Package ‘RiboCrypt’

February 1, 2024

Type Package
Title Interactive visualization in genomics
Version 1.8.0
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, data.table,
   dplyr, GenomeInfoDb, GenomicFeatures, GenomicRanges, ggplot2,
   htmlwidgets, httr, IRanges, jsonlite, knitr, markdown,
   NGLVieweR, plotly, rlang, R Curl, shiny, shinyCSSloaders,
   shinyhelper, shinyjqui, stringr
Suggests testthat, rmarkdown, BiocStyle, BSgenome,
   BSgenome.Hsapiens.UCSC.hg19
RoxygenNote 7.2.3
VignetteBuilder knitr
git_url https://git.bioconductor.org/packages/RiboCrypt
git_branch RELEASE_3_18
git_last_commit f2d4217
git_last_commit_date 2023-10-24
Repository Bioconductor 3.18
Date/Publication 2024-01-31
Description

Get antisense

Usage

antisense(grl)

Value

a GRangesList
createSeqPanelPattern  

Create sequence panel for RiboCrypt

**Description**

Create sequence panel for RiboCrypt

**Usage**

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

**Arguments**

- `start_codons` character vector, default "ATG"
- `stop_codons` character vector, default c("TAA", "TAG", "TGA")
- `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**

```r
DEG_plot(
  dt,
  draw_non_regulated = FALSE,
  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",
```

```r
```
```r```


mRNA abundance = "darkgreen", Expression = "blue", Forwarded = "yellow", Inverse = "aquamarine", Translation = "orange4")

Arguments

- **dt**: a data.table with results from a differential expression run. Normally from: ORFik::DTEG.analysis(df1, df2)
- **draw_non_regulated**: logical, default FALSE. Should non-regulated rows be included in the plot? Will make the plot faster to render if skipped (FALSE)
- **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")

Value

- plotly object

Examples

```r
# Load experiment
df <- ORFik.template.experiment()

# 1 Dimensional analysis
dt <- DEG.analysiss(df[df$libtype == "RNA",])
dt$Regulation[1] <- "Significant" # Fake sig level
DEG_plot(dt, draw_non_regulated = TRUE)

# 2 Dimensional analysis
dt_2d <- DTEG.analysiss(df[df$libtype == "RFP",], df[df$libtype == "RNA",],
output.dir = NULL)
dt_2d$Regulation[4] <- "Translation" # Fake sig level
dt_2d$Regulation[5] <- "Buffering" # Fake sig level
DEG_plot(dt_2d, draw_non_regulated = TRUE)
```


distanceToFollowing  

**Description**

Distance to following range

**Usage**

distanceToFollowing(grl, grl2 = grl, ignore.strand = FALSE)

**Arguments**

- `grl`  
a GRangesList
- `grl2`  
a GRangesList, default 'grl'
- `ignore.strand`  
logical, default FALSE

**Value**

numeric vector of distance

---

fetch_JS_seq  

**Description**

Fetch Javascript sequence

**Usage**

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

**Arguments**

- `target_seq`  
the target sequence
- `nplots`  
number of plots
- `distance`  
numeric, default 50.
- `display_dist`  
display distance
- `aa_letter_code`  
"one_letter"
**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

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

Description
Get index

Usage
getIndexes(ref_granges)

Arguments
- ref_granges: a GRanges object

Value
integer vector, indices
ggplotlyHover  \textit{Call ggplotly with hoveron defined}

\textbf{Description}

Call \texttt{ggplotly} with hoveron defined

\textbf{Usage}

\texttt{ggplotlyHover(x, \ldots)}

\textbf{Arguments}

\begin{itemize}
  \item \texttt{x} \hspace{0.5cm} a \texttt{ggplot} argument
  \item \texttt{\ldots} \hspace{0.5cm} additional arguments for \texttt{ggplotly}
\end{itemize}

\textbf{Value}

a \texttt{ggplotly} object

matchMultiplePatterns  \textit{Match multiple patterns}

\textbf{Description}

Match multiple patterns

\textbf{Usage}

\texttt{matchMultiplePatterns(patterns, Seq)}

\textbf{Arguments}

\begin{itemize}
  \item \texttt{patterns} \hspace{0.5cm} character
  \item \texttt{Seq} \hspace{0.5cm} a \texttt{DNAStringSet}
\end{itemize}

\textbf{Value}

integer vector, indices (named with pattern hit)
**matchToGRanges**

**Description**
Match to GRanges

**Usage**

```r
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 button, there is 1 transition per library given.

**Usage**

```r
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,
BPPARAM = BiocParallel::SerialParam()
)

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

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,
# reads = outputLibs(df, type = "pshifted", output.mode = "envirlist",
# naming = "full", BPPARAM = BiocParallel::SerialParam()))
Description

Customizable html plots for visualizing genomic data.

Usage

```r
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,
    BPPARAM = BiocParallel::SerialParam(),
    summary_track = FALSE,
    summary_track_type = frames_type,
    export.format = "svg"
)
```

Arguments

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

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 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.

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

ylabels character, default NULL. 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). Alternative: custom name for region.

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

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")

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?

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()))
Description

Customizable html plots for visualizing genomic data.

Usage

```r
multiOomicsPlot_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"),
  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,
  BPPARAM = BiocParallel::SerialParam(),
  input_id = "",
  summary_track = FALSE,
  summary_track_type = frames_type,
  export.format = "svg"
)
```
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") 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.

colors_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.
multiOmicsPlot_ORFikExp

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.
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 display 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?

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 = df,
frames_type = "columns")
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"
)

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))
**trimOverlaps**

**Description**

Trim overlaps

**Usage**

`trimOverlaps(overlaps, display_range)`

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>overlaps</td>
<td>GRanges</td>
</tr>
<tr>
<td>display_range</td>
<td>GRanges</td>
</tr>
</tbody>
</table>
trimOverlaps

Value
  GRanges
Index

* internal
  - antisense, 2
  - createSeqPanelPattern, 3
  - geneTrackLayer, 6
  - getCoverageProfile, 7
  - getIndexes, 7
  - ggplotlyHover, 8
  - matchMultiplePatterns, 8
  - matchToGRanges, 9
  - trimOverlaps, 19

antisense, 2
createSeqPanelPattern, 3
DEG_plot, 3
distanceToFollowing, 5

eperiment, 16

FaFile, 10, 13
fetch_JS_seq, 5
fetch_summary, 6

geneTrackLayer, 6
getCoverageProfile, 7
getIndexes, 7
ggplotlyHover, 8
GRanges, 10, 12, 13, 16
GRangesList, 10, 12, 13, 16

matchMultiplePatterns, 8
matchToGRanges, 9
multiOmicsPlot_animate, 9
multiOmicsPlot_list, 12
multiOmicsPlot_ORFikExp, 15

organism_input_select, 18
RiboCrypt_app, 18
trimOverlaps, 19