

Package: gsAnalysis (via r-universe)

September 23, 2026

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, glmmTMB, emmeans, graphics, parallel

Depends R (>= 4.1)

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

VignetteBuilder knitr

Config/pak/sysreqs cmake make libbz2-dev libicu-dev liblzma-dev libpng-dev libxml2-dev libssl-dev libx11-dev xz-utils zlib1g-dev

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

Date/Publication 2025-06-26 20:25:43 UTC

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

RemoteRef HEAD

RemoteSha 0d98b644cb29c88be7b325b1ce12e70343f53557

Contents

.MaxEntScanRun	2
.overlapWidths	3

.strandedShift	3
bam_summary_cigar	4
BranchPointScan	5
diff_breg	5
diff_breg_single	6
IRFinderS_read	7
IRFinderS_readSamples	7
MaxEntScan	9
phastCons	10
readStarGeneCounts	11
RIME	11
rmats_filter	12
rmats_read	13
rmats_toGRange	14
se_subset	14
writeXStringSetNamed	15
Index	16

.MaxEntScanRun	<i>System cell of MaxEntScan perl script</i>
----------------	--

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.

<code>.overlapWidths</code>	<i>Count number of overlapping bases</i>
-----------------------------	--

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.

<code>.strandedShift</code>	<i>Stranded version of GenomicRanges::shift</i>
-----------------------------	---

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	<i>Summarize CIGAR string by addition</i>
-------------------	---

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	<i>Distance from branchpoint to 3' splicing site</i>
-----------------	--

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 branch-point will take NA values.

Examples

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

diff_breg	<i>Differential analysis with beta regression</i>
-----------	---

Description

Differential analysis with beta regression

Usage

```
diff_breg(se, design, target, n.cores = "auto")
```

Arguments

se	SummarizedExperiment containing data to test.
design	Design formula.
target	Which target to perform test.
n.cores	Multicore by parallel. Either "auto", integer, or NA. NA means no parallel processing.

Value

data.frame of test results with FDR.

Examples

```
# TODO
```

diff_breg_single	<i>Differential analysis with beta regression</i>
------------------	---

Description

This takes data.frame input.

Usage

```
diff_breg_single(data, design, target, check = TRUE)
```

Arguments

data	Data to use.
design	Design formula of the problem.
target	Which target to test.
check	Whether to perform sanity checks.

Value

data.frame of test results.

Examples

```
# TODO.
```

IRFinderS_read	<i>Read IRFinder-S output of a single sample</i>
----------------	--

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"

For column names please refer to the function body

Value

tibble of IRFinder-S results

Examples

```
#TODO
```

IRFinderS_readSamples	<i>Read IRFinder-S output of multiple samples</i>
-----------------------	---

Description

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

Usage

```
IRFinderS_readSamples(
  named.result.dirs,
  assay.columns = c("IR.ratio", "depth", "n.reads.spliceExact"),
  join.columns = c("chr", "start", "end", "strand", "symbol"),
  type.samples = "validated",
  min.samples = 3,
  wl = 0
)
```

Arguments

<code>named.result.dirs</code>	Named paths to result directories, see description.
<code>assay.columns</code>	Which column(s) to read in as assays of the output.
<code>join.columns</code>	Which columns define an intron.
<code>type.samples</code>	Forwarded to <code>IRFinderS_read()</code> .
<code>min.samples</code>	How to filter retained introns, see description.
<code>wl</code>	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`.

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

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

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.columns`.

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 data from the 'full' data table are extracted, which will be put as assays in the output SE object. Which data columns are included is defined by `assay.columns`.

wl filtering:

This filtering determines whether an intron is included in a sample. It looks at the warning flags by `IRFinder` and possible values are:

0 -> no filter; 1 -> LowCover introns removed; 2 -> Lowcover & LowSplicing; 3 -> LowCover & LowSplicing & MinorIsoform; 4 -> LowCover & LowSplicing & MinorIsoform & NonUniformIntronCover

Value

SummarizedExperiment object of `IRFinder` results.

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

<code>path.zip.MES</code>	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 .
<code>BSgenome</code>	BSgenome object from Bioconductor.
<code>GRange.intron</code>	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")
```

readStarGeneCounts	<i>Read STAR Aligner Gene Counts</i>
--------------------	--------------------------------------

Description

Read STAR Aligner Gene Counts

Usage

```
readStarGeneCounts(named_paths, clean = FALSE)
```

Arguments

named_paths	Named list of count table paths.
clean	Boolean, whether to remove entries starting with "N_".

Value

SummarizedExperiment of counts. Metadata sample will be names.

Examples

```
# TODO
```

RIME	<i>Intron length ratio to mean neighboring exons</i>
------	--

Description

Intron length ratio to mean neighboring exons

Usage

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

Arguments

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

Value

Numeric vector of the ratios.

Examples

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

rmats_filter	Assign filter labels of rMATS output
--------------	--------------------------------------

Description

Assign filter labels of rMATS output

Usage

```
rmats_filter(
  df,
  readCov = 10,
  maxPSI = 0.95,
  minPSI = 0.05,
  sigFDR = 0.1,
  sigDeltaPSI = 0.05,
  res_column = "filter"
)
```

Arguments

df	rMATS data read by rmats_read().
readCov	Minimum read coverage.
maxPSI	Maximum percent spliced-in PSI.
minPSI	Minimum PSI.
sigFDR	FDR threshold for significance.
sigDeltaPSI	PSI threshold for significance.
res_column	Which column to store filter results

Details

This function reproduces the data filter demonstrated in Xing Lab 2024 Nature Protocol tutorial. See Github repository [Xinglab/rmats-turbo-tutorial](#) and refer to `rmats_filtering.py`.

AS events that does not satisfy any of the following criteria will be given the *remove* label:

1. Average read counts (IJC+SJC) in both Sample 1 and Sample 2 conditions are at least readCov.
2. Average PSI in both Sample 1 and Sample 2 conditions are between minPSI and maxPSI.

If all criteria above are satisfied, events will be labeled by statistical significance:

1. $FDR < sigFDR$ and $IncLevelDifference > sigDeltaPSI$: *up*.
2. $FDR < sigFDR$ and $IncLevelDifference < -sigDeltaPSI$: *down*.
3. Otherwise: *ns*.

Value

rMATS data table with filter label column filter.

Examples

```
# TODO
```

rmats_read	<i>Read rMATS output of different AS patterns</i>
------------	---

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 coordinates 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
```

rmts_toGRange	<i>Converts rMATS data frame to GenomicRange</i>
---------------	--

Description

Convenience function for converting rMATS data to GRanges.

Usage

```
rmts_toGRange(df, ...)
```

Arguments

df	rMATS data read by <code>rmts_read()</code> .
...	<code><tidy-select></code> 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
```

se_subset	<i>Subset SummarizedExperiment</i>
-----------	------------------------------------

Description

1. Subset columns using `expr_colData`.
2. Evaluate `expr_assay` on `assays()` of the subset.
3. Apply function `fn_rowwise` to get subset vector.
4. Use the subset vector to subset rows of the input `se`.

Usage

```
se_subset(se, expr_assay, expr_colData, fn_rowwise)
```

Arguments

<code>se</code>	SummarizedExperiment object.
<code>expr_assay</code>	Expression to evaluate on assays
<code>expr_colData</code>	Expression to evaluate on colData
<code>fn_rowwise</code>	Predicate to define rowwise filter

Value

Subset SummarizedExperiment 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

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

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, [3](#)
.strandedShift, [3](#)

bam_summary_cigar, [4](#)
BranchPointScan, [5](#)

diff_breg, [5](#)
diff_breg_single, [6](#)

IRFinderS_read, [7](#)
IRFinderS_readSamples, [7](#)

MaxEntScan, [9](#)

phastCons, [10](#)

readStarGeneCounts, [11](#)
RIME, [11](#)
rmats_filter, [12](#)
rmats_read, [13](#)
rmats_toGRange, [14](#)

se_subset, [14](#)

writeXStringSetNamed, [15](#)