Package 'RiboCrypt'

May 2, 2026

Type Package

Title Interactive visualization in genomics

Version 1.19.0

License MIT + file LICENSE

Description R Package for interactive visualization and browsing NGS data.
It contains a browser for both transcript and genomic coordinate view.
In addition a QC and general metaplots are included, among others differential translation plots and gene expression plots. The package is still under development.

biocViews Software, Sequencing, RiboSeq, RNASeq,

Encoding UTF-8

LazyData true

BugReports https://github.com/m-swirski/RiboCrypt/issues

URL https://github.com/m-swirski/RiboCrypt

Depends R (>= 3.6.0), ORFik (>= 1.13.12)

Imports bslib, BiocGenerics, BiocParallel, Biostrings, ComplexHeatmap, cowplot, crosstalk, data.table, dplyr, DT, fst, Seqinfo, GenomicFeatures, GenomicRanges, ggplot2, grid, htmlwidgets, httr, IRanges, jsonlite, knitr, markdown, NGLVieweR, plotly, rlang, rclipboard, RCurl, rtracklayer, shiny, shinycssloaders, shinyhelper, shinyjs, shinyjqui, shinyWidgets, stringr, writexl

Suggests testthat, rmarkdown, BiocStyle, BSgenome, BSgenome.Hsapiens.UCSC.hg19

RoxygenNote 7.3.1

VignetteBuilder knitr

git_url https://git.bioconductor.org/packages/RiboCrypt

git_branch devel

git_last_commit cbd1ebc

git_last_commit_date 2026-04-28

Repository Bioconductor 3.24

Date/Publication 2026-05-01

Author Michal Swirski [aut, cre, cph], Haakon Tjeldnes [aut, ctb], Kornel Labun [ctb]

Maintainer Michal Swirski <michal.swirski@uw.edu.pl>

Contents

`antisense` *Get antisense*

Description

Get antisense

Usage

```
antisense(grl)
```

Value

a GRangesList

`browseRC` *Browse a gene on Ribocrypt webpage*

Description

Can also disply local RiboCrypt app if specified in the 'host' argument.

Usage

```
browseRC(
  symbol = NULL,
  gene_id = NULL,
  tx_id = NULL,
  exp = "all_merged-Homo_sapiens_modalities",
  libraries = NULL,
  leader_extension = 0,
  trailer_extension = 0,
  viewMode = FALSE,
  other_tx = FALSE,
  plot_on_start = TRUE,
  frames_type = "columns",
  kmer = 1,
  add_translons = FALSE,
  zoom_range = NULL,
  host = "https://ribocrypt.org",
  browser = getOption("browser")
)
```

Arguments

`symbol`	gene symbol, default NULL
`gene_id`	gene symbol, default NULL
`tx_id`	gene symbol, default NULL
`exp`	experiment name, default "all_merged-Homo_sapiens_modalities"
`libraries`	NULL, default to first in experiment, c("RFP","RNA") would add RNA to default.
`leader_extension`	integer, default 0. (How much to extend view upstream)
`trailer_extension`	integer, default 0. (How much to extend view downstream)
`viewMode`	FALSE (transcript view), TRUE gives genomic.
`other_tx`	FALSE, show all other annotation in region (isoforms etc.)
`plot_on_start`	logical, default TRUE. Plot gene when opening browser.
`frames_type`	"columns"
`kmer`	integer, default 1 (no binning), binning size of windows, to smear out the signal.
`add_translons`	logical, default FALSE. If TRUE, add translons predicted sequences in annotation.

zoom_range	character, zoom values.
host	url, default "https://ribocrypt.org". Set to localhost for local version.
browser	getOption("browser")

Value

browseURL, opens browse with page

Examples

```
browseRC("ATF4", "ENSG00000128272")
```

---

collection_dir_from_exp

Get collection directory

Description

Get collection directory

Usage

```
collection_dir_from_exp(df, must_exists = FALSE, new_format = TRUE)
```

Arguments

df	ORFik experiment
must_exists	logical, stop if dir does not exists
new_format	logical, TRUE is new or old fst format (FALSE)

Value

file.path(resFolder(df), "collection_tables")

Examples

```
df <- ORFik.template.experiment()
collection_dir_from_exp(df)
```

collection_path_from_exp *Get collection path*

Description

For directory and id, must be fst format file

Usage

```
collection_path_from_exp(
  df,
  id,
  gene_name_list = NULL,
  must_exists = TRUE,
  collection_dir = collection_dir_from_exp(df, must_exists),
  grl_all = loadRegion(df)
)
```

Arguments

df	ORFik experiment
id	character, transcript ids
gene_name_list	a data.table, default NULL, with gene ids
must_exists	logical, stop if dir does not exists
collection_dir	= collection_dir_from_exp(df, must_exists)
grl_all	a GRangesList for new format, what genomic range to get.

Value

file.path(resFolder(df), "collection_tables")

Examples

```
df <- ORFik.template.experiment()
tx_id <- "ENST00000012312"
collection_path_from_exp(df, id = tx_id, must_exists = FALSE)
```

collection_to_wide *Cast a collection table to wide format*

Description

Cast a collection table to wide format

Usage

```
collection_to_wide(table, value.var = "logscore")
```

Arguments

table	a data.table in long format
value.var	which column to use as scores, default "logscore"

Value

a table in wide format

`compute_collection_table` *Get collection table normalized in wide format*

Description

Get collection table normalized in wide format

Usage

```
compute_collection_table(
  path,
  lib_sizes,
  df,
  metadata_field,
  normalization,
  kmer,
  metadata,
  min_count = 0,
  format = "wide",
  value.var = "logscore",
  as_list = FALSE,
  subset = NULL,
  group_on_tx_tpm = NULL,
  split_by_frame = FALSE,
  ratio_interval = NULL,
  decreasing_order = FALSE
)
```

Arguments

path	the path to gene counts
lib_sizes	named integer vector, default NULL. If given will do a pre tpm normalization for full library sizes
df	the ORFik experiment to load the precomputed collection from. It must also have defined runIDs() for all samples.
metadata_field	the column name in metadata, to select to group on.
normalization	a character string, which mode, for options see RiboCrypt:::normalizations
kmer	integer, default 1L (off), if > 1 will smooth out signal with sliding window size kmer.
metadata	a data.table of metadata, must contain the Run column to select libraries.

min_count	integer, default 0. Minimum counts of coverage over transcript to be included.
format	character, default "wide", alternative "long". The format of the table output.
value.var	which column to use as scores, default "logscore"
as_list	logical, default FALSE. Return as list of size 2, count data.table and metadata data.table Set to TRUE if you need metadata subset (needed if you subset the table, to get correct matching)
subset	numeric vector, positional interval to subset, must be <= size of whole region.
group_on_tx_tpm	numeric vector, default NULL. tpm values per libraries. Either for that gene or some other gene.
split_by_frame	logical, default FALSE For kmer sliding window, should it split by frame
ratio_interval	numeric vector of size 2 or 4, default NULL. If 2, means you should sort libraries on coverage in that region. If 4, means to sort on ratio of that region in this gene vs the other region in another gene.
decreasing_order	logical, default FALSE. Sort you ordering vector from lowest (default). If TRUE, sort from highest downwards.

Value

a data.table in long or wide (default) format, if as list, it is a list of size 2 (see argument as_list)

`createSeqPanelPattern` *Create sequence panel for RiboCrypt*

Description

Create sequence panel for RiboCrypt

Usage

```
createSeqPanelPattern(
  sequence,
  start_codons = "ATG",
  stop_codons = c("TAA", "TAG", "TGA"),
  frame = 1,
  custom_motif = NULL
)
```

Arguments

sequence	the DNAStringSet
start_codons	character vector, default "ATG"
stop_codons	character vector, default c("TAA", "TAG", "TGA")
frame	frame not used
custom_motif	character vector, default NULL.

Value

a ggplot object

DEG_plot *Differential expression plots (1D or 2D)*

Description

Gives you interactive 1D or 2D DE plots

Usage

```
DEG_plot(
  dt,
  draw_non_regulated = TRUE,
  add_search_bar = TRUE,
  xlim = ifelse(two_dimensions, "bidir.max", "auto"),
  ylim = "bidir.max",
  xlab = ifelse(two_dimensions, "RNA fold change (log2)", "Mean counts (log2)"),
  ylab = ifelse(two_dimensions, "RFP fold change (log2)", "Fold change (log2)"),
  two_dimensions = ifelse("LFC" %in% colnames(dt), FALSE, TRUE),
 color.values = c(`No change` = "black", Significant = "red", Buffering = "purple",
   `mRNA abundance` = "darkgreen", Expression = "blue", Forwarded = "yellow", Inverse =
    "aquamarine", Translation = "orange4"),
  format = "png"
)
```

Arguments

dt	a data.table with results from a differential expression run. Normally from: ORFik::DTEG.analysis(df1, df2)
draw_non_regulated	logical, default TRUE Should non-regulated rows be included in the plot? Will make the plot faster to render if skipped (FALSE)
add_search_bar	logical, default TRUE. Add a crosstalk search bar to search for genes in the plot
xlim	numeric vector or character preset, default: ifelse(two_dimensions, "bidir.max", "auto") (Equal in both + / - direction, using max value + 0.5 of meanCounts(in 1d) / rna(in 2d) column of dt). If you want ggplot to decide limit, set to "auto". For numeric vector, specify min and max x limit: like c(-5, 5)
ylim	numeric vector or character preset, default: "bidir.max" (Equal in both + / - direction, using max value + 0.5 of LFC(in 1d) / rfp(in 2d) column of dt). If you want ggplot to decide limit, set to "auto". For numeric vector, specify min and max x limit: like c(-5, 5)
xlab	character, default: ifelse(two_dimensions, "RNA fold change (log2)", "Mean counts (log2)")
ylab	character, default: ifelse(two_dimensions, "RFP fold change (log2)", "Fold change (log2)")
two_dimensions	logical, default: ifelse("LFC" %in% colnames(dt), FALSE, TRUE) Is this two dimensional, like: Ribo-seq vs RNA-seq. Alternative, FALSE: Then Log fold change vs mean counts

color.values	named character vector, default: c("No change" = "black", "Significant" = "red","Buffering" = "purple", "mRNA abundance" = "darkgreen","Expression" = "blue", "Forwarded" = "yellow","Inverse" = "aquamarine", "Translation" = "orange4")
format	character, default "png". Format for plotly bar.

Value

plotly object or crosstalk bscols if add_search_bar is TRUE.

Examples

```
# Load experiment
df <- ORFik.template.experiment()
df_rna <- df[df$libtype == "RNA",]
# 1 Dimensional analysis
dt <- DEG.analysis(df_rna)
dt$Regulation[1] <- "Significant" # Fake sig level
DEG_plot(dt, draw_non_regulated = TRUE)
# 2 Dimensional analysis
df_rfp <- df[df$libtype == "RFP",]
dt_2d <- DTEG.analysis(df_rfp, df_rna, output.dir = NULL)
dt_2d$Regulation[4] <- "Translation" # Fake sig level
dt_2d$rfp.lfc[4] <- -0.3 # Fake sig level
dt_2d$Regulation[5] <- "Buffering" # Fake sig level
dt_2d$rna.lfc[5] <- -0.3 # Fake sig level
DEG_plot(dt_2d, draw_non_regulated = TRUE)
# Add Gene symbols in ids for easier analysis
dt_2d_with_gene_ids <- ORFik::append_gene_symbols(dt_2d, symbols(df))
DEG_plot(dt_2d_with_gene_ids, draw_non_regulated = TRUE)
```

---

fetch_JS_seq *Fetch Javascript sequence*

Description

Fetch Javascript sequence

Usage

```
fetch_JS_seq(
  target_seq,
  nplots,
  distance = 50,
  display_dist,
  aa_letter_code = "one_letter",
  input_id
)
```

Arguments

target_seq	the target sequence
nplots	number of plots
distance	numeric, default 50.
display_dist	display distance
aa_letter_code	"one_letter"
input_id	shiny id of the object

Value

a list of 2 lists, the nt list (per frame, total 3) and AA list (per frame, total 3)

---

fetch_summary *Fetch summary of uniprot id*

Description

Fetch summary of uniprot id

Usage

```
fetch_summary(qualifier, provider = "alphafold")
```

Arguments

qualifier	uniprot ids
provider	"pdbe", alternatives: "alphafold", "all"

Value

a character of json

---

geneTrackLayer *How many rows does the gene track need*

Description

How many rows does the gene track need

Usage

```
geneTrackLayer(grl)
```

Arguments

grl	a GRangesList

Value

numeric, the track row index

getCoverageProfile *Get coverage profile*

Description

Get coverage profile

Usage

```
getCoverageProfile(grl, reads, kmers = 1, kmers_type = "mean")
```

Arguments

grl	a GRangesList
reads	GRanges
kmers	1
kmers_type	"mean"

Value

data.table of coverage

getIndexes *Get index*

Description

Get index

Usage

```
getIndexes(ref_granges)
```

Arguments

ref_granges	a GRanges object

Value

integer vector, indices

`get_meta_browser_plot_full` *Full plot for allsamples browser*

Description

Full plot for allsamples browser

Usage

```
get_meta_browser_plot_full(
  m,
  heatmap,
  id,
  df,
  summary = TRUE,
  annotation = TRUE,
  region_type,
  plotType = "plotly",
  tx_annotation,
  display_region,
  cds_annotation,
  viewMode,
  collapse_intron_flank,
  rel_heights = c(0.2, 0.75, 0.05)
)
```

Arguments

`m`	data.table of coverage per sample (wide format)
`heatmap`	ComplexHeatmap object of plot from 'm'
`id`	id of transcript
`df`	ORFik experiment
`summary`	logical, default TRUE (add top plot)
`annotation`	logical, default TRUE (add bottom annotation track)
`region_type`	character, "what is the coverage region?" Usually full mrna: "mrna" or "leader+cds".
`plotType`	= "plotly",
`tx_annotation`	a GRangesList of tx annotation
`display_region`	a GRangesList of display region
`cds_annotation`	a GRangesList of cds annotation
`viewMode`	character, "tx" or "genomic"
`collapse_intron_flank`	integer, if TRUE and viewMode genomic, collapse introns to this max size.
`rel_heights`	numeric < 1, default: c(0.2, 0.75, 0.05). Relative heights, sum to 1 and must be length 3.

Value

a cowplot grub

ggplotlyHover *Call ggplotly with hoveron defined*

Description

Call ggplotly with hoveron defined

Usage

```
ggplotlyHover(x, ...)
```

Arguments

x	a a ggplot argument
...	additional arguments for ggplotly

Value

a ggplotly object

load_collection *Load a ORFik collection table*

Description

Load a ORFik collection table

Usage

```
load_collection(path, grl = attr(path, "range"))
```

Arguments

path	the path to gene counts
grl	a GRangesList, default attr(path, "range"), for new fst format, which range to get.

Value

a data.table in long format

make_rc_url *Create URL to browse a gene on Ribocrypt webpage*

Description

Can also make url for local RiboCrypt app' On the actuall app, the function make_url_from_inputs is used on the shiny reactive input object. This one is for manual use.

Usage

```
make_rc_url(
  symbol = NULL,
  gene_id = NULL,
  tx_id = NULL,
  exp = "all_merged-Homo_sapiens_modalities",
  libraries = NULL,
  leader_extension = 0,
  trailer_extension = 0,
  viewMode = FALSE,
  other_tx = FALSE,
  plot_on_start = TRUE,
  frames_type = "columns",
  kmer = 1,
  add_translons = FALSE,
  zoom_range = NULL,
  host = "https://ribocrypt.org"
)
```

Arguments

symbol	gene symbol, default NULL
gene_id	gene symbol, default NULL
tx_id	gene symbol, default NULL
exp	experiment name, default "all_merged-Homo_sapiens_modalities"
libraries	NULL, default to first in experiment, c("RFP","RNA") would add RNA to default.
leader_extension	integer, default 0. (How much to extend view upstream)
trailer_extension	integer, default 0. (How much to extend view downstream)
viewMode	FALSE (transcript view), TRUE gives genomic.
other_tx	FALSE, show all other annotation in region (isoforms etc.)
plot_on_start	logical, default TRUE. Plot gene when opening browser.
frames_type	"columns"
kmer	integer, default 1 (no binning), binning size of windows, to smear out the signal.
add_translons	logical, default FALSE. If TRUE, add translons predicted sequences in annotation.
zoom_range	character, zoom values.
host	url, default "https://ribocrypt.org". Set to localhost for local version.

Value

character, URL.

Examples

```
make_rc_url("ATF4", "ENSG00000128272")
```

---

matchMultiplePatterns *Match multiple patterns*

Description

Match multiple patterns

Usage

```
matchMultiplePatterns(patterns, Seq)
```

Arguments

patterns	character
Seq	a DNAStringSet

Value

integer vector, indices (named with pattern hit)

---

matchToGRanges *Match to GRanges*

Description

Match to GRanges

Usage

```
matchToGRanges(matches, ref_granges)
```

Arguments

matches	integer vector, indices
ref_granges	GRanges

Value

GRanges object

multiOmicsPlot_animate

Multi-omics animation using list input

Description

The animation will move with a play butten, there is 1 transition per library given.

Usage

```
multiOmicsPlot_animate(
  display_range,
  annotation = display_range,
  reference_sequence,
  reads,
  viewMode = c("tx", "genomic")[1],
  custom_regions = NULL,
  leader_extension = 0,
  trailer_extension = 0,
  withFrames = NULL,
  frames_type = "lines",
  colors = NULL,
  kmers = NULL,
  kmers_type = c("mean", "sum")[1],
  ylabels = NULL,
  lib_to_annotation_proportions = c(0.8, 0.2),
  lib_proportions = NULL,
  annotation_proportions = NULL,
  width = NULL,
  height = NULL,
  plot_name = "default",
  plot_title = NULL,
  display_sequence = c("both", "nt", "aa", "none")[1],
  seq_render_dist = 100,
  aa_letter_code = c("one_letter", "three_letters")[1],
  annotation_names = NULL,
  start_codons = "ATG",
  stop_codons = c("TAA", "TAG", "TGA"),
  custom_motif = NULL,
  AA_code = Biostrings::GENETIC_CODE,
  log_scale = FALSE,
  BPPARAM = BiocParallel::SerialParam(),
  summary_track = FALSE,
  summary_track_type = frames_type,
  export.format = "svg",
  frames_subset = "all"
)
```

Arguments

display_range	the whole region to visualize, a GRangesList or GRanges object

annotation
: the whole annotation which your target region is a subset, a GRangesList or GRanges object

reference_sequence
: the genome reference, a FaFile or FaFile convertible object

reads
: the NGS libraries, as a list of GRanges with or without score column for replicates.

viewMode
: character, default "tx" (transcript coordinates, first position is 1, exons are merged into a single sequence)
Alternative: "genomic" (genomic coordinates, first position is first position in display_range argument. Introns are displayed).

custom_regions
: a GRangesList or NULL, default: NULL. The alternative annotation, like self defined uORFs etc. The vertical annotation bars will have a different color.

leader_extension
: integer, default 0. (How much to extend view upstream)

trailer_extension
: integer, default 0. (How much to extend view downstream)

withFrames
: a logical vector, default NULL. Alternative: a length 1 or same length as list length of "reads" argument.

frames_type
: character, default "lines". Alternative:
- columns
- stacks
- area

colors
: character, default NULL (automatic colouring). If "withFrames" argument is TRUE, colors are set to to c("red", "green", "blue") for the 3 frames. Alternative: Character vector of length 1 or length of "reads" list argument.

kmers
: numeric (integer), bin positions into kmers. Default NULL, which is equal to 1, i.e. no binning.

kmers_type
: character, function used for kmers sliding window. default: "mean", alternative: "sum"

ylabels
: character, default NULL. Name of libraries in "reads" list arugment.

lib_to_annotation_proportions
: numeric vector of length 2. relative sizes of profiles and annotation.

lib_proportions
: numeric vector of length equal to displayed libs. Relative sizes of profiles displayed

annotation_proportions
: numeric vector of length 3 (seq displayed), or 2 (seq not displayed). Relative sizes of annotation tracks.

width
: numeric, default NULL. Width of plot.

height
: numeric, default NULL. Height of plot.

plot_name
: = character, default "default" (will create name from display_range name). Alternative: custom name for region.

plot_title
: character, default NULL. A title for plot.

display_sequence
: character/logical, default c("both","nt", "aa", "none")[1]. If TRUE or "both", display nucleotide and aa sequence in plot.

seq_render_dist
: integer, default 100. The sequences will appear after zooming below this threshold.

aa_letter_code
: character, when set to "three_letters", three letter amino acid code is used. One letter by default.

annotation_names
: character, default NULL. Alternative naming for annotation.

start_codons
: character vector, default "ATG"

stop_codons
: character vector, default c("TAA", "TAG", "TGA")

custom_motif
: character vector, default NULL.

AA_code
: Genetic code for amino acid display. Default is SGC0 (standard: Vertebrate). See `Biostrings::GENETIC_CODE_TABLE` for options. To change to bacterial, do: `Biostrings::getGeneticCode("11")`

log_scale
: logical, default FALSE. Log2 scale the count values, for easier visualization of shapes.

BPPARAM
: how many cores/threads to use? default: `BiocParallel::SerialParam()`. To see number of threads used for multicores, do `BiocParallel::bpparam()$workers`. You can also add a time remaining bar, for a more detailed pipeline.

summary_track
: logical, default FALSE. Display a top track, that is the sum of all tracks.

summary_track_type
: character, default is same as 'frames_type' argument

export.format
: character, default: "svg". alternative: "png". when you click the top right image button export, what should it export as?

frames_subset
: character, default "all". Alternatives: "red", "green", "blue".

Value

the plot object

Examples

```
library(RiboCrypt)
df <- ORFik.template.experiment()[9:10,]
cds <- loadRegion(df, "cds")
mrna <- loadRegion(df, "mrna")
multiOmicsPlot_animate(mrna[1], annotation = cds[1], reference_sequence = findFa(df),
                 frames_type = "columns", leader_extension = 30, trailer_extension = 30,
                   withFrames = c(TRUE, TRUE),
                   reads = outputLibs(df, type = "pshifted", output.mode = "envirlist",
                               naming = "full", BPPARAM = BiocParallel::SerialParam()))
```

---

multiOmicsPlot_list *Multi-omics plot using list input*

Description

Customizable html plots for visualizing genomic data.

Usage

```
multiOmicsPlot_list(
  display_range,
  annotation = display_range,
  reference_sequence,
  reads,
  viewMode = c("tx", "genomic")[1],
  custom_regions = NULL,
  leader_extension = 0,
  trailer_extension = 0,
  withFrames = NULL,
  frames_type = "lines",
  colors = NULL,
  kmers = NULL,
  kmers_type = c("mean", "sum")[1],
  ylabels = NULL,
  lib_to_annotation_proportions = c(0.8, 0.2),
  lib_proportions = NULL,
  annotation_proportions = NULL,
  width = NULL,
  height = NULL,
  plot_name = "default",
  plot_title = NULL,
  display_sequence = c("both", "nt", "aa", "none")[1],
  seq_render_dist = 100,
  aa_letter_code = c("one_letter", "three_letters")[1],
  annotation_names = NULL,
  start_codons = "ATG",
  stop_codons = c("TAA", "TAG", "TGA"),
  custom_motif = NULL,
  AA_code = Biostrings::GENETIC_CODE,
  log_scale = FALSE,
  BPPARAM = BiocParallel::SerialParam(),
  summary_track = FALSE,
  summary_track_type = frames_type,
  export.format = "svg",
  frames_subset = "all"
)
```

Arguments

display_range	the whole region to visualize, a GRangesList or GRanges object
annotation	the whole annotation which your target region is a subset, a GRangesList or GRanges object
reference_sequence	the genome reference, a FaFile or FaFile convertible object
reads	the NGS libraries, as a list of GRanges with or without score column for replicates.
viewMode	character, default "tx" (transcript coordinates, first position is 1, exons are merged into a single sequence)

Alternative: "genomic" (genomic coordinates, first position is first position in display_range argument. Introns are displayed).

custom_regions a GRangesList or NULL, default: NULL. The alternative annotation, like self defined uORFs etc. The vertical annotation bars will have a different color.

leader_extension
integer, default 0. (How much to extend view upstream)

trailer_extension
integer, default 0. (How much to extend view downstream)

withFrames a logical vector, default NULL. Alternative: a length 1 or same length as list length of "reads" argument.

frames_type character, default "lines". Alternative:
- columns
- stacks
- area

colors character, default NULL (automatic colouring). If "withFrames" argument is TRUE, colors are set to to c("red", "green", "blue") for the 3 frames. Alternative: Character vector of length 1 or length of "reads" list argument.

kmers numeric (integer), bin positions into kmers. Default NULL, which is equal to 1, i.e. no binning.

kmers_type character, function used for kmers sliding window. default: "mean", alternative: "sum"

ylabels character, default NULL. Name of libraries in "reads" list arugment.

lib_to_annotation_proportions
numeric vector of length 2. relative sizes of profiles and annotation.

lib_proportions
numeric vector of length equal to displayed libs. Relative sizes of profiles displayed

annotation_proportions
numeric vector of length 3 (seq displayed), or 2 (seq not displayed). Relative sizes of annotation tracks.

width numeric, default NULL. Width of plot.

height numeric, default NULL. Height of plot.

plot_name = character, default "default" (will create name from display_range name). Alternative: custom name for region.

plot_title character, default NULL. A title for plot.

display_sequence
character/logical, default c("both","nt", "aa", "none")[1]. If TRUE or "both", display nucleotide and aa sequence in plot.

seq_render_dist
integer, default 100. The sequences will appear after zooming below this threshold.

aa_letter_code character, when set to "three_letters", three letter amino acid code is used. One letter by default.

annotation_names
character, default NULL. Alternative naming for annotation.

start_codons character vector, default "ATG"

stop_codons character vector, default c("TAA", "TAG", "TGA")

custom_motif	character vector, default NULL.
AA_code	Genetic code for amino acid display. Default is SGC0 (standard: Vertebrate). See Biostrings::GENETIC_CODE_TABLE for options. To change to bacterial, do: Biostrings::getGeneticCode("11")
log_scale	logical, default FALSE. Log2 scale the count values, for easier visualization of shapes.
BPPARAM	how many cores/threads to use? default: BiocParallel::SerialParam(). To see number of threads used for multicores, do BiocParallel::bpparam()$workers. You can also add a time remaining bar, for a more detailed pipeline.
summary_track	logical, default FALSE. Display a top track, that is the sum of all tracks.
summary_track_type	character, default is same as 'frames_type' argument
export.format	character, default: "svg". alternative: "png". when you click the top right image button export, what should it export as?
frames_subset	character, default "all". Alternatives: "red", "green", "blue".

Value

the plot object

Examples

```
library(RiboCrypt)
df <- ORFik.template.experiment()[9:10,]
cds <- loadRegion(df, "cds")
mrna <- loadRegion(df, "mrna")
multiOmicsPlot_list(mrna[1], annotation = cds[1], reference_sequence = findFa(df),
                  frames_type = "columns", leader_extension = 30, trailer_extension = 30,
                    reads = outputLibs(df, type = "pshifted", output.mode = "envirlist",
                                naming = "full", BPPARAM = BiocParallel::SerialParam()))
```

---

multiOmicsPlot_ORFikExp

Multi-omics plot using ORFik experiment input

Description

Customizable html plots for visualizing genomic data.

Usage

```
multiOmicsPlot_ORFikExp(
  display_range,
  df,
  annotation = "cds",
  reference_sequence = findFa(df),
 reads = outputLibs(df, type = "pshifted", output.mode = "envirlist", naming = "full",
    BPPARAM = BiocParallel::SerialParam()),
  viewMode = c("tx", "genomic")[1],
  custom_regions = NULL,
```

```
  leader_extension = 0,
  trailer_extension = 0,
 withFrames = libraryTypes(df, uniqueTypes = FALSE) %in% c("RFP", "RPF", "LSU", "TI"),
  frames_type = "lines",
  colors = NULL,
  kmers = NULL,
  kmers_type = c("mean", "sum")[1],
  ylabels = bamVarName(df),
  lib_to_annotation_proportions = c(0.8, 0.2),
  lib_proportions = NULL,
  annotation_proportions = NULL,
  width = NULL,
  height = NULL,
  plot_name = "default",
  plot_title = NULL,
  display_sequence = c("both", "nt", "aa", "none")[1],
  seq_render_dist = 100,
  aa_letter_code = c("one_letter", "three_letters")[1],
  annotation_names = NULL,
  start_codons = "ATG",
  stop_codons = c("TAA", "TAG", "TGA"),
  custom_motif = NULL,
  log_scale = FALSE,
  BPPARAM = BiocParallel::SerialParam(),
  input_id = "",
  summary_track = FALSE,
  summary_track_type = frames_type,
  export.format = "svg",
  frames_subset = "all"
)
```

Arguments

display_range	the whole region to visualize, a GRangesList or GRanges object
df	an ORFik experiment or a list containing ORFik experiments. Usually a list when you have split Ribo-seq and RNA-seq etc.
annotation	the whole annotation which your target region is a subset, a GRangesList or GRanges object
reference_sequence	the genome reference, default ORFik::findFa(df)
reads	the NGS libraries, as a list of GRanges with or without 'score' column for replicates. Can also be a covRle object of precomputed coverage. Default: `outputLibs(df, type = "pshifted", output.mode = "envirlist",naming = "full", BPPARAM = BiocParallel::SerialParam())`
viewMode	character, default "tx" (transcript coordinates, first position is 1, exons are merged into a single sequence) Alternative: "genomic" (genomic coordinates, first position is first position in display_range argument. Introns are displayed).
custom_regions	a GRangesList or NULL, default: NULL. The alternative annotation, like self defined uORFs etc. The vertical annotation bars will have a different color.

`leader_extension`
: integer, default 0. (How much to extend view upstream)

`trailer_extension`
: integer, default 0. (How much to extend view downstream)

`withFrames`
: a logical vector, default `libraryTypes(df, uniqueTypes = FALSE) %in% c("RFP", "RPF", "LSU", "TI")` Alternative: a length 1 or same length as list length of "reads" argument.

`frames_type`
: character, default "lines". Alternative:
- columns
- stacks
- area

`colors`
: character, default NULL (automatic colouring). If "withFrames" argument is TRUE, colors are set to to c("red", "green", "blue") for the 3 frames. Alternative: Character vector of length 1 or length of "reads" list argument.

`kmers`
: numeric (integer), bin positions into kmers. Default NULL, which is equal to 1, i.e. no binning.

`kmers_type`
: character, function used for kmers sliding window. default: "mean", alternative: "sum"

`ylabels`
: character, default `bamVarName(df)`. Name of libraries in "reads" list argument.

`lib_to_annotation_proportions`
: numeric vector of length 2. relative sizes of profiles and annotation.

`lib_proportions`
: numeric vector of length equal to displayed libs. Relative sizes of profiles displayed

`annotation_proportions`
: numeric vector of length 3 (seq displayed), or 2 (seq not displayed). Relative sizes of annotation tracks.

`width`
: numeric, default NULL. Width of plot.

`height`
: numeric, default NULL. Height of plot.

`plot_name`
: character, default "default" (will create name from display_range name).

`plot_title`
: character, default NULL. A title for plot.

`display_sequence`
: character/logical, default `c("both","nt", "aa", "none")[1]`. If TRUE or "both", display nucleotide and aa sequence in plot.

`seq_render_dist`
: integer, default 100. The sequences will appear after zooming below this threshold.

`aa_letter_code`
: character, when set to "three_letters", three letter amino acid code is used. One letter by default.

`annotation_names`
: character, default NULL. Alternative naming for annotation.

`start_codons`
: character vector, default "ATG"

`stop_codons`
: character vector, default c("TAA", "TAG", "TGA")

`custom_motif`
: character vector, default NULL.

`log_scale`
: logical, default FALSE. Log2 scale the count values, for easier visualization of shapes.

BPPARAM	how many cores/threads to use? default: BiocParallel::SerialParam(). To see number of threads used for multicores, do BiocParallel::bpparam()$workers. You can also add a time remaining bar, for a more detailed pipeline.
input_id	character path, default: "", id for shiny to disply structures, should be "" for local users.
summary_track	logical, default FALSE. Display a top track, that is the sum of all tracks.
summary_track_type	character, default is same as 'frames_type' argument
export.format	character, default: "svg". alternative: "png". when you click the top right image button export, what should it export as?
frames_subset	character, default "all". Alternatives: "red", "green", "blue".

Value

the plot object

Examples

```
library(RiboCrypt)
df <- ORFik.template.experiment()[9,] #Use third library in experiment only
cds <- loadRegion(df, "cds")
multiOmicsPlot_ORFikExp(extendLeaders(extendTrailers(cds[1], 30), 30), df,
                        frames_type = "columns")
```

---

normalize_collection *Normalize collection table*

Description

Normalize collection table

Usage

```
normalize_collection(
  table,
  normalization,
  lib_sizes = NULL,
  kmer = 1L,
  add_logscore = TRUE,
  split_by_frame = FALSE
)
```

Arguments

table	a data.table in long format
normalization	a character string, which mode, for options see RiboCrypt:::normalizations
lib_sizes	named integer vector, default NULL. If given will do a pre tpm normalization for full library sizes
kmer	integer, default 1L (off), if > 1 will smooth out signal with sliding window size kmer.
add_logscore	logical, default TRUE, adds a log(score + 1) to table
split_by_frame	logical, default FALSE For kmer sliding window, should it split by frame

Value

a data.table of normalized results

---

organism_input_select *Select box for organism*

Description

Select box for organism

Usage

```
organism_input_select(genomes, ns)
```

Arguments

genomes	name of genomes, returned from list.experiments()
ns	the ID, for shiny session

Value

selectizeInput object

RiboCrypt_app *Create RiboCrypt app*

Description

Create RiboCrypt app

Usage

```
RiboCrypt_app(
  validate.experiments = TRUE,
  options = list(launch.browser = ifelse(interactive(), TRUE, FALSE)),
  all_exp = list.experiments(validate = validate.experiments),
  browser_options = c(),
  init_tab_focus = "browser",
  metadata = NULL,
  all_exp_meta = all_exp[grep("all_samples-", name), ]
)
```

Arguments

validate.experiments	logical, default TRUE, set to FALSE to allow starting the app with malformed experiments, be careful will crash if you try to load that experiment!
options	list of arguments, default `list("launch.browser" = ifelse(interactive(), TRUE, FALSE))`
all_exp	a data.table, default: `list.experiments(validate = validate.experiments)`. Which experiments do you want to allow your app to see, default is all in your system config path.
browser_options	named character vector of browser specific arguments: - default_experiment : Which experiment to select, default: first one - default_gene : Which genes to select, default: first one - default_isoform : Which isoform to select, default: first one - default_libs : Which libraries to select: first one, else a single string, where libs are seperated by ",", like "RFP_WT_r1|RFP_WT_r2". Also support run ids (SRR... etc) - default_kmer : K-mer windowing size, default: 1 - default_frame_type : Ribo-seq line type, default: "lines" - default_view_mode : "tx", alternative "genomic" - default_experiment_meta : Which experiment to select for meta analysis, default: first one - default_gene_meta : Which genes to select for meta analysis, default: first one - default_isoform_meta : Which isoform to select for meta analysis, default: first one - translons : Use translon annotation, default "FALSE" - plot_on_start : Plot when starting, default: "FALSE" - hide_settings : Hide settings bar in browser on start, default "TRUE"
init_tab_focus	character, default "browser". Which tab to open on init.
metadata	a path to csv or a data.table of metadata columns, must contain a "Run" column to merge IDs to ORFik experiments. It is used in the metabrowser tab for grouping of samples.
all_exp_meta	a data.table, default: `all_exp[grep("all_samples-", name),]`. Can also be NULL, to ignore the metabrowser completly. It is the subset of all_exp which are collections (the set of all experiments per organism), this will be fed to the metabrowser, while remaining all_exp are used in all other modules.

Value

RiboCrypt shiny app

Examples

```
run_variable <- 1 # Ignore check test limit
## Default run
# RiboCrypt_app()
## Plot on start
# RiboCrypt_app(browser_options = c(plot_on_start = "TRUE"))
## Init with an experiment and gene (you must of course have the experiment)

#RiboCrypt_app(validate.experiments = FALSE,
#       browser_options = c(plot_on_start = "TRUE",
```

```
#                           default_experiment = "all_merged-Homo_sapiens_2024_8",
#                           default_gene = "ATF4-ENSG00000128272"))
#RiboCrypt_app(validate.experiments = FALSE, all_exp = all_exp,
#browser_options = c(plot_on_start = "TRUE",
#                    default_experiment = "human_all_merged_l50",
#                    default_gene = "RPL12-ENSG00000197958",
#                    default_isoform = "ENST00000361436",
#                    default_view_mode = "genomic"))
#RiboCrypt_app(validate.experiments = FALSE,
#       browser_options = c(plot_on_start = "TRUE",
#                           default_experiment = "all_merged-Saccharomyces_cerevisiae",
#                           default_gene = "EFM5-YGR001",
#                           default_view_mode = "genomic"))
```

`trimOverlaps` *Trim overlaps*

Description

Trim overlaps

Usage

```
trimOverlaps(overlaps, display_range)
```

Arguments

`overlaps`	GRanges
`display_range`	GRanges

Value

GRanges

Index