Package: gsAnalysis (via r-universe)

February 13, 2025

Title Miscellaneous tools for genomic sequence analysis

Version 0.0.2

Description A miscellaneous toolbox for various genomic sequence analysis tasks. Refer to package vignettes for different topics covered in this package. Part of the y3628 analysis suite.

License GPL (>= 3)

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

biocViews CellBiology, Genetics

Imports dplyr, readr, tibble, stringr, BSgenome, Biostrings, GenomicRanges, rtracklayer, universalmotif, GenomicFeatures, SummarizedExperiment, y3628

Depends R (>= 4.1)

Suggests BSgenome.Dmelanogaster.UCSC.dm6, knitr, rmarkdown, TxDb.Dmelanogaster.UCSC.dm6.ensGene

VignetteBuilder knitr

Config/pak/sysreqs make libicu-dev libpng-dev libxml2-dev libssl-dev libx11-dev

Repository https://yeyuan98.r-universe.dev

RemoteUrl https://github.com/yeyuan98/gsAnalysis

RemoteRef HEAD

RemoteSha 1e4dc916612623309cf44d23f24d04e51e3e74d9

Contents

.MaxEntScanRun 2
.overlapWidths 2
.strandedShift 3
bam_summary_cigar 4

BranchPointScan 4
IRFinderS_read 5
IRFinderS_readSamples 6
MaxEntScan 7
phastCons 8
RIME 9
rmats_filter 9
rmats_read 10
rmats_toGRange 11
writeXStringSetNamed 12

Index **13**

`.MaxEntScanRun` *System cell of MaxEntScan perl script*

Description

System cell of MaxEntScan perl script

Usage

```
.MaxEntScanRun(ps.MES, sequences, ME.dir)
```

Arguments

`ps.MES`	What MaxEntScan script to run.
`sequences`	character vector of sequences.
`ME.dir`	Path to the unzipped MaxEntScan directory.

Value

numeric vector of the MaxEntScan return values.

`.overlapWidths` *Count number of overlapping bases*

Description

Count number of overlapping bases

Usage

```
.overlapWidths(query, subject)
```

Arguments

query	GenomicRanges::Ganges of query.
subject	GenomicRanges::Ganges of subject.

Details

'overlapping bases' is counted for each query against ALL subject ranges. If a query overlaps with two subject ranges with 5 and 4 bases, the number reported will be 5+4=9. Implementation is:

1. `GenomicRanges::subtract(query, subject)`
2. Count up the width of the resulting GRangesList
3. (original width of query) - (width of subtracted query)

Value

Integer vector of numbers of overlapping bases. Guaranteed to be the same order as the query ranges.

.strandedShift *Stranded version of GenomicRanges::shift*

Description

Stranded version of GenomicRanges::shift

Usage

```
.strandedShift(GRanges.input, shift)
```

Arguments

GRanges.input	GenomicRanges::GRanges
shift	how many bp to shift each range. For '+' stranded range, a positive shift will be towards 3' of the genomic coordinate. For '-', a positive shift will be towards 5' of the genome.

Value

Shifted GenomicRanges::GRanges.

bam_summary_cigar *Summarize CIGAR string by addition*

Description

Summarize CIGAR string by addition

Usage

```
bam_summary_cigar(cigar, which, FUNC = sum)
```

Arguments

cigar	CIGAR strings as character vector
which	CIGAR operators to extract
FUNC	Summarize function. Accept numeric vector and output length = 1.

Value

Numeric vector of sums of specified CIGAR operators.

Examples

```
cigar.test <- c(
  "148H47M1D113M1D34M1D39M460H",
  "50M"
)
bam_summary_cigar(
  # All Ops that consume reference
  # Therefore sum = reference lengths
  cigar.test, which = c("M", "D", "N", "=", "X")
)
```

BranchPointScan *Distance from branchpoint to 3' splicing site*

Description

Distance from branchpoint to 3' splicing site

Usage

```
BranchPointScan(
  GRanges.intron,
  branchpoint.motif,
  BSgenome,
  logodds.threshold = 0.5
)
```

Arguments

`GRanges.intron`	GenomicRanges::GRanges of the introns.
`branchpoint.motif`	An universalmotif::universalmotif-class object representing the branch point. Can be loaded using universalmotif::read_* methods. For example, `universalmotif::read_meme()`.
`BSgenome`	BSgenome object from Bioconductor.
`logodds.threshold`	logodds threshold used by `universalmotif::scan_sequences()`.

Value

A numeric vector in order of ranges of GRanges.intron. Ranges that do not have identified branchpoint will take NA values.

Examples

```
vignette("intron-properties")
```

IRFinderS_read *Read IRFinder-S output of a single sample*

Description

Read IRFinder-S output of a single sample

Usage

```
IRFinderS_read(result.dir, type.samples = "validated")
```

Arguments

`result.dir`	IRFinder-S output directory
`type.samples`	Which summary type to read in, see details

Details

Two sample types are output by IRFinder-S: `validated` and `full`.

- `validated`: read the "IRFinder-IR-nondir-val.txt"
- `full`: read the "IRFinder-IR-nondir.txt"

Value

tibble of IRFinder-S results

Examples

```
#TODO
```

`IRFinderS_readSamples` *Read IRFinder-S output of multiple samples*

Description

Report a "merged" SummarizedExperiment for differential analysis. Refer to details section.

Usage

```
IRFinderS_readSamples(
  named.result.dirs,
  score.column = "IR.ratio",
  join.column = c("chr", "start", "end", "strand", "symbol"),
  type.samples = "validated",
  min.samples = 3,
  wl = 0
)
```

Arguments

`named.result.dirs`	Named paths to result directories, see description.
`score.column`	Which one column to read in as score.
`join.column`	Which columns define an intron.
`type.samples`	Forwarded to `IRFinderS_read()`.
`min.samples`	How to filter retained introns, see description.
`wl`	How to filter retained introns, see description.

Details

The routine is divided into the following steps:

First, read in retained introns from each sample by `IRFinderS_read`.

Introns that have warnings defined by `wl` is removed from each sample. `wl` definition follows that of the `IRFinder Diff` routine.

Unique introns are defined by columns specified in `join.column`.

Next, consensus introns are determined. If an intron is found in at least `min.samples` of samples it is considered as a consensus intron.

Finally, intron scores from the 'full' data table are extracted, which will be put as the "scores" assay in the output SE object.

`named.result.dirs` must be a named character vector whose:

- names = group name
- values = full path to the sample result directories

Value

SummarizedExperiment object.

Examples

```
#TODO
```

MaxEntScan *Splicing site strength scoring by MaxEntScan*

Description

Splicing site strength scoring by MaxEntScan

Usage

```
MaxEntScan(path.zip.MES = "burgelab.maxent.zip", BSgenome, GRange.intron)
```

Arguments

path.zip.MES	Path to the MaxEntScan perl program (zipped). While not provided by the package, you may obtain a copy from the original MaxEntScan author. Alternatively, you may get an archived copy from Github.
BSgenome	BSgenome object from Bioconductor.
GRange.intron	GenomicRanges::GRanges of the introns. Genome must match that of the BSgenome object. This intron ranges must be stranded (i.e., only contains '+' and '-' strand values.)

Value

data.frame with the following columns. Rows are guaranteed to match order of the input introns.

- MaxEnt.5ss, score for the 5' splicing site
- MaxEnt.3ss, score for the 3' splicing site

Examples

```
vignette("intron-properties")
```

phastCons *Conservation scoring by phastCons*

Description

Conservation scoring by phastCons

Usage

```
phastCons(GRange.intron, bw.phastCons.path, bed.phastCons.path)
```

Arguments

GRange.intron	GenomicRanges::GRanges of the introns.
bw.phastCons.path	Path to a bigwig file of phastCons scores. This may be retrieved from the UCSC (e.g., dm6). Look for the dm6.phastCons124way.bw file.
bed.phastCons.path	Path to a bed file of conserved regions annotated by phastCons. This may be retrieved from the UCSC (e.g., dm6). Look for the phastConsElements124way.txt.gz file.

Details

This function can be used for compute scores for any GRanges of interest.

Value

data.frame. Number of rows is the same as number of ranges of GRange.intron. Results have the following columns

mean Mean phastCons values over each range

perc.in.element Percentage of bases in conserved phastCons elements for each range

Examples

```
vignette("intron-properties")
```

RIME *Intron length ratio to mean neighboring exons*

Description

Intron length ratio to mean neighboring exons

Usage

```
RIME(
  txdb = TxDb.Dmelanogaster.UCSC.dm6.ensGene::TxDb.Dmelanogaster.UCSC.dm6.ensGene,
   GRange.intron
)
```

Arguments

txdb	Bioconductor TxDb. Must match the intron GRanges.
GRange.intron	GenomicRanges::GRanges of the introns.

Value

Numeric vector of the ratios.

Examples

```
vignette("intron-properties")
```

rmats_filter *Filtering rMATS output of different AS patterns*

Description

Filtering rMATS output of different AS patterns

Usage

```
rmats_filter(df, supp_reads = c(5, 2), incLvl_limits = c(0.05, 0.95))
```

Arguments

df	Data frame of parsed rMATS output. Use rmats_read().
supp_reads	Supporting read filter. See details.
incLvl_limits	Inclusion level filter. See details.

Details

Typically, it is desirable to filter the rMATS output to:

- Reject detected AS events that have too few supporting reads.
- Remove AS events whose inclusion levels are extreme.

This function applies the following filters (default parameters assumed):

1. Supporting read count (both EJC and IJC) must be >=5 in >=2 samples.
2. Inclusion level must be in the range of 0.05-0.95.

Value

Filtered rMATS output. See details.

Examples

```
#TODO
```

`rmats_read` *Read rMATS output of different AS patterns*

Description

Read rMATS output of different AS patterns

Usage

```
rmats_read(outputs.dir, method)
```

Arguments

`outputs.dir`	Output directory of rMATS.
`method`	Either "JC" or "JCEC", see details.

Details

This is a convenience function for reading rMATS output, merging different splicing patterns into a single data frame for easier further analysis. Refer to rMATS Github for details on results generated by rMATS: https://github.com/Xinglab/rmats-turbo/blob/v4.3.0/README.md#output

rMATS normalizes lengths of individual splicing variants. There are two methods used for this normalization: JC and JCEC. Refer to the rMATS paper for more details: https://doi.org/10.1073/pnas.1419161111

rMATS coordinated are 0-based; the exact meaning of start and end varies by splicing pattern type:

- A3SS/A5SS: start/end = start/end of the long exon (inclusion form).
- SE: start/end = start/end of the skipped exon (inclusion form).
- MXE: start/end = start of the first exon / end of the second exon.
- RI: start/end = start/end of the retained intron (inclusion form).

Value

Data frame of rMATS output

Examples

```
# TODO
```

`rmats_toGRange` *Converts rMATS data frame to GenomicRange*

Description

Convenience function for converting rMATS data to GRanges.

Usage

```
rmats_toGRange(df, ...)
```

Arguments

`df`	rMATS data read by `rmats_read()`.
`...`	<tidy-select> See details. Three columns are used to fill in GRanges information: `chr`, `start_0base`, `end`. Two columns are by default added to the metadata: `geneSymbol`, `Type`. Extra metadata columns are specified by the tidyselect ellipsis.

Value

GRanges object

Examples

```
#TODO
```

writeXStringSetNamed *writeXStringSet with identifiers*

Description

Write a XStringSet to file (FASTA/FASTQ) with names as identifiers.

Usage

```
writeXStringSetNamed(x, filepath, append = FALSE, compress = FALSE, ...)
```

Arguments

x	Object XString to write to file
filepath	File to write to
append	Must be False (does not support append)
compress	Must be False (does not support compression)
...	Forwarded to Biostrings::writeXStringSet()

Details

Somehow the original Biostrings::writeXStringSet() does not write identifiers. This function uses names provided by name() and write ID.

Examples

```
#TODO
```

Index

.MaxEntScanRun, 2
.overlapWidths, 2
.strandedShift, 3

bam_summary_cigar, 4
BranchPointScan, 4

IRFinderS_read, 5
IRFinderS_readSamples, 6

MaxEntScan, 7

phastCons, 8

RIME, 9
rmats_filter, 9
rmats_read, 10
rmats_toGRange, 11

writeXStringSetNamed, 12