Package ‘riboWaltz’

April 23, 2018

Type Package

Title Optimization of ribosome P-site positioning in ribosome profiling data

Version 1.0.0

Description riboWaltz is an R package designed for the analysis of ribosome profiling (RiboSeq) data aimed at the identification of the P-site offset. The P-site offset (PO) is specified by the localization of the P-site of ribosomes within the fragments of the RNA (reads) resulting from RiboSeq assays. It is defined as the distance of the P-site from the two ends of the reads. Determining the PO is a crucial step for a variety of RiboSeq-based analyses such as verify the so-called 3-nt periodicity of ribosomes along the coding sequence, derive translation initiation and elongation rates and reveal new translational events in unannotated open reading frames and ncRNAs. riboWaltz performs accurate computation of the PO for all the lengths of reads from single or multiple samples, taking advantage from an original two-step algorithm. Moreover, riboWaltz provides the user a variety of graphical representations, laying the groundwork for further positional analyses and new biological discoveries.

License MIT

LazyData TRUE

Depends R (>= 3.3.0)

Imports Biostrings (>= 2.46.0),
data.table (>= 1.10.4.3),
GenomicAlignments (>= 1.14.1),
GenomicFeatures (>= 1.24.5),
GenomicRanges (>= 1.24.3),
ggplot2 (>= 2.2.1),
ggrepel (>= 0.6.5),
IRanges (>= 2.12.0)

biocViews

RoxygenNote 6.0.1

Suggests knitr,
rmarkdown

VignetteBuilder knitr

R topics documented:

bamtobed 2
bamtolist 3
bedtolist 4
codon_coverage 6
codon_usage_psite 7
create_annotation 9
frame_psite 10
frame_psite_length 11
metaheatmap_psite 12
metaprofile_psite 14
mm81cdna 15
psite 16
psite_info 17
psite_offset 19
psite_per_cds 19
reads_list 20
reads_psite_list 21
region_psite 22
rends_heat 23
rlength_distr 24

Index **25**

bamtobed *Convert BAM files into BED files.*

Description

Converts one or several BAM files into a list of BED files containing for each read the name of the reference sequence (i.e. of the transcript) on which it aligns, the leftmost and rightmost position of the read, its length and the associated strand. Please note that thus function calls the bamtobed utility of the BEDTools suite.

Usage

```
bamtobed(bamfolder, bedfolder = NULL)
```

Arguments

bamfolder A character string specifying the path to the directory containing the BAM files. The function recursively looks for BAM format file starting from the specified folder.

bedfolder A character string specifying the (existing or not) location of the directory where the BED files should be stored. By default this argument is NULL, which implies the folder is set as a subdirectory of `bamfolder`, called *bed*.

Examples

```
## path_bam <- "location_of_BAM_files"
## path_bed <- "location_of_output_directory"
## bamtobed(bamfolder = path_bam, bedfolder = path_bed)
```

---

bamtolist *Convert BAM files into a list of data tables or into a GRangesList object.*

Description

Reads one or several BAM files, converts each file into a data table and combines them into a list. Alternatively, it returns a GRangesList i.e. a list of GRanges objects. In both cases the data structure contains for each read the name of the reference sequence (i.e. of the transcript) on which it aligns, the leftmost and rightmost position of the read and its length. Two additional columns are also attached, reporting the leftmost and rightmost position of the CDS of the reference sequence with respect to its 1st nucleotide. Please note: if a transcript is not associated to any annotated CDS then its start and the stop codon are set to 0. Multiple options for treating the read lengths are available.

Usage

```
bamtolist(bamfolder, annotation, transcript_align = TRUE,
  length_filter_mode = "none", length_filter_vector = NULL,
  periodicity_threshold = 50, list_name = NULL, granges = FALSE)
```

Arguments

bamfolder A character string indicating the path to the folder containing the BAM files.

annotation A data table from `create_annotation`. Please make sure that the name of the reference sequences in the annotation data table coincides with those in the BAM files.

transcript_align
A locigal value whether or not the BAM files within `bamfolder` refers to a transcriptome alignment (intended as an alignment based on a reference FASTA of all the transcript sequences). When this parameter is TRUE (the default) no reads mapping on the negative strand should be present and they are therefore removed.

`length_filter_mode`
: Either "none" (the default), "custom" or "periodicity". It specifies how to handle the selection of the read length. "none": all read lengths are included in the analysis; "custom": only read lengths specified by the user are included (see `length_filter_vector`); "periodicity": only read lengths satisfying a periodicity threshold (see `periodicity_threshold`) are included in the analysis. This mode enables the removal of all the reads without periodicity.

`length_filter_vector`
: An integer or an integer vector specifying either the read length or multiple read lengths to be kept, respectively. This parameter is considered only when `length_filter_mode` is set to "custom".

`periodicity_threshold`
: An integer in *[10, 100]*. Only the read lengths satisfying this threshold (i.e. with a higher percentage of read extremities falling in the same frame) are kept. This parameter is considered only when `length_filter_mode` is set to "periodicity". Default is 50.

`list_name`
: A character string vector specifying the desired names for the data tables of the output list. Its length must coincides with the number of BAM files within `bamfolder`. Please pay attention to the order in which they are provided: the first string is assigned to the first file, the second string to the second one and so on. By default this argument is NULL, implying that the data tables are named after the name of the BAM file, leaving their path and extension out.

`granges`
: A logical value whether or not to return a GRangesList object. Default is FALSE, meaning that a list of data tables (the required input for `psite`, `rends_heat` and `rlength_distr`) is returned instead.

Value

A list of data tables or a GRangesList object.

Examples

```
## path_bam <- "location_of_BAM_files"
## annotation_dt <- datatable_with_transcript_annotation
## bamtolist(bamfolder = path_bam, annotation = annotation_dt)
```

`bedtolist` *Convert BED files into a list of data tables or a GRangesList.*

Description

Reads one or several BED files, converts each file into a data table and combines them into a list. Alternatively, it returns a GRangesList i.e. a list of GRanges objects. In both cases two additional columns are attached to the data structures, reporting the leftmost and rightmost position of the CDS of the reference sequence with respect to its 1st nuclotide. Please note: if a transcript is not associated to any annotated CDS then its start and the stop codon are set to 0. Multiple options for treating the read lengths are available.

Usage

```
bedtolist(bedfolder, annotation, transcript_align = TRUE,
  length_filter_mode = "none", length_filter_vector = NULL,
  periodicity_threshold = 50, list_name = NULL, granges = FALSE)
```

Arguments

`bedfolder` A character string indicating the path to the folder containing the BED files.

`annotation` A data table from `create_annotation`. Please make sure that the name of the reference sequences in the annotation data table coincides with those in the BED files.

`transcript_align`
A locigal value whether or not the BED files within `bedfolder` refers to a transcriptome alignment (intended as an alignment based on a reference FASTA of all the transcript sequences). When this parameter is TRUE (the default) no reads mapping on the negative strand should be present and they are therefore removed.

`length_filter_mode`
Either "none" (the default), "custom" or "periodicity". It specifies how to handle the selection of the read length. "none": all read lengths are included in the analysis; "custom": only read lengths specified by the user are included (see `length_filter_vector`); "periodicity": only read lengths satisfying a periodicity threshold (see `periodicity_threshold`) are included in the analysis. This mode enables the removal of all the reads without periodicity.

`length_filter_vector`
An integer or an integer vector specifying either the read length or multiple read lengths to be kept, respectively. This parameter is considered only when `length_filter_mode` is set to "custom".

`periodicity_threshold`
An integer in *[10, 100]*. Only the read lengths satisfying this threshold (i.e. with a higher percentage of read extremities falling in the same frame) are kept. This parameter is considered only when `length_filter_mode` is set to "periodicity". Default is 50.

`list_name` A character string vector specifying the desired names for the data tables of the output list. Its length must coincides with the number of BED files within `bedfolder`. Please pay attention to the order in which they are provided: the first string is assigned to the first file, the second string to the second one and so on. By default this argument is NULL, implying that the data tables are named after the name of the BED file, leaving their path and extension out.

`granges` A logical value whether or not to return a GRangesList object. Default is FALSE, meaning that a list of data tables (the required input for `psite`, `rends_heat` and `rlength_distr`) is returned instead.

Value

A list of data tables or a GRangesList object.

Examples

```
## path_bed <- "location_of_BED_files"
## annotation_dt <- datatable_with_transcript_annotation
## bedtolist(bedfolder = path_bed, annotation = annotation_dt)
```

codon_coverage *Compute the number of reads per codon.*

Description

For the specified sample(s), this function computes the codon coverage defined either as the number of read footprints per codon or as the number of P-sites per codon.

Usage

```
codon_coverage(data, annotation, sample = NULL, psite = FALSE,
  min_overlap = 1, granges = FALSE)
```

Arguments

`data`	A list of data tables from `psite_info`. Data tables generated by `bamtolist` and `bedtolist` can be used only if `psite` is FALSE (the default).
`annotation`	A data table as generated by `create_annotation`.
`sample`	A character string vector specifying the name of the sample(s) of interest. By default this argument is NULL, meaning that the coverage is computed for all the samples in `data`.
`psite`	A logical value whether or not to return the number of P-sites per codon. Default is NULL, meaning that the number of read footprints per codon is computed instead.
`min_overlap`	A positive integer specyfing the minimum number of overlapping positions (in nucleotides) between a reads and a codon to be considered to be overlapping. When `psite` is TRUE this parameter must be 1 (the default).
`granges`	A logical value whether or not to return a GRanges object. Default is FALSE, meaning that a data tables is returned instead.

Details

The sequence of every transcript is divided in triplets starting from the annotated translation initiation site (if any) proceeding towards the UTRs extremities, and eventually discarding the exceeding 1 or 2 nucleotides at the extremities of the transcript. Please note that the transcripts not associated to any annotated *5' UTR*, *CDS* and *3'UTR* and transcripts with coding sequence length not divisible by 3 are automatically discarded.

Value

A data table or a GRanges object.

Examples

```
data(reads_psite_list)
data(mm81cdna)

## Compute the coverage based on the number of ribosome footprint per codon,
## setting the minimum overlap between reads and triplets to 3 nts
## coverage_dt <- codon_coverage(reads_psite_list, mm81cdna, min_overlap = 3)

## Compute the coverage based on the number of P-sites per codon
##coverage_dt <- codon_coverage(reads_psite_list, mm81cdna, psite = TRUE)
```

codon_usage_psite — *Compute and plot empirical codon usage indexes based on the inferred P-sites.*

Description

For a specified sample this function computes an empirical codon usage index based on the frequency of in-frame P-sites along the coding sequence. It computes the codon usage index for all the 64 triplets, normalizes them for the frequency of each codon within the CDS and returns a bar plot with the resulting values. This function also allows to compare the computed codon usage indexes with a set of 64 values provided by the user.

Usage

```
codon_usage_psite(data, annotation, sample, fastapath = NULL,
  fasta_genome = TRUE, bsgenome = NULL, gtfpath = NULL, txdb = NULL,
  dataSource = NA, organism = NA, transcripts = NULL,
  codon_values = NULL, scatter_label = FALSE, aminoacid = FALSE)
```

Arguments

`data` A list of data tables from `psite_info` that may or may not include a column named *psite_codon*. This column reports the three nucleotides covered by the P-site and can be previously generated by the `psite_info` function. If not already present, it is automatically generated throughout the function starting from a FASTA file or a BSgenome data package.

`annotation` A data table as generated by `create_annotation`.

`sample` A character string vector specifying the name of the sample of interest.

`fastapath` An optional character string specifying the path to the FASTA file used in the alignment step, including its name and extension. This file can contain reference nucleotide sequences either of a genome assembly or of all the transcripts (see `fasta_genome`). Please make sure the sequences derive from the same release of the annotation file used in the `create_annotation` function. Note: either `fastapath` or `bsgenome` is required to normalize the data, even if a column *psite_codon* has been previously generated by `psite_info`. Default is NULL.

`fasta_genome` A locigal value whether or not the FASTA file specified by `fastapath` contains nucleotide sequences of a genome assembly. FALSE means that the nucleotide sequences of all the transcripts are provided instead. When this parameter is TRUE (the default), an annotation object is required (see `gtfpath` and `txdb`).

`bsgenome` An optional character string specifying the name of the BSgenome data package containing the genome sequences to be loaded. If it is not already present in your system, it will be installed through the biocLite.R script (check the list of data packages available in the Bioconductor repositories for your version of R/Bioconductor by the `available.genomes` function of the BSgenome package). This parameter also requires an annotation object (see `gtfpath` and `txdb`). Please make sure the sequences included in the specified BSgenome data pakage are in agreement with the sequences used in the alignment step. Note: either `fastapath` or `bsgenome` is required to normalize the data, even if a column *psite_codon* has been previously generated by `psite_info`. Default is NULL.

`gtfpath` A character string specifying the path to te GTF file, including its name and extension. Please make sure the GTF derives from the same release of what is specified by `fastapath` or by `bsgenome`. Note that either `gtfpath` or `txdb` must be specified when the nucleotide sequences of a genome assembly are provided (see `fastapath` or `bsgenome`). Default is NULL.

`txdb` A character string specifying the name of the annotation package for TxDb object(s) to be loaded. If it is not already present in your system, it will be installed through the biocLite.R script (check the list of TxDb annotation packages available in the Bioconductor repositories at http://bioconductor.org/packages/release/BiocViews.html#___Tx)). Please make sure the annotation package derives from the same release of what is specified by `fastapath` or by `bsgenome`. Note that either `gtfpath` or `txdb` must be specified when the nucleotide sequences of a genome assembly are provided (see `fastapath` or `bsgenome`). Default is NULL.

`dataSource` An optional character string describing the origin of the GTF data file. For more information about this parameter please refer to the description of *dataSource* of the `makeTxDbFromGFF` function included in the `GenomicFeatures` package.

`organism` A optional character string reporting the genus and species of the organism when `gtfpath` is specified. For more information about this parameter please refer to the description of *dataSource* of the `makeTxDbFromGFF` function included in the `GenomicFeatures` package.

`transcripts` A character string vector specifying the name of the transcripts to be included in the analysis. By default this argument is NULL, meaning that all the transcripts in `data` are used. Please note that the transcripts not associated to any annotated *5' UTR*, *CDS* and *3'UTR* and transcripts with coding sequence length not divisible by 3 are automatically discarded.

`codon_values` A data table containing codon-specific values provided by the user. These values are compared with the empirical codon usage indexes of the sample of interest. The data table must contain at least the 64 codons and the corresponding values arranged in two columns named *codon* and *value*, respectively. Note that a similar data table is also returned by `codon_usage_psite` itself. Default is NULL.

`scatter_label` A logical value whether or not to label the dots of the scatter plot generated by specifying `codon_values`. Each dot can be labeled either after the three nucleotides of the codon or after the corresponding amino acid (see `aminoacid`). This parameter is considered only if `codon_values` is specified. Default is FALSE.

`aminoacid` A logical value whether or not to label the dots of the scatter plot generated by specifying `codon_values` using the amino acids corresponding to the triplets. Default is FALSE, meaning that the three nucleotides of the codon are used instead. This parameter is considered only if `codon_values` is specified and `scatter_label` is TRUE. Default is FALSE.

Value

A list containing a ggplot2 object (named "plot"), and a data table ("dt") with the associated data. An additional ggplot2 object ("plot_comparison") is returned if `codon_values` is specified.

`create_annotation` *Create an annotation data table.*

Description

Starting from a GTF file or a TxDb object this function generates a dada table containing a basic annotation of the transcripts. The data table includes a column named *transcript* reporting the name of the reference sequences and four columns named *l_tr*, *l_utr5*, *l_cds* and *l_utr3* reporting the length of the transcripts and of their annotated *5' UTR*, *CDS* and *3' UTR*, respectively.

Usage

```
create_annotation(gtfpath = NULL, txdb = NULL, dataSource = NA,
  organism = NA)
```

Arguments

`gtfpath` A character string specifying the path to te GTF file, including its name and extension. Please make sure the GTF derives from the same release of the sequences used in the alignment step. Note that either `gtfpath` or `txdb` must be specified.

`txdb` A character string specifying the name of the annotation package for TxDb object(s) to be loaded. If it is not already present in your system, it will be installed through the biocLite.R script (check the list of TxDb annotation packages available in the Bioconductor repositories at http://bioconductor.org/packages/release/BiocViews.html#___Tx)). Please make sure the annotation package derives from the same release of the sequences used in the alignment step. Note that either `gtfpath` or `txdb` must be specified.

`dataSource` An optional character string describing the origin of the GTF data file. For more information about this parameter please refer to the description of *dataSource* of the `makeTxDbFromGFF` function included in the `GenomicFeatures` package.

organism — A optional character string reporting the genus and species of the organism when gtfpath is specified. For more information about this parameter please refer to the description of *dataSource* of the makeTxDbFromGFF function included in the GenomicFeatures package.

Value

A data table.

Examples

```
## gtf_file <- location_of_GTF_file
## path_bed <- location_of_output_directory
## bamtobed(gtfpath = gtf_file, dataSource = "gencode6", organism = "Mus musculus")
```

frame_psite *Compute the percentage of P-sites per frame.*

Description

For one or several samples this function computes the percentage of P-sites falling on the three reading frames of the transcripts and generates a barplot of the resulting values. This analysis is performed for the annotated 5' UTR, coding sequence and 3' UTR, separately. It is possible to compute the percentage of P-sites per frame using all the read lengths or to restrict the analysis to a sub-range of read lengths.

Usage

```
frame_psite(data, sample = NULL, region = "all", length_range = "all",
  plot_title = NULL)
```

Arguments

data — A list of data tables from psite_info.

sample — A character string vector specifying the name of the sample(s) of interest. By default this argument is NULL, meaning that all the samples in data are included in the analysis.

region — Either "all" or a character string among "5utr", "cds", "3utr" specifying the regions of the transcript (5' UTR, CDS or 3' UTR, respectively) that must be included in the analysis. Default is "all", meaning that the all the regions are considered.

length_range — Either "all", an integer or an integer vector. Default is "all", meaning that all the read lengths are included in the analysis. Otherwise, only the read lengths matching the specified value(s) are kept.

plot_title — Any character string specifying the title of the plot. If "auto", the title of the plot reports the region specified by region (if any) and the length(s) of the reads used for generating the barplot. Default is NULL, meaning that no title will be added to the plot.

Value

A list containing a ggplot2 object and a data table with the associated data.

Examples

```
data(reads_psite_list)

## Generate the barplot for all the read lengths
frame_whole <- frame_psite(reads_psite_list, sample = "Samp1")

## Generate the barplot restricting the analysis to the coding sequence and
## to the reads of 28 nucleotides
frame_sub <- frame_psite(reads_psite_list, sample = "Samp1", region = "cds",
length_range = 28)
```

`frame_psite_length` *Compute the number of P-sites per frame stratified by read length.*

Description

Similar to `frame_psite` but the results are stratified by the length of the reads.

Usage

```
frame_psite_length(data, sample = NULL, region = "all", cl = 100,
  length_range = "all", plot_title = NULL)
```

Arguments

`data`	A list of data tables from `psite_info`.
`sample`	A character string vector specifying the name of the sample(s) of interest. By default this argument is NULL, meaning that all the samples in `data` are included in the analysis.
`region`	Either "all" or a character string among "5utr", "cds", "3utr" specifying the regions of the transcript (5' UTR, CDS or 3' UTR, respectively) that must be included in the analysis. Default is "all", meaning that the all the regions are considered.
`cl`	An integer value in *[1,100]* specifying the confidence level for restricting the analysis to a sub-range of read lengths. Default is 100. This parameter has no effect if `length_range` is specified.
`length_range`	Either "all", an integer or an integer vector. Default is "all", meaning that all the read lengths are included in the analysis. Otherwise, only the read lengths matching the specified value(s) are kept. If specified, this parameter prevails over `cl`.
`plot_title`	Any character string specifying the title of the plot. When "auto", the title of the plot reports the region specified by `region` (if any). Default is NULL, meaning that no title will be added to the plot.

Value

A list containing a ggplot2 object and a data table with the associated data.

Examples

```
data(reads_psite_list)

## Generate the heatmap for all the read lengths
frame_len_whole <- frame_psite_length(reads_psite_list, sample = "Samp1")

## Generate the heatmap for a sub-range of read lengths (the middle 90%) and
## restricting the analysis to the coding sequence
frame_len_sub <- frame_psite_length(reads_psite_list, sample = "Samp1",
region = "cds", cl = 90)
```

`metaheatmap_psite` *Plot ribosome occupancy metaheatmaps at single-nucleotide resolution.*

Description

For one or more sample this function plots a heatmap-like metaprofile based on the P-site of the reads mapping around the start and the stop codon of the annotated CDS (if any). It works similarly to `metaprofile_psite` but the intensity of the signal is represented by a continuous color scale rather than by the height of a line chart. This graphical output is a good option for analyzing several samples at once and for comparing the profiles generated by different reads lengths or in multiple conditions.

Usage

```
metaheatmap_psite(data, annotation, sample, scale_factors = NULL,
  length_range = "all", transcripts = NULL, utr5l = 25, cdsl = 50,
  utr3l = 25, log = F, colour = "black", plot_title = NULL)
```

Arguments

`data` A list of data tables from `psite_info`.

`annotation` A data table as generated by `create_annotation`.

`sample` A list of character string vectors specifying the name of the samples (or of its replicates) of interest. The elements of each vector are merge together using the scale factors specified by `scale_factors`. The name of the elements of the list are used for labelling the raws of the heatmap.

`scale_factors` A numeric vector of scale factors for merging the replicates (if any). The vector must contain at least one value for each replicates, named after the strings listed in `sample`. No specific order is required. Default is NULL, meaning that all the scale factors are set to 1.

`length_range` Either "all", an integer or an integer vector. Default is "all", meaning that all the read lengths are included in the analysis. Otherwise, only the read lengths matching the specified value(s) are kept.

`transcripts` A character string vector specifying the name of the transcripts to be included in the analysis. By default this argument is NULL, meaning that all the transcripts in `data` are used. Note that if either the 5' UTR, the coding sequence or the 3' UTR of a transcript is shorther than `utr5l`, 2*`cdsl` and `utr3l` respectively, the transcript is automatically discarded.

`utr5l` A positive integer specifying the length (in nucleotides) of the 5' UTR region that in the plot flanks the start codon. The default value is 25.

`cdsl` A positive integer specifying the length (in nucleotides) of the CDS region that in the plot will flank both the start and the stop codon. The default value is 50.

`utr3l` A positive integer specifying the length (in nucleotides) of the 3' UTR region that in the plot flanks the start codon. The default value is 25.

`log` A logical value whether or not to use a logarithmic scale colour (strongly suggested in case of large variations of the signal). Default is FALSE.

`colour` A character string specifying the colour of the plot. Default is "black".

`plot_title` Any character string specifying the title of the plot. When "auto", the title of the plot reports the number of the transcripts and the length(s) of the reads considered for generating the metaprofile. Default is NULL, meaning that no title will be added to the plot.

Value

A list containing a ggplot2 object, a data table with the associated data and the transcripts employed for generating the plot.

Examples

```
data(reads_psite_list)

## Generate the metaheatmap for all the read lengths
metaheat_whole <- metaheatmap_psite(reads_psite_list, mm81cdna, sample = list("Whole"=c("Samp1")))

## Generate the metaheatmap employing reads of 27, 28 and 29 nucleotides and
## a subset of transcripts (for example with at least one P-site mapping on the
## translation initiation site)
sample_name <- "Samp1"
sub_reads_psite_list <- subset(reads_psite_list[[sample_name]], psite_from_start == 0)
transcript_names <- as.character(sub_reads_psite_list$transcript)
metaheat_sub <- metaheatmap_psite(reads_psite_list, mm81cdna, sample = list("sub"=sample_name),
length_range = 27:29, transcripts = transcript_names, plot_title = "auto")

## Generate two metaheatmaps displaied in the same plot. In this exampe one
## data table includes all the read lengths while in the other one contains only
## reads of 28 nucleotides
sample_name <- "Samp1"
metaheat_df <- list()
```

```
metaheat_df[["subsample_28nt"]] <- subset(reads_psite_list[[sample_name]], length == 28)
metaheat_df[["whole_sample"]] <- reads_psite_list[[sample_name]]
names_list <- list("Only_28" = c("subsample_28nt"), "All" = c("whole_sample"))
metaheat_comparison <- metaheatmap_psite(metaheat_df, mm81cdna, sample = names_list)
```

`metaprofile_psite` *Plot ribosome occupancy metaprofiles at single-nucleotide resolution.*

Description

For a specified sample this function generates a metaprofile based on the P-site of the reads mapping around the start and the stop codon of the annotated CDS (if any). It sums up the number of P-sites (defined by their first nucleotide) per nucleotide computed for all the transcripts starting from one ore more replicates.

Usage

```
metaprofile_psite(data, annotation, sample, scale_factors = NULL,
  length_range = "all", transcripts = NULL, utr5l = 25, cdsl = 50,
  utr3l = 25, plot_title = NULL)
```

Arguments

`data`	A list of data tables from `psite_info`.
`annotation`	A data table as generated by `create_annotation`.
`sample`	A character string vector specifying the name of the sample (or of its replicates) of interest. Its elements are merge together using the scale factors specified by `scale_factors`.
`scale_factors`	A numeric vector of scale factors for merging the replicates (if any). The vector must contain at least one value for each replicates, named after the strings listed in `sample`. No specific order is required. Default is NULL, meaning that all the scale factors are set to 1.
`length_range`	Either "all", an integer or an integer vector. Default is "all", meaning that all the read lengths are included in the analysis. Otherwise, only the read lengths matching the specified value(s) are kept.
`transcripts`	A character string vector specifying the name of the transcripts to be included in the analysis. By default this argument is NULL, meaning that all the transcripts in `data` are used. Note that if either the 5' UTR, the coding sequence or the 3' UTR of a transcript is shorter than `utr5l`, 2∗`cdsl` and `utr3l` respectively, the transcript is automatically discarded.
`utr5l`	A positive integer specifying the length (in nucleotides) of the 5' UTR region that in the plot flanks the start codon. The default value is 25.
`cdsl`	A positive integer specifying the length (in nucleotides) of the CDS region that in the plot will flank both the start and the stop codon. The default value is 50.

`utr3l` A positive integer specifying the length (in nucleotides) of the 3' UTR region that in the plot flanks the start codon. The default value is 25.

`plot_title` Any character string specifying the title of the plot. When "auto", the title of the plot reports the sample(s) specified by `sample` and the number of the transcripts and the length(s) of the reads considered for generating the metaprofile. Default is NULL, meaning that no title will be added to the plot.

Value

A list containing a ggplot2 object, a data table with the associated data and the transcripts employed for generating the plot.

Examples

```
data(reads_psite_list)
data(mm81cdna)

## Generate the metaprofile for all the read lengths
metaprof_whole <- metaprofile_psite(reads_psite_list, mm81cdna, sample = "Samp1")
metaprof_whole[["plot"]]

## Generate the metaprofile employing reads of 27, 28 and 29 nucleotides and
## a subset of transcripts (for example with at least one P-site mapping on
## the translation initiation site)
sample_name <- "Samp1"
sub_reads_psite_list <- subset(reads_psite_list[[sample_name]], psite_from_start == 0)
transcript_names <- as.character(sub_reads_psite_list$transcript)
metaprof_sub <- metaprofile_psite(reads_psite_list, mm81cdna, sample = sample_name,
length_range = 27:29, transcripts = transcript_names)
```

`mm81cdna` *Annotation*

Description

A dataset containing basic information about 109,712 mouse mRNA (using the Ensembl v81 transcript annotation).

Usage

```
mm81cdna
```

Format

A data table with 109,712 rows and 5 variables (the lengths are expressed in nucleotides):

transcript Name of the transcript (ENST ID and version, dot separated)

l_tr Length of the transcript

l_utr5 Length of the annotated 5' UTR (if any)

l_cds Length of the annotated CDS (if any)

l_utr3 Length of the annotated 3' UTR (if any)

psite *Identify the ribosome P-site position within the reads.*

Description

This function identifies within each read the position of the ribosome P-site, determined by the localisation of its first nucleotide. The function processes the samples separately starting from the reads aligning on the reference codon (selected by the user between the start codon and the second to last codon) of any annotated coding sequence. It then returns the position of the P-site specifically inferred for all the read lengths. It also allows to plot a collection of read length-specific occupancy metaprofiles showing the P-sites offsets computed throughout the two steps of the algorithm.

Usage

```
psite(data, flanking = 6, start = TRUE, extremity = "auto",
  plot = FALSE, plotdir = NULL, plotformat = "png", cl = 99)
```

Arguments

`data`	A list of data tables from either `bamtolist` or `bedtolist`.
`flanking`	An integer that specifies how many nucleotides, at least, of the reads mapping on the reference codon must flank the reference codon in both directions. Default is 6.
`start`	A logical value whether ot not to compute the P-site offsets starting from the reads aligning on the translation initiation site. FALSE implies that the reads mapping on the last triplet before the stop codon are used instead. Default is TRUE.
`extremity`	A character string specifing which extremity of the reads should be used in the correction step of the algorithm. It can be either "5end" or "3end" for the 5' and the 3' extremity, respectively. Default is "auto", meaning that the best extremity is automatically selected.
`plot`	A logical value whether or not to plot the occupancy metaprofiles showing the P-sites offsets computed throughout the two steps of the algorithm. Default is FALSE.
`plotdir`	A character string specifying the (existing or not) location of the directory where the occupancy metaprofiles shuold be stored. This parameter is considered only if `plot` is TRUE. By default this argument is NULL, which implies it is set as a subfolder of the working directory, called *offset_plot*.
`plotformat`	Either "png" (the default) or "pdf", this parameter specifies the file format of the generated metaprofiles. It is considered only if `plot` is TRUE.
`cl`	An integer value in *[1,100]* specifying the confidence level for restricting the generation of the occupancy metaprofiles to a sub-range of read lengths. By default it is set to 99. This parameter is considered only if `plot` is TRUE.

Value

A data table.

Examples

```
data(reads_list)

## Compute the P-site offset automatically selecting the otimal read
## extremity for the correction step and not plotting any metaprofile
psite(reads_list, flanking = 6, extremity="auto")

## Compute the P-site offset specifying the extremity used in the correction
## step and plotting the metaprofiles only for a sub-range of read lengths (the
## middle 95%). The plots will be placed in the current working directory.
psite_offset <- psite(reads_list, flanking = 6, extremity="3end", plot = TRUE, cl = 95)
```

psite_info *Update reads information according to the inferred P-sites.*

Description

Starting ftom the P-site position identfied by psite, this function updates the data tables that contains information about the reads. It attaches to the data tables 4 columns reporting the P-site position with respect to the 1st nucleotide of the transcript, the start and the stop codon of the annotated coding sequence (if any) and the region of the transcript (5' UTR, CDS, 3' UTR) that includes the P-site. Please note: if a transcript is not associated to any annotated CDS then the positions of the P-site from both the start and the stop codon is set to NA. If either a FASTA file or a BSgenome data package with the nucleotide sequences is provided, an additional column reporting the three nucleotides covered by the P-site is attached.

Usage

```
psite_info(data, offset, fastapath = NULL, fasta_genome = TRUE,
  bsgenome = NULL, gtfpath = NULL, txdb = NULL, dataSource = NA,
  organism = NA, granges = FALSE)
```

Arguments

data	A list of data tables from either bamtolist or bedtolist.
offset	A data table from psite.
fastapath	An optional character string specifying the path to the FASTA file used in the alignment step, including its name and extension. This file can contain reference nucleotide sequences either of a genome assembly or of all the transcripts (see fasta_genome). Please make sure the sequences derive from the same release of the annotation file used in the create_annotation function. Note: either fastapath or bsgenome is required to generate an additional column reporting the three nucletotides covered by the P-sites. Default is NULL.

`fasta_genome` A locigal value whether or not the FASTA file specified by `fastapath` contains nucleotide sequences of a genome assembly. FALSE means that the nucleotide sequences of all the transcripts are provided instead. When this parameter is TRUE (the default), an annotation object is required (see `gtfpath` and `txdb`).

`bsgenome` An optional character string specifying the name of the BSgenome data package containing the genome sequences to be loaded. If it is not already present in your system, it will be installed through the biocLite.R script (check the list of data packages available in the Bioconductor repositories for your version of R/Bioconductor by the `available.genomes` function of the BSgenome package). This parameter also requires an annotation object (see `gtfpath` and `txdb`). Please make sure the sequences included in the specified BSgenome data pakage are in agreement with the sequences used in the alignment step. Note: either `fastapath` or `bsgenome` is required to generate an additional column reporting the three nucletotides covered by the P-sites. Default is NULL.

`gtfpath` A character string specifying the path to te GTF file, including its name and extension. Please make sure the GTF derives from the same release of what is specified by `fastapath` or by `bsgenome`. Note that either `gtfpath` or `txdb` must be specified when the nucleotide sequences of a genome assembly are provided (see `fastapath` or `bsgenome`). Default is NULL.

`txdb` A character string specifying the name of the annotation package for TxDb object(s) to be loaded. If it is not already present in your system, it will be installed through the biocLite.R script (check the list of TxDb annotation packages available in the Bioconductor repositories at http://bioconductor.org/packages/release/BiocViews.html#___Tx
)). Please make sure the annotation package derives from the same release of what is specified by `fastapath` or by `bsgenome`. Note that either `gtfpath` or `txdb` must be specified when the nucleotide sequences of a genome assembly are provided (see `fastapath` or `bsgenome`). Default is NULL.

`dataSource` An optional character string describing the origin of the GTF data file. For more information about this parameter please refer to the description of *dataSource* of the `makeTxDbFromGFF` function included in the `GenomicFeatures` package.

`organism` A optional character string reporting the genus and species of the organism when `gtfpath` is specified. For more information about this parameter please refer to the description of *dataSource* of the `makeTxDbFromGFF` function included in the `GenomicFeatures` package.

`granges` A logical value whether or not to return a GRangesList object. Default is FALSE, meaning that a list of data tables (the required input for the downstream analyses and graphical outputs provided by riboWaltz) is returned instead.

Value

A list of data tables or a GRangesList object.

Examples

```
data(reads_list)
data(psite_offset)
data(mm81cdna)
```

```
reads_psite_list <- psite_info(reads_list, psite_offset)
```

`psite_offset` *P-site offsets*

Description

This dataset contains information on the offset computed by `psite` starting from `reads_list`.

Usage

```
psite_offset
```

Format

A data table with 31 rows and 9 variables (the lengths and the distances are expressed in nucleotides):

length Length of the read

total_percentage Percentage of reads of the considered length in the whole dataset

start_percentage Percentage of reads of the considered length aligning on the start codon (if any)

around_start A logical value reporting whether at least one read of the specified length aligns on the start codon (T = yes, F = no)

offset_from_5 Temporary P-site offset from the 5' end of read (before the correction step)

offset_from_3 Temporary P-site offset from the 3' end of read (before the correction step)

adj_offset_from_5 P-site offset from the 5' end of read after the correction step

adj_offset_from_3 P-site offset from the 3' end of read after the correction step

sample Name of the sample

`psite_per_cds` *Compute the number of in-frame P-sites per coding sequence.*

Description

For each sample and each transcript this function computes the number of P-sites in frame 0 within the coding sequence. It is possible to exclude from the analysis a specified number of nucleotides at the beginiing and/or at the end of the CDS, restricting the analysis to a subsequence of the coding region. Please note that only the transcripts associated to an annotated CDS are kept for the analysis. The resulting data table reports the name of the transripts along with the length of the considered region (in nucleotides) and the associated number of P-sites for all the samples.

Usage

```
psite_per_cds(data, annotation, start_nts = 0, stop_nts = 0)
```

Arguments

`data` A list of data tables from `psite_info`.

`annotation` A data table as generated by `create_annotation`.

`start_nts` A positive integer specifying the number of nucleotides at the beginning of the coding sequences to be exluded from the analisys Default is 0.

`stop_nts` A positive integer specifying the number of nucleotides at the end of the coding sequences to be exluded from the analisys. Default is 0.

Value

A data table.

Examples

```
data(reads_psite_list)
data(mm81cdna)

## Compute the number of P-sites in frame on the whole coding sequence.
psite_cds <- psite_per_cds(reads_psite_list, mm81cdna)

## Compute the number of P-sites in frame on the coding sequence exluding
## the first 15 nucleotides and the last 10 nucleotides.
psite_cds <- psite_per_cds(reads_psite_list, mm81cdna, start_nts = 15, stop_nts = 10)
```

`reads_list` *Reads information*

Description

This dataset contains details on mapping reads from BAM or BED files.

Usage

```
reads_list
```

Format

A list of data tables with 1 object (named *Samp1*) of 393,338 rows and 6 variables (the lengths and the distances are expressed in nucleotides):

transcript Name of the transcript (ENST ID and version, dot separated)

end5 Position of the 5’ end of the read with respect to the first nuclotide of the transcript

end3 Position of the 3’ end of the read with respect to the first nuclotide of the transcript

length Length of the read

start_pos Leftmost position of the annotated CDS with respect to the first nuclotide of the transcript

stop_pos Rightmost position of the annotated CDS with respect to the first nuclotide of the transcript

`reads_psite_list` *P-sites and reads information*

Description

This dataset contains details on mapping reads after the identification of the P-site and the update of `reads_list`.

Usage

```
reads_psite_list
```

Format

A list of data tables with 1 object (named *Samp1*) of 393,338 rows and 10 variables (the lengths and the distances are expressed in nucleotides):

transcript Name of the transcript (ENST ID and version, dot separated)

end5 Position of the 5' end of the read with respect to the first nuclotide of the transcript

psite Position of the P-site with respect to the first nuclotide of the transcript

end3 Position of the 3' end of the read with respect to the first nuclotide of the transcript

length Length of the read

start_pos Leftmost position of the CDS with respect to the first nuclotide of the transcript

stop_pos Rightmost position of the CDS with respect to the first nuclotide of the transcript

psite_from_start Position of the P-site with respect to the first nuclotide of the annotated CDS (if any)

psite_from_stop Position of the P-site with respect to the last nuclotide of the annotated CDS (if any)

psite_region Region of the transcript that includes the P-site (5utr, cds, 3utr) `region_psite` *Plot the percentage of P-sites per transcript region.*

Description

For one or several samples this function computes the percentage of P-sites falling in the three annotated regions of the transcripts (5' UTR, CDS and 3'UTR) and generates a barplot of the resulting values. The function also calculates and plots the percentage of region length for the selected transcripts (reported in column "RNAs").

Usage

```
region_psite(data, annotation, sample = NULL, transcripts = NULL,
  label = NULL, colour = c("gray70", "gray40", "gray10"))
```

Arguments

`data`	A list of data tables from `psite_info`.
`annotation`	A data table as generated by `create_annotation`.
`sample`	A character string vector specifying the name of the sample(s) of interest. By default this argument is NULL, meaning that all the samples in `data` are included in the analysis.
`transcripts`	A character string vector specifying the name of the transcripts to be included in the analysis. By default this argument is NULL, meaning that all the transcripts in `data` are used. Please note that the transcripts not associated to any annotated *5' UTR*, *CDS* and *3'UTR* are automatically discarded.
`label`	A character string vector of the same length of `sample` specifying the name of the samples to be displaied in the plot. By default this argument is NULL meaning that the name of the samples are used.
`colour`	A character string vector of three elements specifying the colours of the bars corresponding to the *5' UTR*, the *CDS* and the *3'UTR* respectively. The default is a grayscale.

Value

A list containing a ggplot2 object, and a data table with the associated data.

Examples

```
data(reads_psite_list)
data(mm81cdna)

reg_psite <- region_psite(reads_psite_list, mm81cdna, sample = "Samp1")
reg_psite[["plot"]]
```

rends_heat *Plot metaheatmaps based on the two extremities of the reads.*

Description

For a specified sample this function plots four metaheatmaps showing the abundance of the 5' and the 3' end of the reads mapping around the start and the stop codon of the annotated CDS (if any), stratified by their length. It is possible to visualise the metaheatmaps for all the read lengths or to restrict the graphical output to a sub-range of read lengths.

Usage

```
rends_heat(data, annotation, sample, transcripts = NULL, cl = 95,
  utr5l = 50, cdsl = 50, utr3l = 50, log = F, colour = "black")
```

Arguments

`data`	A list of data tables from either `bamtolist` or `bedtolist`.
`annotation`	A data table as generated by `create_annotation`.
`sample`	A character string specifying the name of the sample of interest.
`transcripts`	A character string vector specifying the name of the transcripts to be included in the analysis. By default this argument is NULL, meaning that all the transcripts in `data` are used. Note that if either the 5' UTR, the coding sequence or the 3' UTR of a transcript is shorter than `utr5l`, 2∗`cdsl` and `utr3l` respectively, the transcript is automatically discarded.
`cl`	An integer value in *[1,100]* specifying the confidence level for restricting the plot to a sub-range of read lengths. Default is 95.
`utr5l`	A positive integer specifying the length (in nucleotides) of the 5' UTR region that in the plot flanks the start codon. The default value is 50.
`cdsl`	A positive integer specifying the length (in nucleotides) of the CDS region that in the plot will flank both the start and the stop codon. The default value is 50.
`utr3l`	A positive integer specifying the length (in nucleotides) of the 3' UTR region that in the plot flanks the start codon. The default value is 50.
`log`	A logical value whether or not to use a logarithmic scale colour (strongly suggested in case of large variations of the signal). Default is FALSE.
`colour`	A character string specifying the colour of the plot. Default is "black".

Value

A list containing a ggplot2 object, and a data table with the associated data.

Examples

```
data(reads_list)
data(mm81cdna)

## Visualise the metaheatmaps for all the read lengths
heatend_whole <- rends_heat(reads_list, mm81cdna, sample = "Samp1", cl = 100)

## Visualise the metaheatmaps for a sub-range of read lengths (the middle
## 95%) reducing the flanking regions around the start and the stop codon
heatend_sub95 <- rends_heat(reads_list, mm81cdna, sample = "Samp1", cl = 95,
utr5l = 30, cdsl = 40, utr3l = 30)
```

---

rlength_distr *Plot read length distributions.*

Description

For a specified sample this function plots the read length distribution. It is possible to visualise the distribution for all the read lengths or to restrict the graphical output to a sub-range of read lengths.

Usage

```
rlength_distr(data, sample, cl = 100)
```

Arguments

data	A list of data tables from either bamtolist or bedtolist.
sample	A character string specifying the name of the sample of interest.
cl	An integer value in *[1,100]* specifying the confidence level for restricting the plot to a sub-range of read lengths. By default it is set to 100, meaning that the whole distribution is displayed.

Value

A list containing a ggplot2 object, and a data table with the associated data.

Examples

```
data(reads_list)

## Visualise distribution for all the read lengths
lendist_whole <- rlength_distr(reads_list, sample = "Samp1", cl = 100)
lendist_whole[["plot"]]

## Visualise the metaheatmaps for a sub-range of read lengths (the middle 95%)
lendist_sub95 <- rlength_distr(reads_list, sample = "Samp1", cl = 95)
lendist_sub95[["plot"]]
```

Index

∗Topic **datasets**
mm81cdna, 15
psite_offset, 19
reads_list, 20
reads_psite_list, 21

available.genomes, *8*, *18*

bamtobed, 2
bamtolist, 3, *6*, *16*, *17*, *23*, *24*
bedtolist, 4, *6*, *16*, *17*, *23*, *24*

codon_coverage, 6
codon_usage_psite, 7, *8*
create_annotation, *3*, *5–7*, 9, *12*, *14*, *17*, *20*, *22*, *23*

frame_psite, 10, *11*
frame_psite_length, 11

makeTxDbFromGFF, *8–10*, *18*
metaheatmap_psite, 12
metaprofile_psite, *12*, 14
mm81cdna, 15

psite, *4*, *5*, 16, *17*, *19*
psite_info, *6–8*, *10–12*, *14*, 17, *20*, *22*
psite_offset, 19
psite_per_cds, 19

reads_list, *19*, 20, *21*
reads_psite_list, 21
region_psite, 22
rends_heat, *4*, *5*, 23
rlength_distr, *4*, *5*, 24