermutedSets, MatrixOrmatrix, MatrixOrmatrix-method (makePermutedSets), 39
- makePermutedSets, MatrixOrmatrix, SummarizedExperiment-method (makePermutedSets), 39
- makePermutedSets, RangedSummarizedExperiment, list-method (makePermutedSets), 39
- makePermutedSets, RangedSummarizedExperiment, MatrixOrmatrix-method (makePermutedSets), 39
- makePermutedSets, RangedSummarizedExperiment, SummarizedExperiment-method (makePermutedSets), 39
- makePermutedSets, SummarizedExperiment, list-method (makePermutedSets), 39

makePermutedSets, SummarizedExperiment, MatrixOrmatrix-method  
    (makePermutedSets), 39

makePermutedSets, SummarizedExperiment, SummarizedExperiment-method  
    (makePermutedSets), 39

matchKmers, 41

matchKmers, character, character-method  
    (matchKmers), 41

matchKmers, character, DNASTring-method  
    (matchKmers), 41

matchKmers, character, DNASTringSet-method  
    (matchKmers), 41

matchKmers, character, GenomicRanges-method  
    (matchKmers), 41

matchKmers, character, RangedSummarizedExperiment-method  
    (matchKmers), 41

matchKmers, DNASTring, ANY-method  
    (matchKmers), 41

matchKmers, DNASTringSet, ANY-method  
    (matchKmers), 41

matchKmers, numeric, ANY-method  
    (matchKmers), 41

mini\_counts, 43, 44

mini\_dev, 43, 43, 44

mini\_ix, 43, 44, 44

PFMatrixList, 31, 47

plotDeviationsTsne, 44

plotKmerMismatch, 45

plotVariability, 9, 46

PWMatrixList, 47

pwmDistance, 46

RangedSummarizedExperiment, 18, 43, 44

rbind, chromVARDeviations-method, 47

readNarrowpeaks, 32, 48

SummarizedExperiment, 7