Package 'SaTAnn'

July 19, 2019

Title Splice-Aware Translatome Annotation

Version 0.99.0

Description
SaTAnn is a method that annotates and quantifies translation at the single ORF level using Ribo-seq data.

Depends rtracklayer, BSgenome, devtools, Biostrings, GenomicFeatures, foreach, doMC, multitaper, GenomicAlignments, GenomicFiles, reshape2, ggplot2, cowplot, grid, BiocGenerics, knitr, gridExtra, rmarkdown

License GPL-3 or above

Encoding UTF-8

LazyData FALSE

Name SaTAnn

biocViews RiboSeq, GenomeAnnotation, Transcriptomics, Software

RoxygenNote 6.1.1

NeedsCompilation no

Author Lorenzo Calviello [aut, cre], Uwe Ohler [rev, fnd]

Maintainer Lorenzo Calviello <calviello.l.bio@gmail.com>

R topics documented:

`annotate_ORFs` *Annotate detected ORFs in transcript and genome space*

Description

This function annotates quantified ORFs with respect to other detected ORFs and annotated ones, in both genome and transcript space.

Usage

```
annotate_ORFs(results_ORFs, Annotation, genome_sequence, region,
  genetic_code)
```

Arguments

`results_ORFs`	Full list of detected ORFs, from `select_quantify_ORFs`
`Annotation`	Rannot object containing annotation of CDS and transcript structures (see `prepare_annotation_files`
`genome_sequence`	BSgenome object
`region`	genomic region being analyzed
`genetic_code`	GENETIC_CODE table to use

Details

As multiple transcripts can contain the same ORF, all the transcript and transcript biotypes are indicated, with a preference for protein_coding transcripts in the "compatible" columns (to be conservative when assessing translation of non-protein coding transcripts). Such compatibility is also output considering the most upstream start codon for that ORF.
Splice features of each orf is annotated with respect to the longest coding transcripts and to the highest translated ORF in that gene.
Variants in N or C terminus of the translated proteins are also indicated (Beta).
ORF annotation with respect to the annotated transcript is also indicated, as follows:

novel: no ORF annotated in the transcript.
ORF_annotated: same exact ORF as annotated.
N_extension: N terminal extension.
N_truncation: N terminal extension.
uORF: upstream ORF.
overl_uORF: upstream overlappin uORF.
NC_extension: N and C termini extension.
dORF: downstream ORF.
overl_dORF: downstream overlapping ORF.
nested_ORF: nested ORF.
C_truncation: C terminal truncation.
C_extension: C terminal extension.

As transcipt-specific annotation can be misleading due to a plethora of different transcripts, it is important to distinguish ORFs also on the basis of their overlap with know CDS regions. ORF annotation with respect to the entire set of CDS exon for the analyzed genomic regions is indicated as follows:

novel: No CDS region is annotated in the entire region.
novel_Upstream: ORF is upstream of annotated CDS regions (does not overlap).
novel_Downstream: ORF is downstream of annotated CDS regions (does not overlap).
novel_Internal: genomic location of the ORF is present between the start of the first, and the end of the last CDS region (does not overlap).
exact_start_stop: Same start and end locations.
Alt5_start: Different start region, upstream.
Alt3_start: Different start region, downstream.
Alt5_stop: Different end region, upstream.
Alt3_stop: Different end region, downstream.

Another layer of annotation is performed by checking the position of the ORF stop codon with respect to the last exon-exon junction.

Value

Exon structure of detected ORF including possible missing exons from reference, together with a spl_type column including the annotation for each exon (e.g. alternative acceptors or donor).

Additional columns are added to the ORFs_tx object:
compatible_with: Set of transcript ids possibly containing the entire ORF structure.
compatible_biotype: Compatible transcript biotype; if a protein coding transcript can contain the ORF, this is set to protein_coding.
compatible_tx: One selected compatible transcript (preference if protein_coding).
compatible_ORF_id_tr: ORF_id_tr id if selecting the compatible transcript.
compatible_with_longest: Same as compatible_with but using the most upstream start codon.
compatible_ORF_id_tr_longest: Same as compatible_ORF_id_tr but using the most upstream start codon .
ref_id: transcript_id of the transcript used to annotate splicing (longest) .
ref_id_maxORF: ORF_id_tr of the ORF used to annotated splicing (most translated of the gene).
NC_protein_isoform: Annotation of possible N or C termini variant (when transcript is pro-

tein_coding) .
ORF_category_Tx: ORF annotation with respect to ORF position in the transcript .
ORF_category_Tx_compatible: ORF annotation with respect to ORF position in the transcript, using the compatible_ORF_id_tr .
ORF_category_Gen: ORF annotation with respect to its genomic position .
NMD_candidate: TRUE or FALSE, depending on the presence of an additional exon-exon junction downstream the stop codon.
Distance_to_lastExEx: Distance (in nt) between the last exon-exon junction and the stop codon.

Author(s)

Lorenzo Calviello, <calviello.l.bio@gmail.com>

See Also

select_quantify_ORFs, annotate_splicing

---

annotate_splicing *Annotate splice features of detected ORFs*

Description

This function detects usage of different exons and exonic boundaries of one ORF with respect to a reference ORF.

Usage

```
annotate_splicing(orf_gen, ref_cds)
```

Arguments

orf_gen	Exon structure of a detected ORF
ref_cds	Exon structure of a reference ORF

Details

each exon is aligned to the closest one to match acceptor and donor sites, or to annotate missing exons. 5ss and 3ss indicate exon 5' and 3', respectively. CDS_spanning indicates retained intron; missing_CDS indicates no overlapping exon (missed or included); monoCDS indicates a single-exon ORF; firstCDS and lastCDS indicate first CDS exon or last CDS exon.

Value

Exon structure of detected ORF including possible missing exons from reference, together with a spl_type column including the annotation for each exon (e.g. alternative acceptors or donor).

Author(s)

Lorenzo Calviello, <calviello.l.bio@gmail.com>

See Also

detect_translated_orfs, annotate_ORFs

---

calc_orf_pval *Collect ORF Ribo-seq statistics*

Description

This function calculates statistics for the analysis of P_sites profiles for each ORF

Usage

```
calc_orf_pval(ORFs, P_sites_rle, P_sites_uniq_rle, P_sites_uniq_mm_rle,
  cutoff = 0.5, tapers = 24, bw = 12)
```

Arguments

ORFs	Set of detected ORFs
P_sites_rle	Rle signal of P_sites along the transcript
P_sites_uniq_rle	Rle signal of uniquely mapping P_sites along the transcript
P_sites_uniq_mm_rle	Rle signal of uniquely mapping P_sites with mismatches along the transcript
cutoff	cutoff of average in-frame signal for each codon in the ORF. Defaults to .5
tapers	Number of tapers to use in the multitaper analysis. Defaults to 24
bw	time_bw parameter to use in the multitaper analysis. Defaults to 12

Details

Number of P_sites (uniquely mapping or all), frame percentage and multitaper test statistics are collected for each ORF. The parameter space for the multitaper analysis was explored in the RiboTaper paper.

Value

Set of detected ORFs, including info about the possible longest ORF for that frame.

Author(s)

Lorenzo Calviello, <calviello.l.bio@gmail.com>

See Also

detect_translated_orfs, get_orfs, take_Fvals_spect

`create_SaTAnn_html_report` *Create an html report summarizing SaTAnn results*

Description

This function creates an html report showing summary statistics for SaTAnn-detected ORFs.

Usage

```
create_SaTAnn_html_report(input_files, input_sample_names, output_file)
```

Arguments

`input_files`	Character vector with full paths to plot files (*SaTAnn_plots_RData) generated with `plot_SaTAnn_results`. Must be of same length as `input_sample_names`.
`input_sample_names`	Character vector containing input names. Must be of same length as `input_files`.
`output_file`	String; full path to html report file.

Details

This function creates the html report visualizing final SaTAnn results.

Input are two lists of the same length:

a) `input_files`: list of full paths to one or multiple input files (*SaTAnn_plots_RData files generated with `plot_SaTAnn_results`) and

b) `input_sample_names`: list of corresponding names describing the file content (these are used as names in the report).

For the report, a RMarkdown file is rendered as html document, saved as `output_file`.

Value

The function saves the html report file with the file path `output_file`.

Author(s)

Lorenzo Calviello, <calviello.bio@gmail.com>

See Also

`plot_SaTAnn_results`, `run_SaTAnn`

`detect_readthrough` *Analyzed translation on possible readthrough regions (beta)*

Description

This function uses the multitaper method to look for readthrough translation

Usage

```
detect_readthrough(results_orf, P_sites, P_sites_uniq, P_sites_uniq_mm,
  genome_sequence, annotation, genetic_code_table, cutoff_fr_ave = 0.5)
```

Arguments

`results_orf`	Full list of detected ORFs, from `select_quantify_ORFs` and `annotate_ORFs`
`P_sites`	GRanges object with P_sites positions
`P_sites_uniq`	GRanges object with uniquely mapping P_sites positions
`P_sites_uniq_mm`	Rle signal of uniquely mapping P_sites with mismatches along the transcript
`genome_sequence`	BSgenome object
`annotation`	Rannot object containing annotation of CDS and transcript structures (see `prepare_annotation_files`)
`genetic_code_table`	GENETIC_CODE table to use
`cutoff_fr_ave`	`cutoff` parameter for the `calc_orf_pval` functions

Details

The function looks for stop-stop pairs after the stop codon of the detected ORF

Value

GRanges object with the set of translated readthrough regions

Author(s)

Lorenzo Calviello, <calviello.l.bio@gmail.com>

See Also

`detect_translated_orfs`, `select_quantify_ORFs`, `annotate_ORFs`, `get_reathr_seq`

detect_translated_orfs *Detect actively translated ORFs*

Description

This function detects translated ORFs

Usage

```
detect_translated_orfs(selected_txs, genome_sequence, annotation, P_sites,
  P_sites_uniq, P_sites_uniq_mm, genomic_region, genetic_code,
  all_starts = T, nostarts = F, start_sel_cutoff = NA,
  start_sel_cutoff_ave = 0.5, cutoff_fr_ave = 0.5)
```

Arguments

selected_txs	set of selected transcripts, output from select_txs
genome_sequence	BSgenome object
annotation	Rannot object containing annotation of CDS and transcript structures (see prepare_annotation_files)
P_sites	GRanges object with P_sites positions
P_sites_uniq	GRanges object with uniquely mapping P_sites positions
P_sites_uniq_mm	GRanges object with uniquely mapping (with mismatches) P_sites positions
genomic_region	GRanges object with genomic coordinates of the genomic region analyzed
genetic_code	GENETIC_CODE table to use
all_starts	get_all_starts parameter for the get_orfs function
nostarts	Stop_Stop parameter for the get_orfs function
start_sel_cutoff	cutoff parameter for the select_start function
start_sel_cutoff_ave	cutoff_ave parameter for the select_start function
cutoff_fr_ave	cutoff parameter for the calc_orf_pval functions

Details

A set of transcripts, together with genome sequence and Ribo-signal are analyzed to extract translated ORFs

Value

A list with transcript coordinates, exonic coordinates and statistics for each ORF exonic bin and junction(from `select_txs`).

The value for each column is as follows:

`ave_pct_fr`: average percentage of in-frame reads for each codon in the ORF `pct_fr`: percentage of in-frame reads in the ORF `ave_pct_fr`: average percentage of in-frame reads for each codon in the ORF `ave_pct_fr_st`: average percentage of in-frame reads per each codon between the selected start codon and the next candidate one `pct_fr_st`: percentage of in-frame reads between the selected start codon and the next candidate one `longest_ORF`: GRanges coordinates for the longest ORF with the same stop codon `pval`: P-value for the multitaper F-test at 1/3 using the ORF P_sites profile `pval_uniq`: P-value for the multitaper F-test at 1/3 using the ORF P_sites profile (only uniquely mapping reads) `P_sites_raw`: Raw number of P_sites mapping to the ORF `P_sites_raw_unique`: Uniquely mapping P_sites mapping to the ORF `ORF_id_tr`: ORF id containing <tx_id>_<start>_<end> `Protein`: AAString sequence of the translated protein `region`: Genomic coordinates of the analyzed region `gene_id`: gene_id for the corresponding analyzed transcript `gene_biotype`: gene biotype for the corresponding analyzed transcript `gene_name`: gene name for the corresponding analyzed transcript `transcript_id`: transcript_id for the corresponding analyzed ORF `transcript_biotype`: transcript biotype for the corresponding analyzed ORF

Author(s)

Lorenzo Calviello, <calviello.l.bio@gmail.com>

See Also

`select_txs`, `get_orfs`, `take_Fvals_spect`, `select_start`, `prepare_annotation_files`

---

from_tx_togen *Map transcript coordinates to genomic coordinates*

Description

This function uses the `mapFromTranscripts` function to switch between transcript and genomic coordinates

Usage

```
from_tx_togen(ORFs, exons, introns)
```

Arguments

`ORFs`	Set of detected ORFs from the `calc_orf_pval` function
`exons`	exonic regions of the analyzed transcripts, as a GRangesList object
`introns`	intronic regions of the analyzed transcripts, as a GRangesList object

Value

exonic coordinates for each ORF.

Author(s)

Lorenzo Calviello, <calviello.l.bio@gmail.com>

See Also

mapFromTranscripts

---

get_orfs *Find ATG-starting ORFs in a sequence*

Description

This function loads the annotation created by the prepare_annotation_files function

Usage

```
get_orfs(tx_name, sequence, get_all_starts = T, Stop_Stop = F,
  scores = c(1, 0.5), genetic_code_table)
```

Arguments

tx_name	transcript_id
sequence	DNAString object containing the sequence of the transcript
get_all_starts	Output all possible start codons? Defaults to TRUE
Stop_Stop	Find Stop-Stop pairs (no defined start codon)? Defaults to FALSE
scores	Deprecated
genetic_code_table	GENETIC_CODE table to use

Value

GRanges object containing coordinates for the detected ORFs

Author(s)

Lorenzo Calviello, <calviello.l.bio@gmail.com>

See Also

detect_translated_orfs

get_ps_fromsplicemin *Offset spliced reads on minus strand*

Description

This function calculates P-sites positions for spliced reads on the minus strand

Usage

```
get_ps_fromsplicemin(x, cutoff)
```

Arguments

x	a GAlignments object with a cigar string
cutoff	number representing the offset value

Value

a GRanges object with offset reads

Author(s)

Lorenzo Calviello, <calviello.l.bio@gmail.com>

See Also

prepare_for_SaTAnn

get_ps_fromspliceplus *Offset spliced reads on plus strand*

Description

This function calculates P-sites positions for spliced reads on the plus strand

Usage

```
get_ps_fromspliceplus(x, cutoff)
```

Arguments

x	a GAlignments object with a cigar string
cutoff	number representing the offset value

Value

a GRanges object with offset reads

Author(s)

Lorenzo Calviello, <calviello.l.bio@gmail.com>

See Also

prepare_for_SaTAnn

---

get_reathr_seq *Extract possible readthrough sequences (beta)*

Description

This function extracts readthrough regions for subsequent analysis

Usage

```
get_reathr_seq(tx_name, orf, sequence, genetic_code)
```

Arguments

tx_name	transcript_id
orf	transcript-level ORF coordinates
sequence	DNAString object containing the sequence of the transcript
genetic_code	GENETIC_CODE table to use

Details

The function looks for stop-stop pairs after the stop codon of the detected ORF

Value

GRanges object with the set of possible readthrough sequences

Author(s)

Lorenzo Calviello, <calviello.l.bio@gmail.com>

See Also

detect_translated_orfs, select_quantify_ORFs

load_annotation *Load genomic features and genome sequence*

Description

This function loads the annotation created by the prepare_annotation_files function

Usage

```
load_annotation(path)
```

Arguments

path	Full path to the *Rannot R file in the annotation directory used in the prepare_annotation_files funct

Value

introduces a GTF_annotation object and a genome_seq object in the parent environment

Author(s)

Lorenzo Calviello, <calviello.l.bio@gmail.com>

See Also

prepare_annotation_files

plot_SaTAnn_results *Plot general statistics about SaTAnn results*

Description

This function produces a series of plots and statistics about the set ORFs called by SaTAnn compared to the annotation. IMPORTANT: Use only on transcriptome-wide SaTAnn results. See run_SaTAnn

Usage

```
plot_SaTAnn_results(for_SaTAnn_file, SaTAnn_output_file, annotation_file,
  coverage_file_plus = NA, coverage_file_minus = NA,
  output_plots_path = NA, prefix = NA)
```

Arguments

`for_SaTAnn_file`
: path to the "for_SaTAnn" file containing P_sites positions and junction reads

`SaTAnn_output_file`
: Full path to the "_final_SaTAnn_results" RData object output by SaTAnn. See `run_SaTAnn`

`annotation_file`
: Full path to the *Rannot R file in the annotation directory used in the `prepare_annotation_files funct`

`coverage_file_plus`
: Full path to a Ribo-seq coverage (no P-sites but read coverage) bigwig file (plus strand), as the ones created by `RiboseQC`

`coverage_file_minus`
: Full path to a Ribo-seq coverage (no P-sites but read coverage) bigwig file (minus strand), as the ones created by `RiboseQC`

`output_plots_path`
: Full path to the directory where plots in .pdf format are stored.

`prefix`
: prefix appended to output filenames

Value

the function exports a RData object (*SaTAnn_plots_RData) containing data to produce all plots, and produces different QC plots in .pdf format. The plots created are as follows:

`ORFs_found`: Number of ORF categories detected per gene biotype.
`ORFs_found_pct_tr`: Distribution of ORF_pct_P_sites (`ORFs_found_ORFs_pM`: Distribution of ORFs_pM (ORFs per Million, similar to TPM) for different ORF categories and gene biotypes.
`ORFs_found_len`: Distribution of ORF length for different ORF categories and gene biotypes.
`ORFs_genes`: Number of detected ORFs per gene.
`ORFs_genes_tpm`: Gene level TPM values, plotted by number of ORFs detected.
`ORFs_maxiso`: Number of genes plotted against the percentages of gene translation of their most translated ORF.
`ORFs_maxiso_tpm`: Gene level TPM values, plotted against the percentages of gene translation of their most translated ORF.
`Sel_txs_genes`: Number of genes plotted against the number of selected transcripts.
`Sel_txs_genes_tpm`: Gene level TPM values, plotted against the number of selected transcripts.
`Sel_txs_genes_pct`: Percentages of annotated trascripts per gene, plotted against the number of selected transcripts.
`Sel_txs_bins_juns`: Percentages of covered exonic bins or junctions, using all annotated transcripts, coding transcripts only, or the set of selected transcripts.
`Meta_splicing_coverage`: Aggregate signal of Ribo-seq coverage and normalized ORF coverage across different splice sites combinations, with different mixtures of translated overlapping ORFs.

Author(s)

Lorenzo Calviello, <calviello.l.bio@gmail.com>

See Also

run_SaTAnn

---

prepare_annotation_files *Prepare comprehensive sets of annotated genomic features*

Description

This function processes a gtf file and a twobit file (created using faToTwoBit from ucsc tools: http://hgdownload.soe.ucsc.edu/admin/exe/) to create a comprehensive set of genomic regions of interest in genomic and transcriptomic space (e.g. introns, UTRs, start/stop codons). In addition, by linking genome sequence and annotation, it extracts additional info, such as gene and transcript biotypes, genetic codes for different organelles, or chromosomes and transcripts lengths.

Usage

```
prepare_annotation_files(annotation_directory, twobit_file, gtf_file,
  scientific_name = "Homo.sapiens", annotation_name = "genc25",
  export_bed_tables_TxDb = T, forge_BSgenome = T, create_TxDb = T)
```

Arguments

annotation_directory	The target directory which will contain the output files
twobit_file	Full path to the genome file in twobit format
gtf_file	Full path to the annotation file in GTF format
scientific_name	A name to give to the organism studied; must be two words separated by a ".", defaults to Homo.sapiens
annotation_name	A name to give to annotation used; defaults to genc25
export_bed_tables_TxDb	Export coordinates and info about different genomic regions in the annotation_directory? It defaults to TRUE
forge_BSgenome	Forge and install a BSgenome package? It defaults to TRUE
create_TxDb	Create a TxDb object and a *Rannot object? It defaults to TRUE

Details

This function uses the makeTxDbFromGFF function to create a TxDb object and extract genomic regions and other info to a *Rannot R file; the mapToTranscripts and mapFromTranscripts functions are used to map features to genomic or transcript-level coordinates. GTF file mist contain "exon" and "CDS" lines, where each line contains "transcript_id" and "gene_id" values. Additional values such as "gene_biotype" or "gene_name" are also extracted. Regarding sequences, the twobit

file, together with input scientific and annotation names, is used to forge and install a BSgenome package using the `forgeBSgenomeDataPkg` function.

The resulting GTF_annotation object (obtained after runnning `load_annotation`) contains:

`txs`: annotated transcript boundaries.
`txs_gene`: GRangesList including transcript grouped by gene.
`seqinfo`: indicating chromosomes and chromosome lengths.
`start_stop_codons`: the set of annotated start and stop codon, with respective transcript and gene_ids. reprentative_mostcommon,reprentative_boundaries and reprentative_5len represent the most common start/stop codon, the most upstream/downstream start/stop codons and the start/stop codons residing on transcripts with the longest 5'UTRs
`cds_txs`: GRangesList including CDS grouped by transcript.
`introns_txs`: GRangesList including introns grouped by transcript.
`cds_genes`: GRangesList including CDS grouped by gene.
`exons_txs`: GRangesList including exons grouped by transcript.
`exons_bins`: the list of exonic bins with associated transcripts and genes.
`junctions`: the list of annotated splice junctions, with associated transcripts and genes.
`genes`: annotated genes coordinates.
`threeutrs`: collapsed set of 3'UTR regions, with correspinding gene_ids. This set does not overlap CDS region.
`fiveutrs`: collapsed set of 5'UTR regions, with correspinding gene_ids. This set does not overlap CDS region.
`ncIsof`: collapsed set of exonic regions of protein_coding genes, with correspinding gene_ids. This set does not overlap CDS region.
`ncRNAs`: collapsed set of exonic regions of non_coding genes, with correspinding gene_ids. This set does not overlap CDS region.
`introns`: collapsed set of intronic regions, with correspinding gene_ids. This set does not overlap exonic region.
`intergenicRegions`: set of intergenic regions, defined as regions with no annotated genes on either strand.
`trann`: DataFrame object including (when available) the mapping between gene_id, gene_name, gene_biotypes, transcript_id and transcript_biotypes.
`cds_txs_coords`: transcript-level coordinates of ORF boundaries, for each annotated coding transcript. Additional columns are the same as as for the `start_stop_codons` object.
`genetic_codes`: an object containing the list of genetic code ids used for each chromosome/organelle. see GENETIC_CODE_TABLE for more info.
`genome_package`: the name of the forged BSgenome package. Loaded with `load_annotation` function.
`stop_in_gtf`: stop codon, as defined in the annotation.

Value

a TxDb file and a *Rannot files are created in the specified `annotation_directory`. In addition, a BSgenome object is forged, installed, and linked to the *Rannot object

Author(s)

Lorenzo Calviello, <calviello.l.bio@gmail.com>

See Also

load_annotation, forgeBSgenomeDataPkg, makeTxDbFromGFF, run_SaTAnn.

---

prepare_for_SaTAnn *Prepare the "for_SaTAnn" file*

Description

Prepare the "for_SaTAnn" file

Usage

```
prepare_for_SaTAnn(annotation_file, bam_file,
  path_to_rl_cutoff_file = NA, chunk_size = 5e+06,
  path_to_P_sites_plus_bw = NA, path_to_P_sites_minus_bw = NA,
  path_to_P_sites_uniq_plus_bw = NA,
  path_to_P_sites_uniq_minus_bw = NA,
  path_to_P_sites_uniq_mm_plus_bw = NA,
  path_to_P_sites_uniq_mm_minus_bw = NA, dest_name = NA)
```

Arguments

annotation_file	Full path to the annotation file (*Rannot)
bam_file	Full path to the bam file
path_to_rl_cutoff_file	path to the rl_cutoff_file file specifying in 3 columns the read lengths, cutoffs and compartments ("nucl" for standard chromosomes)
chunk_size	the number of alignments to read at each iteration, defaults to 5000000, increase when more RAM is available
path_to_P_sites_plus_bw	path to a bigwig file containing P_sites positions on the plus strand
path_to_P_sites_minus_bw	path to a bigwig file containing P_sites positions on the minus strand
path_to_P_sites_uniq_plus_bw	(Optional) path to a bigwig file containing uniquely mapping P_sites positions on the plus strand
path_to_P_sites_uniq_minus_bw	(Optional) path to a bigwig file containing uniquely mapping P_sites positions on the minus strand

path_to_P_sites_uniq_mm_plus_bw
: (Optional) path to a bigwig file containing uniquely mapping (with mismatches) P_sites positions on the plus strand

path_to_P_sites_uniq_mm_minus_bw
: (Optional) path to a bigwig file containing uniquely mapping (with mismatches) P_sites positions on the minus strand

dest_name
: prefix to use for the output files. Defaults to same as bam_file (appends "for_SaTAnn" to its filename)

Details

This function uses a list of pre-determined read lengths, cutoffs and compartments to calculate P_sites positions.
Alternatively, bigwig files containing P_sites position for each strand can be specified. Optional bigwig files for uniquely mapping P_sites position (with and without mismatches) can be specified to obtain more statistics on the SaTAnn-identified ORFs

Author(s)

Lorenzo Calviello, <calviello.l.bio@gmail.com>

See Also

run_SaTAnn

---

run_SaTAnn *Run the SaTAnn pipeline*

Description

This wrapper function runs the entire SaTAnn pipeline

Usage

```
run_SaTAnn(for_SaTAnn_file, annotation_file, n_cores,
  prefix = for_SaTAnn_file, gene_name = NA, gene_id = NA,
  genomic_region = NA, write_temp_files = T, write_GTF_file = T,
  write_protein_fasta = T, interactive = T,
  stn.orf_find.all_starts = T, stn.orf_find.nostarts = F,
  stn.orf_find.start_sel_cutoff = NA,
  stn.orf_find.start_sel_cutoff_ave = 0.5,
  stn.orf_find.cutoff_fr_ave = 0.5, stn.orf_quant.cutoff_cums = NA,
  stn.orf_quant.cutoff_pct = 2, stn.orf_quant.cutoff_P_sites = NA)
```

Arguments

for_SaTAnn_file	REQUIRED - path to the "for_SaTAnn" file containing P_sites positions and junction reads
annotation_file	REQUIRED - path to the *Rannot R file in the annotation directory used in the prepare_annotation_files function
n_cores	REQUIRED - number of cores to use
prefix	prefix to use for the output files. Defaults to same as for_SaTAnn_file (appends to its filename)
gene_name	character vector of gene names to analyze.
gene_id	character vector of gene ids to analyze
genomic_region	GRanges object with genomic regions to analyze
write_temp_files	write temporary files. Defaults to TRUE
write_GTF_file	write a GTF files with the ORF coordinates. Defaults to TRUE
write_protein_fasta	write a protein fasta file. Defaults to TRUE
interactive	should put R object in global environment? Defaults to TRUE
stn.orf_find.all_starts	orf_find.all_starts parameter for the SaTAnn function
stn.orf_find.nostarts	orf_find.nostarts parameter for the SaTAnn function
stn.orf_find.start_sel_cutoff	orf_find.start_sel_cutoff parameter for the SaTAnn function
stn.orf_find.start_sel_cutoff_ave	orf_find.start_sel_cutoff_ave parameter for the SaTAnn functio
stn.orf_find.cutoff_fr_ave	orf_find.cutoff_fr_ave parameter for the SaTAnn function
stn.orf_quant.cutoff_cums	orf_quant.cutoff_cums parameter for the SaTAnn function
stn.orf_quant.cutoff_pct	orf_quant.cutoff_pct parameter for the SaTAnn function
stn.orf_quant.cutoff_P_sites	orf_quant.cutoff_P_sites parameter for the SaTAnn function

Details

A set of transcripts, together with genome sequence and Ribo-signal are analyzed to extract translated ORFs

Value

A set of output files containing transcript coordinates, exonic coordinates and annotation for each ORF, including optional GTF and protein fasta files.

The description for each list object is as follows:

`tmp_SaTAnn_results`: (Optional) RData object file containing the entire set of results for each genomic region.
`final_SaTAnn_results`: RData object file containing the final SaTAnn results, see `SaTAnn`.
`Protein_sequences.fasta`: (Optional) Fasta file containing the set of translated proteins .
`Detected_ORFs.gtf`: GTF file containing coordinates of the detected ORFs.

In addition, new columns are added in the ORFs_tx file:

`ORFs_pM`: number of P_sites for each ORF, divided by ORF length and summing up to a million (akin to TPM).

Author(s)

Lorenzo Calviello, <calviello.l.bio@gmail.com>

See Also

`prepare_annotation_files`, `load_annotation`, `SaTAnn`

`SaTAnn` *Detection, quantification and annotation of translated ORFs in a genomic region*

Description

This function detects, quantifies and annotates actively translated ORF in a genomic region

Usage

```
SaTAnn(region, for_SaTAnn, genetic_code_region, orf_find.all_starts = T,
  orf_find.nostarts = F, orf_find.start_sel_cutoff = NA,
  orf_find.start_sel_cutoff_ave = 0.5, orf_find.cutoff_fr_ave = 0.5,
  orf_quant.cutoff_cums = NA, orf_quant.cutoff_pct = 2,
  orf_quant.cutoff_P_sites = NA)
```

Arguments

`region`	GRanges object with genomic coordinates of the genomic region analyzed
`for_SaTAnn`	"for_SaTAnn" Robject containing P_sites positions and junction reads

genetic_code_region
: GENETIC_CODE table to use

orf_find.all_starts
: get_all_starts parameter for the detect_translated_orfs function

orf_find.nostarts
: Stop_Stop parameter for the detect_translated_orfs function

orf_find.start_sel_cutoff
: cutoff parameter for the detect_translated_orfs function

orf_find.start_sel_cutoff_ave
: cutoff_ave parameter for the detect_translated_orfs function

orf_find.cutoff_fr_ave
: cutoff parameter for the detect_translated_orfs function

orf_quant.cutoff_cums
: cutoff_cums parameter for the select_quantify_ORFs function

orf_quant.cutoff_pct
: cutoff_pct parameter for the select_quantify_ORFs function

orf_quant.cutoff_P_sites
: cutoff_P_sites parameter for the select_quantify_ORFs function

Details

A set of transcripts, together with genome sequence and Ribo-signal are analyzed to extract translated ORFs

Value

A list containing transcript coordinates, exonic coordinates and annotation for each ORF.

The description for each list object is as follows:

ORFs_tx: transcript coordinates of the detected ORFs.
ORFs_gen: genomic (exon) coordinates of the detected ORFs.
ORFs_feat: list of ORF features together with mapping reads and uniqueness.
ORFs_txs_feats: list of transcript features present in the genomic region, together with mapping reads and uniqueness.
ORFs_spl_feat_longest: splicing annotation for each ORF exon, with respect to the longest annotated coding transcript for each gene.
ORFs_spl_feat_maxORF: splicing annotation for each ORF exon, with respect to the most translated ORF in each gene.
selected_txs: character vector containing the transcript ids of the selected transcripts.
ORFs_readthroughs: (Beta) transcript coordinates of the detected ORFs readthroughs.

Author(s)

Lorenzo Calviello, <calviello.l.bio@gmail.com>

See Also

select_txs, detect_translated_orfs, select_quantify_ORFs, annotate_ORFs, detect_readthrough

---

`select_quantify_ORFs` *Select and quantify ORF translation*

Description

This function selects a subset of detected ORFs and quantifies their translation

Usage

```
select_quantify_ORFs(results_ORFs, P_sites, P_sites_uniq,
  cutoff_cums = NA, cutoff_pct = 2, cutoff_P_sites = NA,
  optimiz = FALSE, scaling = TRUE)
```

Arguments

`results_ORFs`	Full list of detected ORFs, from `detect_translated_ORFs`
`P_sites`	GRanges object with P_sites positions
`P_sites_uniq`	GRanges object with uniquely mapping P_sites positions
`cutoff_cums`	cutoff to select ORFs until <x> percentage of total gene translation. Defaults to 99
`cutoff_pct`	minimum percentage of total gene translation for an ORF to be selected. Defaults to 1
`cutoff_P_sites`	minimum number of P_sites assigned to the ORF to be selected. Defaults to 10
`optimiz`	(Beta) should numerical optimization (minimizing distance between observed coverage and expected coverage) be used to quantify ORF translation? Defaults to FALSE
`scaling`	Additional scaling value taking into account total signal on the detected ORFs to adjust quantification estimates (recommended). Defaults to TRUE

Details

ORFs are first selected using the same method as in the `select_txs` function, but using ORF features (ORF structures are treated as transcript structures).
Ribo-seq coverage (reads/length) on bins and junctions (set to a length of 60) is used to derive a scaling factor (0-1) for each ORF, which indicates how much of the ORF coverage can be assigned to such ORF (1 when no other ORF is present). When no unique features are present on an ORF, an adjusted scaling value is calculated subtracting coverage expected from a ORF with a unique feature. When no unique features are present on any ORF, scaling values are calculated assuming uniform coverage on each ORF.
ORFs are then further filtered to exclude lowly translated ORFs and quantification/selection is re-iterated until no ORF is further filtered out. Percentage of total gene translation and length-adjusted quantification estimates are produced. More details about the quantificatin procedure can be found

in the SaTAnn manuscript.

Additional columns are added to the ORFs_tx object:
`P_sites`: P_sites_raw value from `detect_translated_ORFs` divided by the ORF scaling value.
`ORF_pct_P_sites`: Percentage of gene translation output for the ORF, derived using P_sites values.
`ORF_pct_P_sites_pN`: Percentage of gene translation ouptut (adjusted by length) for the ORF, derived using P_sites values.
`unique_features_reads`: initial number of reads on each unique ORF feature. `NA` when no unique feature is present.
`adj_unique_features_reads`: final number of reads on each unique ORF feature after the ORF filtering/quantification procedure. `NA` when no unique feature is present.
`scaling_factors`: Set of 3 scaling factors assigned to the ORF using intial unique ORF features, after adjusting for the presence of ORFs with no unique features, and final scaling factor after correcting for total Ribo-seq coverage on the gene.

Value

modified `results_ORFs` object with the selected ORFs including quantification estimates.

Author(s)

Lorenzo Calviello, <calviello.l.bio@gmail.com>

See Also

`detect_translated_orfs`, `select_txs`

`select_start` *Select start codon*

Description

This function selects the start codon for ORFs in the same transcript

Usage

```
select_start(ORFs, P_sites_rle, cutoff = NA, cutoff_ave = 0.5)
```

Arguments

`ORFs`	Set of detected ORFs
`P_sites_rle`	Rle signal of P_sites along the transcript
`cutoff`	cutoff of total in-frame signal between start codons (sensitive to outliers). Defaults to NA
`cutoff_ave`	cutoff for frequency of in-frame codons between two start codons (less sensitive to outliers). Defaults to .5

Details

ORFs are divided based on stop codon and Ribo-seq signal between start codons is used to select one.
When more than `cutoff_ave` fraction of codons is in-frame between two candidate start codons, the most upstream is selected.

Value

Set of detected ORFs, including info about the possible longest ORF for that frame.

Author(s)

Lorenzo Calviello, <calviello.l.bio@gmail.com>

See Also

`detect_translated_orfs`, `get_orfs`

---

select_txs *Select a subset of transcripts with Ribo-seq data*

Description

This function flattens all annotated transcript structures and uses Ribo-seq to select a subset of transcripts.

Usage

```
select_txs(region, annotation, P_sites, P_sites_uniq, junction_counts)
```

Arguments

`region`	genomic region being analyzed
`annotation`	Rannot object containing annotation of CDS and transcript structures (see `prepare_annotation_files`)
`P_sites`	GRanges object with P_sites positions
`P_sites_uniq`	GRanges object with uniquely mapping P_sites positions
`junction_counts`	GRanges object containing Ribo-seq counts on the set of annotated junctions

Details

Features (bins and junctions) are divided into shared and unique features, and into with support and without support (with or without reads mapping). A set of logical rules filters out transcripts with internal features with no support and no unique features with reads. More specific details can be found in the SaTAnn manuscript.

Value

GRanges object with the set of counts on each exonic bin and junctions, together with the list of selected transcripts

Author(s)

Lorenzo Calviello, <calviello.l.bio@gmail.com>

See Also

prepare_annotation_files

`take_Fvals_spect` *Extract output from multitaper analysis of a signal*

Description

This function uses the multitaper tool to extract F-values and multitaper spectral coefficients

Usage

```
take_Fvals_spect(x, n_tapers, time_bw, slepians_values)
```

Arguments

`x`	numeric signal to analyze
`n_tapers`	n of tapers to use
`time_bw`	time_bw parameter
`slepians_values`	set of calculated slepian functions to use in the multitaper analysis

Details

Values reported correspond to the closest frequency to 1/3 (same parameters as in RiboTaper). Padding to a minimum length of 1024 is performed to increase spectral resolution.

Value

two numeric values representing the F-value for the multitaper test and its corresponding spectral coefficient at the closest frequency to 1/3

Author(s)

Lorenzo Calviello, <calviello.l.bio@gmail.com>

See Also

detect_translated_orfs, spec.mtm, dpss

Index