

# Package: chromVAR (via r-universe)

June 7, 2026

**Type** Package

**Title** Chromatin Variation Across Regions

**Version** 1.34.1

**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, Seqinfo, GenomicRanges, ggplot2, nabor, BiocParallel, BiocGenerics, Biostrings, pwalgn, 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++14

**VignetteBuilder** knitr

**RoxygenNote** 7.3.2

**Config/pak/sysreqs** cmake make libgs10-dev libbz2-dev libicu-dev liblzma-dev libuv1-dev libxml2-dev libssl-dev xz-utils zlib1g-dev

**Repository** <https://bioc-release.r-universe.dev>

**Date/Publication** 2026-05-08 04:21:24 UTC

**RemoteUrl** <https://github.com/bioc/chromVAR>

**RemoteRef** RELEASE\_3\_23

**RemoteSha** ffff247d9c99c5fc6d705479b643929cfaabd5a6

## Contents

addGCBias	3
annotationMatches	4
assembleKmers	5
cbind,chromVARDeviations-method	5
chromVAR	6
chromVAR_theme	6
chromVARDeviations-class	7
computeDeviations	7
computeExpectations	10
computeVariability	11
counts,SummarizedExperiment-method	12
deviations	13
deviationScores	14
deviationsCovariability	14
deviationsTsne	15
differentialDeviations	16
differentialVariability	17
example_counts	18
filterPeaks	18
filterSamples	19
filterSamplesPlot	21
getAnnotationCorrelation	22
getAnnotations	24
getAnnotationSynergy	25
getBackgroundPeaks	28
getCisGroups	29
getCounts	30
getFragmentsPerPeak	32
getFragmentsPerSample	33
getJasparMotifs	34
getPeaks	34
getPermutedData	35
getSampleCorrelation	36
getSampleDepths	37
getSampleDistance	38
getTotalFragments	39
makeBiasBins	40
makePermutedSets	41
matchKmers	43
mini_counts	45
mini_dev	46
mini_ix	46
plotDeviationsTsne	47
plotKmerMismatch	48
plotVariability	48
pwmDistance	49



---

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	<i>cbind method for chromVARDeviations</i>
---------------------------------	--

---

**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.

**Author(s)**

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

Other contributors:

- Jason Buenrostro [contributor]
- Caleb Lareau [contributor]
- William Greenleaf [thesis advisor]
- Stanford University [copyright holder]

---

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**

base_size	base font size
base_family	base font family

**Value**

ggplot2 theme

**Author(s)**

Alicia Schep

**Examples**

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

---

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](#))

---

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)

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

- computeExpectations(MatrixOrmatrix): method for Matrix or matrix
- computeExpectations(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 computeDeviations
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            *deviations*

---

**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	<i>deviationsCovariability</i>
-------------------------	--------------------------------

---

**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	<i>example_counts</i>
----------------	-----------------------

---

### Description

Very small sample data set for trying out chromVAR

### Usage

```
data(example_counts)
```

### Value

[RangedSummarizedExperiment](#)

### Examples

```
data(example_counts)
```

---

filterPeaks	<i>filterPeaks</i>
-------------	--------------------

---

### 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](#)

**Examples**

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

---

filterSamples

*filterSamples*

---

**Description**

function to get indices of samples that pass filters





---

```
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)**

- `getAnnotationCorrelation( object = SummarizedExperiment, annotations = SummarizedExperiment )`: object and annotations are SummarizedExperiment
- `getAnnotationCorrelation( object = SummarizedExperiment, annotations = MatrixOrmatrix )`: object is SummarizedExperiment, annotations are Matrix
- `getAnnotationCorrelation(object = SummarizedExperiment, annotations = list)`: object is SummarizedExperiment, annotations are list

- `getAnnotationCorrelation( object = MatrixOrmatrix, annotations = SummarizedExperiment )`: object and annotations are `SummarizedExperiment`
- `getAnnotationCorrelation(object = MatrixOrmatrix, annotations = MatrixOrmatrix)`: object is `SummarizedExperiment`, annotations are `Matrix`
- `getAnnotationCorrelation(object = MatrixOrmatrix, annotations = list)`: object is `SummarizedExperiment`, annotations are `list`

---

<code>getAnnotations</code>	<i>getAnnotations</i>
-----------------------------	-----------------------

---

## 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

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

## Value

`SummarizedExperiment` object with 'matches' assay

**Methods (by class)**

- `getAnnotations(GRangesList)`: get annotation matrix from GRangesList
- `getAnnotations(MatrixOrmatrix)`: get annotation matrix from Matrix or matrix
- `getAnnotations(data.frame)`: get annotation matrix from data.frame
- `getAnnotations(list)`: get annotation matrix from list
- `getAnnotations(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)

- `getAnnotationSynergy(object = SummarizedExperiment, annotations = SummarizedExperiment)`: object and annotations are SummarizedExperiment
- `getAnnotationSynergy(object = SummarizedExperiment, annotations = MatrixOrmatrix)`: object is SummarizedExperiment, annotations are Matrix
- `getAnnotationSynergy(object = SummarizedExperiment, annotations = list)`: object is SummarizedExperiment, annotations are list
- `getAnnotationSynergy(object = MatrixOrmatrix, annotations = SummarizedExperiment)`: object and annotations are SummarizedExperiment
- `getAnnotationSynergy(object = MatrixOrmatrix, annotations = MatrixOrmatrix)`: object is SummarizedExperiment, annotations are Matrix
- `getAnnotationSynergy(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

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

- `getBackgroundPeaks(SummarizedExperiment)`: method for `SummarizedExperiment`
- `getBackgroundPeaks(RangedSummarizedExperiment)`: method for `RangedSummarizedExperiment`
- `getBackgroundPeaks(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`

*getCisGroups*

---

**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)**

- `getCisGroups(RangedSummarizedExperiment)`: method for RangedSummarizedExperiment
- `getCisGroups(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**

alignment_files	filenames for bam or bed files with aligned reads
peaks	GRanges object with peaks
paired	paired end data?
by_rg	use RG tags in bam to separate groups?
format	bam or bed? default is bam
colData	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)

- `getFragmentsPerPeak(SummarizedExperiment)`: method for SummarizedExperiment object with counts assay
- `getFragmentsPerPeak(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)

- `getFragmentsPerSample(SummarizedExperiment)`: method for SummarizedExperiment object with counts assay
- `getFragmentsPerSample(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)
```

---

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

---

**Description**

Function to get motifs from JASPAR database

**Usage**

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

**Arguments**

species	Which species? use either jasper code or latin name. default is 'Homo sapiens'
collection	Which collection to use? default is 'CORE'
...	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)
```

---

<code>getPermutedData</code>	<code>getPermutedData</code>
------------------------------	------------------------------

---

**Description**

Function to get permuted data while maintaining biases

**Usage**

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

**Arguments**

<code>object</code>	SummarizedExperiment
<code>niterations</code>	number of background peaks to sample
<code>w</code>	parameter controlling similarity of background peaks
<code>bs</code>	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**

<code>object</code>	deviations result
<code>threshold</code>	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)
```

---

<code>getSampleDistance</code>	<i>getSampleDistance</i>
--------------------------------	--------------------------

---

**Description**

Get distance between samples based on bias corrected deviations

**Usage**

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

**Arguments**

<code>object</code>	deviations result
<code>threshold</code>	threshold for variability
<code>initial_dims</code>	initial dimentions for preliminary dimensionality reduction via pca
<code>distance_function</code>	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)**

- `getTotalFragments(SummarizedExperiment)`: method for SummarizedExperiment object with counts assay
- `getTotalFragments(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_frgs <- getTotalFragments(mini_counts)
```

---

makeBiasBins

*makeBiasBins*

---

**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**

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

**Details**

Will create 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)**

- `makeBiasBins(SummarizedExperiment)`: method for SummarizedExperiment
- `makeBiasBins(RangedSummarizedExperiment)`: method for RangedSummarizedExperiment
- `makeBiasBins(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)
```

---

<code>makePermutedSets</code>	<i>makePermutedSets</i>
-------------------------------	-------------------------

---

**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)**

- `makePermutedSets(object = SummarizedExperiment, annotations = SummarizedExperiment)`: method for `SummarizedExperiment` and `SummarizedExperiment`
- `makePermutedSets(object = RangedSummarizedExperiment, annotations = SummarizedExperiment)`: method for `RangedSummarizedExperiment` and `SummarizedExperiment`
- `makePermutedSets(object = MatrixOrmatrix, annotations = SummarizedExperiment)`: method for `Matrix` or `matrix` and `SummarizedExperiment`
- `makePermutedSets(object = SummarizedExperiment, annotations = MatrixOrmatrix)`: method for `SummarizedExperiment` and `MatrixOrmatrix`
- `makePermutedSets(object = RangedSummarizedExperiment, annotations = MatrixOrmatrix)`: method for `RangedSummarizedExperiment` and `MatrixOrmatrix`
- `makePermutedSets(object = MatrixOrmatrix, annotations = MatrixOrmatrix)`: method for `Matrix/matrix` and `Matrix/matrix`
- `makePermutedSets(object = SummarizedExperiment, annotations = list)`: method for `SummarizedExperiment` and `list`
- `makePermutedSets(object = RangedSummarizedExperiment, annotations = list)`: method for `RangedSummarizedExperiment` and `list`
- `makePermutedSets(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 = Seqinfo::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 'DNAStringSet,ANY'
matchKmers(k, subject, ...)

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

```

**Arguments**

k	k
subject	either <a href="#">GenomicRanges</a> , <a href="#">DNAStringSet</a> , <a href="#">DNAString</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	BSgenome 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)**

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

**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

*mini\_counts*

---

**Description**

Tiny sample data set for chromVAR funtion 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)
```

---

mini_ix	<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)
```

---

`plotDeviationsTsne`     *plotDeviationsTsne*

---

**Description**

plots sample similarity tsne

**Usage**

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

**Arguments**

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

---

### 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, 18
  - mini\_counts, 45
  - mini\_dev, 46
  - mini\_ix, 46
- 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, 5
- chromVAR, 6
- chromVAR-package (chromVAR), 6
- chromVAR\_theme, 6
- chromVARDeviations-class, 7
- computeDeviations, 7, 7, 12, 45, 46
- computeDeviations, MatrixOrmatrix, list-method
  - (computeDeviations), 7
- computeDeviations, MatrixOrmatrix, MatrixOrmatrix-method
  - (computeDeviations), 7
- computeDeviations, MatrixOrmatrix, missingOrNULL-method
  - (computeDeviations), 7
- computeDeviations, MatrixOrmatrix, SummarizedExperiment-method
  - (computeDeviations), 7
- computeDeviations, SummarizedExperiment, list-method
  - (computeDeviations), 7
- computeDeviations, SummarizedExperiment, MatrixOrmatrix-method
  - (computeDeviations), 7
- computeDeviations, SummarizedExperiment, missingOrNULL-method
  - (computeDeviations), 7
- computeDeviations, SummarizedExperiment, SummarizedExperiment-method
  - (computeDeviations), 7
- computeExpectations, 9, 10
- computeExpectations, MatrixOrmatrix-method
  - (computeExpectations), 10
- computeExpectations, SummarizedExperiment-method
  - (computeExpectations), 10
- computeVariability, 10, 11, 48
- 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, 14
- deviationScores, chromVARDeviations-method
  - (deviationScores), 14
- deviationsCovariability, 14, 48
- deviationsTsne, 15, 47
- differentialDeviations, 16
- differentialVariability, 17
- DNAStrng, 44
- DNAStrngSet, 44
- example\_counts, 18
- filterPeaks, 18, 20, 21, 35
- filterSamples, 19, 19, 31, 37
- filterSamplesPlot, 21
- GenomicRanges, 35, 44

- getAnnotationCorrelation, 22
- getAnnotationCorrelation, MatrixOrmatrix, list-method (getAnnotationCorrelation), 22
- getAnnotationCorrelation, MatrixOrmatrix, MatrixOrmatrix-method (getAnnotationCorrelation), 22
- getAnnotationCorrelation, MatrixOrmatrix, SummarizedExperiment-method (getAnnotationCorrelation), 22
- getAnnotationCorrelation, SummarizedExperiment, list-method (getAnnotationCorrelation), 22
- getAnnotationCorrelation, SummarizedExperiment, MatrixOrmatrix-method (getAnnotationCorrelation), 22
- getAnnotationCorrelation, SummarizedExperiment, SummarizedExperiment-method (getAnnotationCorrelation), 22
- getAnnotations, 24, 45
- getAnnotations, character-method (getAnnotations), 24
- getAnnotations, data.frame-method (getAnnotations), 24
- getAnnotations, GRangesList-method (getAnnotations), 24
- getAnnotations, list-method (getAnnotations), 24
- getAnnotations, MatrixOrmatrix-method (getAnnotations), 24
- getAnnotationSynergy, 25
- getAnnotationSynergy, MatrixOrmatrix, list-method (getAnnotationSynergy), 25
- getAnnotationSynergy, MatrixOrmatrix, MatrixOrmatrix-method (getAnnotationSynergy), 25
- getAnnotationSynergy, MatrixOrmatrix, SummarizedExperiment-method (getAnnotationSynergy), 25
- getAnnotationSynergy, SummarizedExperiment, list-method (getAnnotationSynergy), 25
- getAnnotationSynergy, SummarizedExperiment, MatrixOrmatrix-method (getAnnotationSynergy), 25
- getAnnotationSynergy, SummarizedExperiment, SummarizedExperiment-method (getAnnotationSynergy), 25
- getBackgroundPeaks, 9, 28
- getBackgroundPeaks, MatrixOrmatrix-method (getBackgroundPeaks), 28
- getBackgroundPeaks, RangedSummarizedExperiment-method (getBackgroundPeaks), 28
- getBackgroundPeaks, SummarizedExperiment-method (getBackgroundPeaks), 28
- getCisGroups, 29
- getCisGroups, GenomicRanges-method (getCisGroups), 29
- getCisGroups, RangedSummarizedExperiment-method (getCisGroups), 29
- getCisGroups, 29
- getFragments, 19–21, 30, 35, 37
- getFragmentsPerPeak, 32, 33, 40
- getFragmentsPerPeak, MatrixOrmatrix-method (getFragmentsPerPeak), 32
- getFragmentsPerPeak, SummarizedExperiment-method (getFragmentsPerPeak), 32
- getFragmentsPerSample, 32, 33, 40
- getFragmentsPerSample, MatrixOrmatrix-method (getFragmentsPerSample), 33
- getFragmentsPerSample, SummarizedExperiment-method (getFragmentsPerSample), 33
- getJasparMotifs, 34
- getMatrixSet, 34
- getPeaks, 19–21, 31, 34
- getPermutedData, 35
- getSampleCorrelation, 36, 39
- getSampleDepths, 31, 37
- getSampleDistance, 36, 38
- getTotalFragments, 32, 33, 39
- getTotalFragments, MatrixOrmatrix-method (getTotalFragments), 39
- getTotalFragments, SummarizedExperiment-method (getTotalFragments), 39
- JASPAR2016, 34
- makeBiasBins, 40
- makeBiasBins, MatrixOrmatrix-method (makeBiasBins), 40
- makeBiasBins, RangedSummarizedExperiment-method (makeBiasBins), 40
- makeBiasBins, SummarizedExperiment-method (makeBiasBins), 40
- makePermutedSets, 41
- makePermutedSets, MatrixOrmatrix, list-method (makePermutedSets), 41
- makePermutedSets, MatrixOrmatrix, MatrixOrmatrix-method (makePermutedSets), 41
- makePermutedSets, MatrixOrmatrix, SummarizedExperiment-method (makePermutedSets), 41
- makePermutedSets, RangedSummarizedExperiment, list-method (makePermutedSets), 41
- makePermutedSets, RangedSummarizedExperiment, MatrixOrmatrix-method (makePermutedSets), 41
- makePermutedSets, RangedSummarizedExperiment, SummarizedExperiment-method (makePermutedSets), 41
- makePermutedSets, SummarizedExperiment, list-method (makePermutedSets), 41

makePermutedSets, SummarizedExperiment, MatrixOrmatrix-method  
(makePermutedSets), 41

makePermutedSets, SummarizedExperiment, SummarizedExperiment-method  
(makePermutedSets), 41

matchKmers, 43

matchKmers, character, character-method  
(matchKmers), 43

matchKmers, character, DNASTring-method  
(matchKmers), 43

matchKmers, character, DNASTringSet-method  
(matchKmers), 43

matchKmers, character, GenomicRanges-method  
(matchKmers), 43

matchKmers, character, RangedSummarizedExperiment-method  
(matchKmers), 43

matchKmers, DNASTring, ANY-method  
(matchKmers), 43

matchKmers, DNASTringSet, ANY-method  
(matchKmers), 43

matchKmers, numeric, ANY-method  
(matchKmers), 43

mini\_counts, 45, 46, 47

mini\_dev, 46, 46, 47

mini\_ix, 46, 46

PFMatrixList, 34, 49

plotDeviationsTsne, 47

plotKmerMismatch, 48

plotVariability, 10, 48

PWMatrixList, 49

pwmDistance, 49

RangedSummarizedExperiment, 18, 45, 46

rbind, chromVARDeviations-method, 50

readNarrowpeaks, 35, 50

SummarizedExperiment, 7