

# 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. Bootstrapping 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 . . . . .	2
get_demo_bam_files . . . . .	3
get_demo_design . . . . .	4
get_demo_metagene . . . . .	4
get_demo_regions . . . . .	5
get_promoters_txdb . . . . .	5
metagene . . . . .	6
permutation_test . . . . .	9
plot_metagene . . . . .	10
promoters_hg18 . . . . .	10
promoters_hg19 . . . . .	11
promoters_mm10 . . . . .	11
promoters_mm9 . . . . .	12

<b>Index</b>	<b>13</b>
--------------	-----------

---

Bam_Handler	<i>A class to manage BAM files.</i>
-------------	-------------------------------------

---

### 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	<i>Get regions filenames for demo</i>
------------------	---------------------------------------

---

**Description**

Get regions filenames for demo

**Usage**

```
get_demo_regions()
```

**Value**

A vector of regions filenames

**Examples**

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

---

get_promoters_txdb	<i>Extract Entrez genes promoters from TxDb object.</i>
--------------------	---

---

**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 = SerialParam())
```

**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(filenamees)
```

**filenamees** 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(filenamees)
```

**filenamees** 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)
```





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)
```

---

<code>promoters_hg19</code>	<i>Promoters regions of hg19 Entrez genes.</i>
-----------------------------	--

---

**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)
```

---

<code>promoters_mm10</code>	<i>Promoters regions of mm10 Entrez genes.</i>
-----------------------------	--

---

**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	<i>Promoters regions of mm9 Entrez genes.</i>
---------------	---

---

**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

## \*Topic **datasets**

- Bam\_Handler, [2](#)
- metagene, [6](#)
- promoters\_hg18, [10](#)
- promoters\_hg19, [11](#)
- promoters\_mm10, [11](#)
- promoters\_mm9, [12](#)

Bam\_Handler, [2](#)

- get\_demo\_bam\_files, [3](#)
- get\_demo\_design, [4](#)
- get\_demo\_metagene, [4](#)
- get\_demo\_regions, [5](#)
- get\_promoters\_txdb, [5](#), [10-12](#)

metagene, [6](#)

- permutation\_test, [9](#)
- plot\_metagene, [10](#)
- promoters\_hg18, [10](#)
- promoters\_hg19, [11](#)
- promoters\_mm10, [11](#)
- promoters\_mm9, [12](#)