## Package ‘VplotR’

May 7, 2024

<table>
<thead>
<tr>
<th>Type</th>
<th>Package</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title</td>
<td>Set of tools to make V-plots and compute footprint profiles</td>
</tr>
<tr>
<td>Version</td>
<td>1.14.0</td>
</tr>
<tr>
<td>Date</td>
<td>2021-11-21</td>
</tr>
<tr>
<td>Encoding</td>
<td>UTF-8</td>
</tr>
</tbody>
</table>

**Description**  The pattern of digestion and protection from DNA nucleases such as DNase I, micrococcal nuclease, and Tn5 transposase can be used to infer the location of associated proteins. This package contains useful functions to analyze patterns of paired-end sequencing fragment density. VplotR facilitates the generation of V-plots and footprint profiles over single or aggregated genomic loci of interest.

**URL**  [https://github.com/js2264/VplotR](https://github.com/js2264/VplotR)

**BugReports**  [https://github.com/js2264/VplotR/issues](https://github.com/js2264/VplotR/issues)

**Depends**  R (>= 4.0), GenomicRanges, IRanges, ggplot2

**Imports**  cowplot, magrittr, GenomeInfoDb, GenomicAlignments, RColorBrewer, zoo, Rsamtools, S4Vectors, parallel, reshape2, methods, graphics, stats

**Suggests**  GenomicFeatures, TxDb.Scerevisiae.UCSC.sacCer3.sgdGene, testthat, covr, knitr, rmarkdown, pkgdown

**VignetteBuilder**  knitr

**biocViews**  NucleosomePositioning, Coverage, Sequencing, BiologicalQuestion, ATACSeq, Alignment

**License**  GPL (>= 3)

**RoxygenNote**  7.2.3

**git_url**  [https://git.bioconductor.org/packages/VplotR](https://git.bioconductor.org/packages/VplotR)

**git_branch**  RELEASE_3_19

**git_last_commit**  b7629ca

**git_last_commit_date**  2024-04-30

**Repository**  Bioconductor 3.19
Description

Genomic loci with a REB1 binding motifs according to http://jaspar.genereg.net/api/v1/matrix/MA0265.1.jaspar. PWM and scanning done with TFBSTools.

Usage

data(ABF1_sacCer3)

Format

An object of class "GRanges".

References

Rossi, Lai & Pugh 2018 Genome Research

Examples

data(ABF1_sacCer3)

alignToTSS

A function to re-align a GRanges object to TSSs

Description

This function re-aligns ranges (typically regulatory elements) to a set of coordinates, either the TSS column or the TSS.fwd and TSS.rev columns. If none are found, the function assumes the ranges are promoters and that the end or the ranges are the TSSs.

Usage

alignToTSS(granges, upstream = 0, downstream = 1)

Arguments

granges A stranded GRanges object with a TSS column or TSS.rev and TSS.fwd columns
upstream How many bases upstream of the TSS should the GRanges object be extended by? [Default: 0]
downstream How many bases downstream of the TSS should the GRanges object be extended by? [Default: 1]
**Value**

GRanges aligned to the TSS column or to TSS.rev and TSS.fwd columns, and extended by up-steam/downstream bp.

**Examples**

```r
data(cell_proms)
cell_proms
alignToTSS(cell_proms)
```

---

**Description**

A sample of ATAC-seq fragments from individual worm tissues (Serizay et al. 2020, "Tissue-specific profiling reveals distinctive regulatory architectures for ubiquitous, germline and somatic genes", BiorXiv)

**Usage**

```r
data(ATAC_cell1_Serizay2020)
```

**Format**

An object of class "list".

**Examples**

```r
data(ATAC_cell1_Serizay2020)
ATAC_cell1_Serizay2020
```

---

**Description**

A .bam file sample

**Usage**

```r
data(bam_test)
```

**Format**

An object of class "GRanges".
ce11_all_REs

Examples

```r
data(bam_test)
bam_test
```

table(ce11_all_REs$regulatory_class)
table(ce11_all_REs$which.tissues)

Description


Usage

```r
data(ce11_all_REs)
```

Format

GRanges

Source

BiorXiv

References

Serizay et al. 2020, "Tissue-specific profiling reveals distinctive regulatory architectures for ubiquitous, germline and somatic genes", BiorXiv. (DOI)
computeNucleosomeEnrichmentOverBackground

---

**Description**

Promoters annotated in *C. elegans* (ce11) according to Serizay et al. 2020, "Tissue-specific profiling reveals distinctive regulatory architectures for ubiquitous, germline and somatic genes", BiorXiv.

**Usage**

```r
data(ce11_proms)
```

**Format**

An object of class "GRanges".

**Source**

BiorXiv

**References**

Serizay et al. 2020, "Tissue-specific profiling reveals distinctive regulatory architectures for ubiquitous, germline and somatic genes", BiorXiv. (DOI)

**Examples**

```r
data(ce11_proms)
table(ce11_proms$which.tissues)
```

---

computeNucleosomeEnrichmentOverBackground

*Internal function*

---

**Description**

A function to compute nucleosome enrichment of a Vmat
Usage

computeNucleosomeEnrichmentOverBackground(
  Vmat,
  background = NULL,
  plus1_nuc_only = FALSE,
  minus1_nuc = list(c(xmin = -150, xmax = -70), c(ymin = 165, ymax = 260)),
  minus1_nuc_neg = list(c(xmin = -150, xmax = -70), c(ymin = 60, ymax = 145)),
  plus1_nuc = list(c(xmin = 70, xmax = 150), c(ymin = 165, ymax = 260)),
  plus1_nuc_neg = list(c(xmin = 70, xmax = 150), c(ymin = 50, ymax = 145)),
  ...
)

Arguments

Vmat        A Vmat computed by nucleosomeEnrichment function
background  a background Vmat
plus1_nuc_only Boolean Should compute nucleosome enrichment only for +1 nucleosome?
minus1_nuc  list where the -1 nucleosome is located
minus1_nuc_neg where the background of the -1 nucleosome is located
plus1_nuc   where the +1 nucleosome is located
plus1_nuc_neg where the background of the +1 nucleosome is located
...         additional parameters

Value

list

computeVmat A function to compute Vplot matrix

Description

This function computes the underlying matrix shown as a heatmap in Vplots. For each pair of coordinates (x: distance from fragment midpoint to center of GRanges of interest; y: fragment size), the function computes how many fragments there are.

Usage

computeVmat(
  bam_granges,
  granges,
  cores = 1,
  xlims = c(-250, 250),
  ylims = c(50, 300)
)
Arguments

- **bam_granges**: GRanges, paired-end fragments
- **granges**: GRanges, regions to map the fragments onto
- **cores**: Integer, nb of threads to parallelize fragments subsetting
- **xlims**: The x limits of the computed Vmat
- **ylims**: The y limits of the computed Vmat

Value

A table object

Examples

```r
data(bam_test)
data(ce11_all_REs)
Vmat <- computeVmat(bam_test, ce11_all_REs)
dim(Vmat)
Vmat[seq(1,5), seq(1,10)]
```

Description

high-score CTCF binding motifs, obtained from JASPAR

Usage

```r
data(CTCF_hg38)
```

Format

An object of class "GRanges".

Examples

```r
data(CTCF_hg38)
CTCF_hg38
```
deconvolveBidirectionalPromoters

A function to duplicate bi-directional GRanges

Description

This function splits bi-directional ranges into + and - stranded ranges. It duplicates the ranges which are '*'.

Usage

deconvolveBidirectionalPromoters(granges)

Arguments

granges A stranded GRanges object

Value

GRanges with only '+' and '-' strands. GRanges with '*' strand have been duplicated and split into forward and reverse strands.

Examples

data(cell_all_REs)
library(GenomicRanges)
proms <- cell_all_REs[grepl('prom', cell_all_REs$regulatory_class)]
proms
table(strand(proms))
proms <- deconvolveBidirectionalPromoters(proms)
proms
table(strand(proms))

getCuts

Internal function

Description

Function to extract cuts (i.e. extremities) of fragments stored as GRanges.

Usage

getcuts(gr)

Arguments

gr GRanges Paired-end fragments used to extract their extremities
getFragmentsDistribution

A function to compute sizes distribution for paired-end fragments

Description

This function takes fragments and compute the distribution of their sizes over a set or multiple sets of GRanges.

Usage

getFragmentsDistribution(
  fragments,
  granges_list = NULL,
  extend_granges = c(-500, 500),
  limits = c(0, 600),
  roll = 3,
  cores = 1
)

Arguments

fragments    GRanges object containing paired-end fragments. See importPEBamFiles for more details on how to create such object.
granges_list GRanges, can be a list of different sets of GRanges.extend_granges numeric vector of length 2, how the GRanges should be extended.limits numeric vector of length 2, only consider fragments within this window of sizes.roll Integer, apply a moving average of this sizecores Integer, number of threads used to compute fragment size distribution

Value

A list of tbl, one for each .bam file.

Examples

data(bam_test)
data(cell_proms)
df <- getFragmentsDistribution(
  bam_test,
  cell_proms,
  extend_granges = c(-500, 500)
)
head(df)
which.max(df$y)
importPEBamFiles

A function to import paired end bam files as GRanges

Description
This function takes bam file paths and read them into GRanges objects. Note: Can be quite lengthy for .bam files with 5+ millions fragments.

Usage
importPEBamFiles(
  files,
  genome = NULL,
  where = NULL,
  max_insert_size = 1000,
  shift_ATAC_fragments = FALSE,
  cores = 10,
  verbose = TRUE
)

Arguments
  files character vector, each element of the vector is the path of an individual .bam file.
  genome character, genome ID (e.g. "sacCer3", "ce11", "dm6", "mm10" or "hg38").
  where GRanges, only import the fragments mapping to the input GRanges (can fasten the import process a lot).
  max_insert_size Integer, filter out fragments larger than this size.
  shift_ATAC_fragments Boolean, if the fragments come from ATAC-seq, one might want to shift the extremities by +5/-4 bp.
  cores Integer, number of cores to use when indexing bam files
  verbose Boolean

Value
A GRanges object containing fragments from the input .bam file.

Examples
bamfile <- system.file("extdata", "ex1.bam", package = "Rsamtools")
fragments <- importPEBamFiles(
  bamfile,
  shift_ATAC_fragments = TRUE
)
fragments
MNase_sacCer3_Henikoff2011

Description
A sample of MNase-seq fragments from yeast (Henikoff et al. 2011, "Epigenome characterization at single base-pair resolution", PNAS)

Usage
data(MNase_sacCer3_Henikoff2011)

Format
An object of class "GRanges".

Examples
data(MNase_sacCer3_Henikoff2011)
MNase_sacCer3_Henikoff2011

MNase_sacCer3_Henikoff2011_subset

Description
A sample of fragments from multiple MNase-seq experiments performed in yeast (Henikoff et al. 2011, "Epigenome characterization at single base-pair resolution", PNAS), mapping over chrXV:186,400-187,400.

Usage
data(MNase_sacCer3_Henikoff2011_subset)

Format
An object of class "GRanges".

Examples
data(MNase_sacCer3_Henikoff2011_subset)
MNase_sacCer3_Henikoff2011_subset
normalizeVmat

A function to normalize a Vmat

Description

This function normalizes a Vmat. Several different approaches have been implemented to normalize the Vmats.

Usage

normalizeVmat(
  Vmat,
  bam_granges,
  granges,
  normFun = c("zscore"),
  s = 0.99,
  roll = 1,
  verbose = TRUE
)

Arguments

Vmat
A Vmat, usually output of computeVmat

bam_granges
GRanges, the paired-end fragments

granges
GRanges, the regions to map the fragments onto

normFun
character. A Vmat should be scaled either by:
  • 'libdepth+nloci', e.g. the library depth and the number of loci used to compute the Vmat;
  • zscore, if relative patterns of fragment density are more important than density per se;
  • Alternatively, the Vmat can be scaled to a chosen quantile ('quantile') or to the max Vmat value ('max').

s
A float indicating which quantile to use if 'quantile' normalization is chosen

roll
integer, to use as the window to smooth the Vmat rows by rolling mean.

verbose
Boolean

Value

A normalized Vmat object
Examples

```r
data(bam_test)
data(ce11_all_REs)
Vmat <- computeVmat(bam_test, ce11_all_REs)
Vmat <- normalizeVmat(  
  Vmat,
  bam_test,
  ce11_all_REs,
  normFun = c('libdepth+nloci')
)
```

---

nucleosomeEnrichment  A function to compute nucleosome enrichment over a set of GRanges

Description

A function to compute nucleosome enrichment over a set of GRanges

Usage

```r
nucleosomeEnrichment(x, ...)
```

Arguments

- `x` a GRanges or Vmat
- `...` additional parameters

Value

list

Examples

```r
data(bam_test)
data(ce11_proms)
n <- nucleosomeEnrichment(bam_test, ce11_proms)
n$fisher_test
n$plot
nucleosomeEnrichment.GRanges

A function to compute nucleosome enrichment over a set of GRanges

Description

A function to compute nucleosome enrichment over a set of GRanges

Usage

## S3 method for class 'GRanges'
nucleosomeEnrichment(x, granges, plus1_nuc_only = FALSE, verbose = TRUE, ...)

Arguments

- **x**: GRanges, paired-end fragments
- **granges**: GRanges, loci to map the fragments onto
- **plus1_nuc_only**: Boolean, should compute nucleosome enrichment only for +1 nucleosome?
- **verbose**: Boolean
- **...**: additional parameters

Value

list

Examples

data(bam_test)
data(ce11_proms)
n <- nucleosomeEnrichment(bam_test, ce11_proms)
n$fisher_test
n$plot

nucleosomeEnrichment.Vmat

A function to compute nucleosome enrichment over a Vmat

Description

A function to compute nucleosome enrichment over a Vmat

Usage

## S3 method for class 'Vmat'
nucleosomeEnrichment(x, background, plus1_nuc_only = FALSE, ...)

Arguments

- **x**: Vmat
- **background**: Vmat
- **plus1_nuc_only**: Boolean, should compute nucleosome enrichment only for +1 nucleosome?
- **...**: additional parameters

Value

list

Examples

data(bam_test)
data(ce11_proms)
n <- nucleosomeEnrichment(bam_test, ce11_proms)
n$fisher_test
n$plot
plotFootprint

Arguments

- **x**: a computed Vmat. Should be un-normalized.
- **background**: a background Vmat. Should be un-normalized.
- **plus1_nuc_only**: Boolean, should compute nucleosome enrichment only for +1 nucleosome?
- **...**: additional parameters

Value

list

Examples

```r
data(bam_test)
data(ce11_proms)
V <- plotVmat(
  bam_test,
  ce11_proms,
  normFun = '',
  return_Vmat = TRUE
)
V_bg <- plotVmat(
  bam_test,
  sampleGRanges(ce11_proms),
  normFun = '',
  return_Vmat = TRUE
)
n <- nucleosomeEnrichment(V, V_bg)
n$fisher_test
n$plot
```

plotFootprint

Description

This function takes paired-end fragments, extract the "cuts" (i.e. extremities) and plot the footprint profile over a set of GRanges.

Usage

```r
plotFootprint(
  frags,
  targets,
  split_strand = FALSE,
  plot_central = TRUE,
  xlim = c(-75, 75),
  bin = 1,
  verbose = 1
)
```
plotProfile

Arguments

- **frags**: GRanges, the paired-end fragments
- **targets**: GRanges, the loci to map the fragments onto
- **split_strand**: Boolean, should the + and - strand be splitted?
- **plot_central**: plot grey rectangle over the loci
- **xlim**: numeric vector of length 2, the x limits of the computed Vmat
- **bin**: Integer, bin used to smooth the footprint profile
- **verbose**: Integer

Value

A footprint ggplot

Examples

```r
data(bam_test)
data(ce11_proms)
plotFootprint(bam_test, ce11_proms)
```

---

**plotProfile**

A function to generate a Vplot along chromosome coordinates

Description

The paired-end fragments overlapping a locus of interest (e.g., binding sites, provided in the 'loci' argument) are shown in red while the remaining fragments mapping to the genomic window are displayed in black. Marginal curves are also plotted on the side of the distribution plot. They highlight the smoothed distribution of the position of paired-end fragment midpoints (top) or of the paired-end fragment length (right)

Usage

```r
plotProfile(
  fragments,
  window = loc,
  loci = NULL,
  annots = NULL,
  min = 50,
  max = 200,
  alpha = 0.5,
  size = 1,
  with_densities = TRUE,
  verbose = TRUE
)
```
plotVmat

A function to generate a Vplot

Description

See individual methods for further detail

Usage

plotVmat(x, ...)

Arguments

x  GRanges or list or Vmat

...  additional parameters
plotVmat.default

Value

A Vmat ggplot

Examples

```r
data(bam_test)
data(ce11_proms)
V <- plotVmat(
  bam_test,
  ce11_proms,
  normFun = 'libdepth+nloci'
)
```

Description

The default plotVmat method generates a ggplot representing a heatmap of fragment density.

Usage

```r
## Default S3 method:
plotVmat(
  x,
  hm = 90,
  colors = COLORSCALE_VMAT,
  breaks = NULL,
  xlim = c(-250, 250),
  ylim = c(50, 300),
  main = "",
  xlab = "Distance from center of elements",
  ylab = "Fragment length",
  key = "Score",
  ...
)
```

Arguments

- `x` A computed Vmat (ideally, should be normalized)
- `hm` Integer, should be between 0 and 100. Used to automatically scale the range of colors (best to keep between 90 and 100)
- `colors` a vector of colors
- `breaks` a vector of breaks. length(breaks) == length(colors) + 1
- `xlim` vector of two integers, x limits
- `ylim` vector of two integers, y limits
main character, title of the plot
xlab character, x-axis label
ylab character, y-axis label
key character, legend label
... additional parameters

Value
A Vmat ggplot

Examples

```r
data(bam_test)
data(cell_proms)
V <- plotVmat(
  bam_test,
  cell_proms,
  normFun = 'libdepth+nloci',
  return_Vmat = TRUE
)
plotVmat(V)
```

Description
The `plotVmat.GRanges()` method computes and normalizes a Vmat before passing it to `plotVmat.Vmat()` method.

Usage
```r
## S3 method for class 'GRanges'
plotVmat(
  x,
  granges,
  xlims = c(-250, 250),
  ylims = c(50, 300),
  normFun = "",
  s = 0.95,
  roll = 3,
  cores = 1,
  return_Vmat = FALSE,
  verbose = 1,
  ...
)
```
### Arguments

- **x**: GRanges, paired-end fragments
- **granges**: GRanges, loci to map the fragments onto
- **xlims**: x limits of the computed Vmat
- **ylims**: y limits of the computed Vmat
- **normFun**: character. A Vmat should be scaled either by:
  - 'libdepth+nloci', e.g. the library depth and the number of loci used to compute the Vmat;
  - zscore, if relative patterns of fragment density are more important than density per se;
  - Alternatively, the Vmat can be scaled to a chosen quantile ('quantile') or to the max Vmat value ('max').
- **s**: A float indicating which quantile to use if 'quantile' normalization is chosen
- **roll**: integer, to use as the window to smooth the Vmat rows by rolling mean.
- **cores**: Integer, number of threads to parallelize fragments subsetting
- **return_Vmat**: Boolean, should the function return the computed Vmat rather than the plot?
- **verbose**: Boolean
  - ... additional parameters

### Value

A Vmat ggplot

### Examples

```r
data(bam_test)
data(ce11_proms)
V <- plotVmat(
  bam_test,
  ce11_proms,
  normFun = 'libdepth+nloci',
  roll = 5
)
```

---

**plotVmat.list**

*A function to compute (and plot) several Vmats.*

---

### Description

The `plotVmat.GRanges()` method computes and normalizes multiple Vmats before passing them to `plotVmat.VmatList()` method.
### Usage
```r
## S3 method for class 'list'
plotVmat(
  x,
  cores = 1,
  cores_subsetting = 1,
  nrow = NULL,
  ncol = NULL,
  xlims = c(-250, 250),
  ylims = c(50, 300),
  normFun = "libdepth+nloci",
  s = 0.95,
  roll = 3,
  return_Vmat = FALSE,
  verbose = 1,
  ...
)
```  

### Arguments
- **x**: list Each element of the list should be a list containing paired-end fragments and GRanges of interest.
- **cores**: Integer, number of cores to parallelize the plots
- **cores_subsetting**: Integer, number of threads to parallelize fragments subsetting
- **nrow**: Integer, how many rows in facet?
- **ncol**: Integer, how many cols in facet?
- **xlims**: x limits of the computed Vmat
- **ylims**: y limits of the computed Vmat
- **normFun**: character. A Vmat should be scaled either by:
  - "libdepth+nloci", e.g. the library depth and the number of loci used to compute the Vmat;
  - zscore, if relative patterns of fragment density are more important than density per se;
  - Alternatively, the Vmat can be scaled to a chosen quantile ('quantile') or to the max Vmat value ('max').
- **s**: A float indicating which quantile to use if 'quantile' normalization is chosen
- **roll**: integer, to use as the window to smooth the Vmat rows by rolling mean.
- **return_Vmat**: Boolean, should the function return the computed Vmat rather than the plot?
- **verbose**: Boolean
- **...**: additional parameters

### Value
A list of Vmat ggplots
Examples

data(bam_test)
data(cell_proms)
list_params <- list(
  'germline' = list(
    bam_test,
    cell_proms[cell_proms$which.tissues == 'Germline']
  ),
  'muscle' = list(
    bam_test,
    cell_proms[cell_proms$which.tissues == 'Muscle']
  )
)
V <- plotVmat(
  list_params,
  normFun = 'libdepth+nloci',
  roll = 5
)

plotVmat.Vmat A function to plot a computed Vmat

Description

The plotVmat.Vmat() method forwards the Vmat to plotVmat.default().

Usage

## S3 method for class 'Vmat'
plotVmat(x, ...)

Arguments

  x A computed Vmat (ideally, should be normalized)

  ... additional parameters

Value

A Vmat ggplot

Examples

data(bam_test)
data(cell_proms)
V <- plotVmat(
  bam_test,
  cell_proms,
  normFun = 'libdepth+nloci',
  return_Vmat = TRUE
plotVmat.VmatList

A function to plot a computed VmatList

Description

The plotVmat.VmatList() method forwards the Vmat to plotVmat.default().

Usage

## S3 method for class 'VmatList'
plotVmat(x, nrow = NULL, ncol = NULL, dir = "v", ...)

Arguments

x
A VmatList (output of plotVmat.list())

nrow
Integer, how many rows in facet?

ncol
Integer, how many cols in facet?

dir
str, direction of facets?

...
additional parameters

Value

A Vmat ggplot

Examples

data(bam_test)
data(cell_proms)
list_params <- list(
  'germline' = list(
    bam_test,
    cell_proms[cell_proms$which.tissues == 'Germline']
  ),
  'muscle' = list(
    bam_test,
    cell_proms[cell_proms$which.tissues == 'Muscle']
  )
)
V <- plotVmat(
  list_params,
  normFun = 'libdepth+nloci',
  roll = 5
)
Description

Genomic loci with a REB1 binding motifs according to http://jaspar.genereg.net/api/v1/matrix/MA0363.1.jaspar. PWM and scanning done with TFBSTools.

Usage

data(REB1_sacCer3)

Format

An object of class "GRanges".

References

Rossi, Lai & Pugh 2018 Genome Research

Examples

data(REB1_sacCer3)
REB1_sacCer3

---

sampleGRanges

A function to sample GRanges from GRanges

Description

This function takes a given GRanges and returns another GRanges object. The new GRanges has the same number of ranges and the same chromosome, width and strand distributions than the original GRanges.

Usage

sampleGRanges(
  x,
  n = NULL,
  width = NULL,
  exclude = FALSE,
  avoid_overlap = FALSE
)


Arguments

- **x**: GRanges object
- **n**: Integer, number of sampled GRanges
- **width**: Integer, width of sampled GRanges
- **exclude**: Boolean, should the original GRanges be excluded?
- **avoid_overlap**: Boolean, should the sampled GRanges not be overlapping?

Value

A GRanges object of length n

Examples

```r
data(cell_proms)
sampleGRanges(cell_proms, 100)
```

Description

This function takes a given GRanges and returns another GRanges object. The new GRanges has the same number of ranges and the same chromosome, width and strand distributions than the original GRanges.

Usage

```r
## S3 method for class 'GRanges'
sampleGRanges(
  x,
  n = NULL,
  width = NULL,
  exclude = FALSE,
  avoid_overlap = FALSE
)
```

Arguments

- **x**: GRanges object
- **n**: Integer, number of sampled GRanges
- **width**: Integer, width of sampled GRanges
- **exclude**: Boolean, should the original GRanges be excluded?
- **avoid_overlap**: Boolean, should the sampled GRanges not be overlapping?
shiftATACGranges

Value
A GRanges object of length n

Examples

data(cell_proms)
sampleGRanges(cell_proms, 100)

shiftATACGranges  
A function to shift GRanges fragments by 5/-4. This is useful when dealing with fragments coming from ATAC-seq.

Description
A function to shift GRanges fragments by 5/-4. This is useful when dealing with fragments coming from ATAC-seq.

Usage
shiftATACGranges(g, pos_shift = 4, neg_shift = 5)

Arguments

  g       GRanges of ATAC-seq fragments
  pos_shift  Integer. How many bases should fragments on direct strand be shifted by?
  neg_shift  Integer. How many bases should fragments on negative strand be shifted by?

Value
A GRanges object containing fragments from the input .bam file.

Examples

data(bam_test)
shiftATACGranges(bam_test)
shuffleVmat  
*A function to shuffle a Vmat*

**Description**

This function works on a Vmat (the output of computeVmat()). It shuffles the matrix to randomize the fragment densities.

**Usage**

```r
shuffleVmat(Vmat)
```

**Arguments**

- `Vmat`  
  A Vmat, usually output of computeVmat

**Value**

A shuffled Vmat object

**Examples**

```r
data(bam_test)
data(ce11_all_REs)
Vmat <- computeVmat(bam_test, ce11_all_REs)
Vmat <- shuffleVmat(Vmat)
```

---

**theme_ggplot2**

*Personal ggplot2 theming function, adapted from roboto-condensed at https://github.com/hrbrmstr/hrbrthemes/*

**Description**

Personal ggplot2 theming function, adapted from roboto-condensed at https://github.com/hrbrmstr/hrbrthemes/

**Usage**

```r
theme_ggplot2(
  grid = TRUE,
  border = TRUE,
  base_family = "",
  base_size = 8,
  plot_title_family = base_family,
  plot_title_size = 12,
  plot_title_face = "plain",
  plot_title_margin = 5,
)```
theme_ggplot2

subtitle_size = 11,
subtitle_face = "plain",
subtitle_margin = 5,
strip_text_family = base_family,
strip_text_size = 10,
strip_text_face = "bold",
caption_size = 9,
caption_face = "plain",
caption_margin = 3,
axis_text_size = base_size,
axis_title_family = base_family,
axis_title_size = 9,
axis_title_face = "plain",
axis_title_just = "rt",
panel_spacing = grid::unit(2, "lines"),
grid_col = "#cccccc",
plot_margin = margin(12, 12, 12, 12),
axis_col = "#cccccc",
axis = FALSE,
ticks = FALSE
)

Arguments

grid panel grid (‘TRUE’, ‘FALSE’, or a combination of ‘X’, ‘x’, ‘Y’, ‘y’)
border border if ‘TRUE’ add border
base_family, base_size base font family and size
plot_title_family, plot_title_face plot title family, face
plot_title_size, plot_title_margin plot title size and margin
subtitle_face, subtitle_size plot subtitle family, face and size
subtitle_margin plot subtitle margin bottom (single numeric value)
strip_text_family, strip_text_face, strip_text_size facet label font family, face and size
caption_face, caption_size, caption_margin plot caption family, face, size and margin
axis_text_size font size of axis text
axis_title_family, axis_title_face, axis_title_size axis title font family, face and size
axis_title_just axis title font justificationk one of ‘[blmcrt]’
panel_spacing panel spacing (use ‘unit()’)
grid_col  grid color
plot_margin  plot margin (specify with ggplot2::margin)
axis_col  axis color
axis  add x or y axes? ‘TRUE’, ‘FALSE’, “xy”
ticks  ticks if ‘TRUE’ add ticks

Value
theme A ggplot theme

Examples
library(ggplot2)

ggplot(mtcars, aes(mpg, wt)) +
  geom_point() +
  labs(x="Fuel efficiency (mpg)",
       y="Weight (tons)",
       title="Seminal ggplot2 scatterplot example") +
  theme_ggplot2()
Index

* datasets
  - ABF1_sacCer3, 3
  - ATAC_ce11_Serizay2020, 4
  - bam_test, 4
  - cell1_all_REs, 5
  - cell1_proms, 6
  - CTCF_hg38, 8
  - MNase_sacCer3_Henikoff2011, 12
  - MNase_sacCer3_Henikoff2011_subset, 12
  - REB1_sacCer3, 25

* internal
  - computeNucleosomeEnrichmentOverBackground, 6
  - getCuts, 9
  - ABF1_sacCer3, 3
  - alignToTSS, 3
  - ATAC_ce11_Serizay2020, 4
  - bam_test, 4
  - cell1_all_REs, 5
  - cell1_proms, 6
  - computeNucleosomeEnrichmentOverBackground, 6
  - computeVmat, 7
  - CTCF_hg38, 8
  - deconvolveBidirectionalPromoters, 9
  - getCuts, 9
  - getFragmentsDistribution, 10
  - importPEBamFiles, 11
  - MNase_sacCer3_Henikoff2011, 12
  - MNase_sacCer3_Henikoff2011_subset, 12
  - normalizeVmat, 13
  - nucleosomeEnrichment, 14
  - nucleosomeEnrichment.GRanges, 15
  - nucleosomeEnrichment.Vmat, 15
  - plotFootprint, 16
  - plotProfile, 17
  - plotVmat, 18
  - plotVmat.default, 19
  - plotVmat.GRanges, 20
  - plotVmat.list, 21
  - plotVmat.Vmat, 23
  - plotVmat.VmatList, 24
  - REB1_sacCer3, 25
  - sampleGRanges, 25
  - sampleGRanges.GRanges, 26
  - shiftATACGranges, 27
  - shuffleVmat, 28
  - theme_ggplot2, 28