Package 'metagene'

October 18, 2017

Version 2.8.0

Date 2014-05-05

Title A package to produce metagene plots

Author Charles Joly Beauparlant <charles.joly-beauparlant@crchul.ulaval.ca>, Fabien Claude Lamaze <fabien.lamaze.1@ulaval.ca>, Rawane Samb <rawane.samb.1@ulaval.ca>, Astrid Louise Deschenes <Astrid-Louise.Deschenes@crchudequebec.ulaval.ca> and Arnaud Droit <arnaud.droit@crchuq.ulaval.ca>.

Author@R c(person(``Charles", ``Joly Beauparlant", email=``charles.joly-beauparlant@crchul.ulaval.ca"), person(``Fabien Claude", ``Lamaze", email=``fabien.lamaze.1@ulaval.ca"), person(``Rawane", ``Samb", email=``rawane.samb.1@ulaval.ca"), person(``Astrid Louise",``Deschenes", email=``Astrid-Louise.Deschenes@crchudequebec.ulaval.ca"), person(``Arnaud", ``Droit", email=``arnaud.droit@crchuq.ulaval.ca"))

Maintainer Charles Joly Beauparlant <charles.joly-beauparlant@crchul.ulaval.ca>

Description This package produces metagene plots to compare the behavior of DNA-interacting proteins at selected groups of genes/features. Bam files are used to increase the resolution. Multiple combination of group of bam files and/or group of genomic regions can be compared in a single analysis. Bootstraping analysis is used to compare the groups and locate regions with statistically different enrichment profiles.

biocViews ChIPSeq, Genetics, MultipleComparison, Coverage, Alignment, Sequencing

License Artistic-2.0 | file LICENSE

LazyData true

BugReports https://github.com/CharlesJB/metagene/issues

VignetteBuilder knitr

Depends R (>= 3.2.0), R6 (>= 2.0), GenomicRanges, BiocParallel

Imports rtracklayer, gplots, tools, GenomicAlignments, GenomeInfoDb, GenomicFeatures, IRanges, ggplot2, muStat, Rsamtools, DBChIP, matrixStats

Suggests RUnit, BiocGenerics, knitr, BiocStyle, rmarkdown, similaRpeak

RoxygenNote 5.0.1

NeedsCompilation no

R topics documented:

`Bam_Handler` *A class to manage BAM files.*

Description

This class will allow to load, convert and normalize alignments and regions files/data.

Usage

```
Bam_Handler
```

Format

A BAM manager

Value

`Bam_Handler$new` returns a `Bam_Handler` object which contains coverage related information for every BAM files.

Constructor

```
bh <- Bam_Handler$new(bam_files, cores = SerialParam())
```

bam_files A `vector` of BAM filenames. The BAM files must be indexed. i.e.: if a file is named file.bam, there must be a file named file.bam.bai in the same directory.

cores The number of cores available to parallelize the analysis. Either a positive integer or a `BiocParallelParam`. Default: `SerialParam()`.

`Bam_Handler$new` returns a `Bam_Handler` object that contains and manages BAM files. Coverage related information as alignment count can be obtain by using this object.

Methods

`bh$get_aligned_count(bam_file)`

bam_file The name of the BAM file.

`bg$get_bam_name(bam_file)`

bam_file The name of the BAM file.

`bh$get_rpm_coefficient(bam_file)`

bam_file The name of the BAM file.

`bh$index_bam_files(bam_files)`

bam_files A `vector` of BAM filenames.

`bh$get_bam_files()`

`bh$get_coverage(bam_file, regions, force_seqlevels = FALSE)`

bam_file The name of the BAM file.

regions A not empty `GRanges` object.

force_seqlevels If `TRUE`, Remove regions that are not found in bam file header. Default: `FALSE`.

`bh$get_normalized_coverage(bam_file, regions, force_seqlevels = FALSE)`

bam_file The name of the BAM file.

regions A not empty `GRanges` object.

force_seqlevels If `TRUE`, Remove regions that are not found in bam file header. Default: `FALSE`.

`bh$get_noise_ratio(chip_bam_file, input_bam_file)`

chip_bam_file The path to the chip bam file.

input_bam_file The path to the input (control) bam file.

Examples

```
bam_file <- get_demo_bam_files()[1]
bh <- metagene:::Bam_Handler$new(bam_files = bam_file)
bh$get_aligned_count(bam_file)
```

`get_demo_bam_files` *Get BAM filenames for demo*

Description

Get BAM filenames for demo

Usage

```
get_demo_bam_files()
```

Value

A vector of BAM filenames

Examples

```
bam_files <- get_demo_bam_files()
```

`get_demo_design` *Get a demo design object*

Description

Get a demo design object

Usage

```
get_demo_design()
```

Value

A `data.frame` corresponding to a valid design.

Examples

```
mg <- get_demo_design()
```

`get_demo_metagene` *Get a demo metagene object*

Description

Get a demo metagene object

Usage

```
get_demo_metagene()
```

Value

A metagene object

Examples

```
mg <- get_demo_metagene()
```

get_demo_regions *Get regions filenames for demo*

Description

Get regions filenames for demo

Usage

```
get_demo_regions()
```

Value

A vector of regions filenames

Examples

```
regions <- get_demo_regions()
```

get_promoters_txdb *Extract Entrez genes promoters from TxDb object.*

Description

Extract Entrez genes promoters from TxDb object.

Usage

```
get_promoters_txdb(txdb, upstream = 1000, downstream = 1000)
```

Arguments

txdb	A valid TxDb object.
upstream	The number of nucleotides upstream of TSS.
downstream	The number of nucleotides downstream of TSS.

Value

A GRanges object that contains the promoters infos.

Examples

```
## Not run:
    # require(TxDb.Hsapiens.UCSC.hg19.knownGene)
    txdb <- TxDb.Hsapiens.UCSC.hg19.knownGene
    promoters_hg19 <- get_promoters_txdb(txdb)

## End(Not run)
```

metagene *A class to manage metagene analysis.*

Description

This class will allow to load, convert and normalize alignments and regions files/data. Once the data is ready, the user can then chose to produce metagene plots on the data (or a subset of the data).

Usage

```
metagene
```

Format

A metagene experiment manager

Value

`metagene$new` returns a `metagene` object which contains the normalized coverage values for every regions and for every BAM files.

Constructor

```
mg <- metagene$new(regions, bam_files, padding_size = 0,                    cores = SerialPar
```

regions Either a `vector` of BED, narrowPeak or broadPeak filenames, a `GRanges` object or a `GRangesList` object.

bam_files A `vector` of BAM filenames. The BAM files must be indexed. i.e.: if a file is named file.bam, there must be a file named file.bam.bai in the same directory.

padding_size The regions will be extended on each side by the value of this parameter. The padding_size must be a non-negative integer. Default = 0.

cores The number of cores available to parallelize the analysis. Either a positive integer or a `BiocParallelParam`. Default: `SerialParam()`.

verbose Print progression of the analysis. A logical constant. Default: `FALSE`.

force_seqlevels If `TRUE`, Remove regions that are not found in bam file header. Default: `FALSE`.

`metagene$new` returns a `metagene` object that contains the coverages for every BAM files in the regions from the `regions` param.

Methods

```
mg$plot(region_names = NULL, exp_names = NULL,              title = NULL, x_label = NULL)
```

region_names The names of the regions to extract. If `NULL`, all the regions are returned. Default: `NULL`.

exp_names The names of the experiments to extract. If a design was added to the `metagene` object, `exp_names` correspond to the column names in the design, otherwise `exp_names` corresponds to the BAM name or the BAM filename. If `NULL`, all the experiments are returned. Default: `NULL`.

title A title to add to the graph. If `NULL`, will be automatically created. Default: NULL

x_label X-axis label to add to the metagene plot. If NULL, metagene will use generic label. Default: NULL.

```
mg$produce_matrices(design, bin_count, noise_removal,                normalization, flip_regions,
```

design A data.frame that describe to experiment to plot. see plot function for more details. NA can be used keep previous design value. Default: NA.

bin_count The number of bin to create. NA can be used to keep previous bin_count value. A bin_count value of 100 will be used if no value is specified. Default: NA.

noise_removal The algorithm to use to remove control(s). Possible values are NA, NULL or "NCIS". By default, value is NULL. Use NA keep previous noise_removal value (i.e. if produce_matrices was called before). See Liand and Keles 2012 for the NCIS algorithm.

normalization The algorithm to use to normalize samples. Possible default, value is NULL and no normalization will be performed. Use NA keep previous normalization value (i.e. if produce_matrices was called before).

flip_regions Should regions on negative strand be flip_regions? Default: FALSE.

bin_size Deprecated.

```
mg$produce_data_frame(stat = "bootstrap", range = NULL                    ...)
```

stat The stat to use to calculate the values of the ribbon in the metagene plot. Must be "bootstrap" or "basic". "bootstrap" will estimate the range of average ("mean" or "median") that could have produce the observed distribution of values for each bin. With the "basic" approach, the ribbon will represent the range of the values between 1 - alpha / 2 and alpha / 2 (see ... param.

range Deprecated.

... Extra params for the calculation of the ribbon values. See following param descriptions.

alpha The range of the estimation to be shown with the ribbon. 1 - alpha / 2 and alpha / 2 will be used. Default: 0.05.

average The function to use to summarize the values of each bins. "mean" or "median". Default: "mean".

sample_count With "bootstrap" only. The number of draw to do in the bootstrap calculation. Default: 1000.

```
mg$get_params()
```

```
mg$get_design()
```

```
mg$get_regions(region_names = NULL)
```

region_names The names of the regions to extract. If NULL, all the regions are returned. Default: NULL.

```
mg$get_matrices(region_names = NULL, exp_name = NULL)
```

region_names The names of the regions to extract. If NULL, all the regions are returned. Default: NULL.

exp_names The names of the experiments to extract. If a design was added to the metagene object, exp_names correspond to the column names in the design, otherwise exp_names corresponds to the BAM name or the BAM filename. If NULL, all the experiments are returned. Default: NULL.

```
mg$get_data_frame(region_names = NULL, exp_name = NULL)
```

region_names The names of the regions to extract. If NULL, all the regions are returned. Default: NULL.

exp_names The names of the experiments to extract. If a design was added to the metagene object, exp_names correspond to the column names in the design, otherwise exp_names corresponds to the BAM name or the BAM filename. If NULL, all the experiments are returned. Default: NULL.

get_plot = function()

get_raw_coverages = function(filenames)

filenames The name of the file to extract raw coverages. Can be the filename with the extension of the name of the bam file (if a named bam files was used during the creation of the metagene object). If NULL, returns the coverage of every bam files. Default: NULL.

get_normalized_coverages = function(filenames)

filenames The name of the file to extract normalized coverages (in RPM). Can be the filename with the extension of the name of the bam file (if a named bam files was used during the creation of the metagene object). If NULL, returns the coverage every bam files. Default: NULL.

```
mg$export(bam_file, region, file)
```

bam_file The name of the bam file to export.

region The name of the region to export.

file The name of the ouput file.

```
mg$add_design(design = NULL, check_bam_files = FALSE)
```

design A data.frame that describe to experiment to plot. See plot function for more details. NA can be used keep previous design value. Default: NA.

check_bam_files Force check that all the bam files from the first columns of the design are present in current metagene object. Default: FALSE

```
mg$unflip_regions()
```

```
mg$flip_regions()
```

```
mg$unflip_regions()
```

Examples

```
region <- get_demo_regions()[1]
bam_file <- get_demo_bam_files()[1]
mg <- metagene$new(regions = region, bam_files = bam_file)
## Not run:
    df <- metagene$plot()

## End(Not run)
```

`permutation_test` *Perform a permutation test on 2 matrices*

Description

The goal of this function is to calculate the values of a test performed by `FUN` after each of `sample_count` permutations.

Usage

```
permutation_test(matrix1, matrix2, sample_size, sample_count, FUN, ...)
```

Arguments

`matrix1`	The first matrix.
`matrix2`	The second matrix.
`sample_size`	The number of element to draw for each matrix.
`sample_count`	The number of permutations.
`FUN`	The function to use to compare the 2 matrices. First two params must be `numeric` vector and the return must be a single `numeric` value.
`...`	Extra param for `FUN`.

Details

Each round of the permutation test, two new matrices will be randomly sampled from using the combination of the two original matrices. The means of each columns will be calculated to produce the vectors that will be sent `FUN`.

Value

A `vector` of numeric corresponding to the result of each permutation.

Examples

```
# Get some matrices
mg <- get_demo_metagene()
mg$produce_matrices()
m1 <- mg$get_matrices()$list1$align1_rep1$input
m2 <- mg$get_matrices()$list1$align2_rep1$input

# Perform permutation test
sample_size <- min(nrow(m1), nrow(m2))
FUN = function(a, b) { mean(a) - mean(b) } # Dummy function for demo purpose
# A sample_count >= 1000 should be used in a real analysis
permutation_results <- permutation_test(m1, m2, sample_size = sample_size,
                                        sample_count = 10, FUN = FUN)
```

plot_metagene *Produce a metagene plot*

Description

Produce a metagene plot

Usage

```
plot_metagene(df)
```

Arguments

df	a data.frame obtained with the get_data_frame function. Must have the following columns: "group", "position", "value", "qinf" and "qsup".

Value

A ‘ggplot‘ object.

Examples

```
region <- get_demo_regions()[1]
bam_file <- get_demo_bam_files()[1]
mg <- metagene$new(regions = region, bam_files = bam_file)
mg$produce_data_frame()
df <- mg$get_data_frame()
p <- plot_metagene(df)
```

promoters_hg18 *Promoters regions of hg18 Entrez genes.*

Description

Each regions have a width of 2000 nucleotide centered at the transcription start site.

Usage

```
promoters_hg18
```

Format

A GRanges object with 19742 ranges.

Value

A GRanges.

See Also

get_promoters_txdb

Examples

```
data(promoters_hg18)
```

`promoters_hg19` *Promoters regions of hg19 Entrez genes.*

Description

Each regions have a width of 2000 nucleotide centered at the transcription start site.

Usage

```
promoters_hg19
```

Format

A `GRanges` object with 23056 ranges.

Value

A `GRanges`.

See Also

`get_promoters_txdb`

Examples

```
data(promoters_hg19)
```

`promoters_mm10` *Promoters regions of mm10 Entrez genes.*

Description

Each regions have a width of 2000 nucleotide centered at the transcription start site.

Usage

```
promoters_mm10
```

Format

A `GRanges` object with 23653 ranges.

Value

A `GRanges`.

See Also

get_promoters_txdb

Examples

```
data(promoters_mm10)
```

---

promoters_mm9 *Promoters regions of mm9 Entrez genes.*

Description

Each regions have a width of 2000 nucleotide centered at the transcription start site.

Usage

```
promoters_mm9
```

Format

A GRanges object with 21677 ranges.

Value

A GRanges.

See Also

get_promoters_txdb

Examples

```
data(promoters_mm9)
```

Index