Package: gDNAx (via r-universe)

November 1, 2026

Type Package

Title Diagnostics for assessing genomic DNA contamination in RNA-seq data

Version 1.11.0

Description Provides diagnostics for assessing genomic DNA contamination in RNA-seq data, as well as plots representing these diagnostics. Moreover, the package can be used to get an insight into the strand library protocol used and, in case of strand-specific libraries, the strandedness of the data. Furthermore, it provides functionality to filter out reads of potential gDNA origin.

License Artistic-2.0

Encoding UTF-8

Depends R (>= 4.3)

Imports methods, BiocGenerics, BiocParallel, matrixStats, Biostrings, S4Vectors, IRanges, Seqinfo, GenomeInfoDb, GenomicRanges, GenomicFiles, GenomicAlignments, GenomicFeatures, Rsamtools, AnnotationHub, RColorBrewer, AnnotationDbi, bitops, plotrix, SummarizedExperiment, grDevices, graphics, stats, utils, cli

Suggests BiocStyle, knitr, rmarkdown, RUnit, TxDb.Hsapiens.UCSC.hg38.knownGene, gDNAinRNAseqData

biocViews Transcription, Transcriptomics, RNASeq, Sequencing, Preprocessing, Software, GeneExpression, Coverage, DifferentialExpression, FunctionalGenomics, SplicedAlignment, Alignment

VignetteBuilder knitr

Roxygen list(markdown=TRUE)

RoxygenNote 7.3.3

URL https://github.com/functionalgenomics/gDNAx

BugReports https://github.com/functionalgenomics/gDNAx/issues

Collate 'AllGenerics.R' 'AllClasses.R' 'utils.R' 'dx.R'
'filterBAMtx.R' 'tx.R' 'strandedness.R' 'gDNAx.R'
'gDNAx-pkg-deprecated.R'

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

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

Date/Publication 2026-04-28 13:01:25 UTC

RemoteUrl https://github.com/bioc/gDNAx

RemoteRef HEAD

RemoteSha b94edc32cd52828080032a51dbb43c56c62dc09c

Contents

gDNAx-package 2
filterBAMtx 3
gDNAdx 6
gDNAtx 8
gDNAx-class 10
identifyStrandMode 13

Index **18**

`gDNAx-package` *gDNAx: diagnostics for assessing genomic DNA contamination in RNA-seq data*

Description

The gDNAx package provides diagnostics for assessing genomic DNA contamination in RNA-seq data, as well as plots representing these diagnostics. Moreover, the package can be used to get an insight into the strand library protocol used and, in case of strand-specific libraries, the strandedness of the data. Furthermore, it provides functionality to filter out reads of potential gDNA origin.

Details

The main functions are:

- `gDNAdx()` - calculate diagnostics for assessing the presence of genomic DNA in RNA-seq data over a subset of the alignments in the input BAM files.
- `getDx()` and `plot()` - get and plot statistics on genomic DNA contamination levels, respectively.
- `strandedness()` - obtain estimates of strandedness in RNA-seq data samples based on the proportion of reads aligning to the same or opposite strand as transcripts in the annotations.
- `classifyStrandMode()` - classify the output of `strandedness()` into strand modes for each BAM file.

- filterBAMtxFlag and filterBAMtx - filter alignments in a BAM file using criteria based on a transcriptome annotation.

For detailed information on usage, see the package vignette, by typing `vignette("gDNAx")`.

All questions and bug reports should be posted to the Bioconductor Support Site:

https://support.bioconductor.org

The code of the development version of the package is available at the GitHub repository:

https://github.com/functionalgenomics/gDNAx

Author(s)

Maintainer: Robert Castelo <robert.castelo@upf.edu>

Authors:

- Beatriz Calvo-Serra <beatriz.calvo@upf.edu>

See Also

Useful links:

- https://github.com/functionalgenomics/gDNAx
- Report bugs at https://github.com/functionalgenomics/gDNAx/issues

---

`filterBAMtx` *Filter alignments in a BAM file using a transcriptome*

Description

Filter alignments in a BAM file using criteria based on a transcriptome annotation.

Use 'filterBAMtxFlag()' to set what types of alignment in a BAM file should be filtered using the function 'filterBAMtx()', among being splice-compatible with one or more exon-exon junctions, splice-compatible exonic, splice-compatible exonic in a window, intronic or intergenic.

Usage

```
filterBAMtx(
  object,
  path = ".",
  txflag = filterBAMtxFlag(),
  param = ScanBamParam(),
  yieldSize = 1e+06,
  wsize = 1000,
  wstep = 100,
  pstrness = 0.6,
  p.value = 0.05,
  p.adj.method = p.adjust.methods,
```

```
  verbose = TRUE,
  BPPARAM = SerialParam(progressbar = verbose)
)

filterBAMtxFlag(
  isSpliceCompatibleJunction = FALSE,
  isSpliceCompatibleExonic = FALSE,
  isInStrandedWindow = FALSE,
  isIntronic = FALSE,
  isIntergenic = FALSE
)

testBAMtxFlag(flag, value)
```

Arguments

object	gDNAx object obtained with the function 'gDNAdx()'.
path	Directory where to write the output BAM files.
txflag	A value from a call to the function 'filterBAMtxFlag()'.
param	A 'ScanBamParam' object.
yieldSize	(Default 1e6) Number of records in the input BAM file to yield each time the file is read. The lower the value, the smaller memory consumption, but in the case of large BAM files, values below 1e6 records may decrease the overall performance.
wsize	(Default 1000) Window size employed when the argument `txflag` includes the value `isInStrandedWindow=TRUE`.
wstep	(Default 100) Window step employed when the argument `txflag` includes the value `isInStrandedWindow=TRUE`.
pstrness	(Default 0.6) Strandedness value above which we consider a target read alignment to occur in stranded window.
p.value	(Default 0.05) Numeric value between 0 and 1 specifying the adjusted p-value cutoff under which we reject the null hypothesis that a target read alignment occurs in a window with an strandedness value below the one given in the parameter `pstrness`. This parameter is only used when the argument `txflag` includes the value `isInStrandedWindow=TRUE`.
p.adj.method	(Default "holm") Method used to adjust p-values that are compared against the cutoff value specified in the parameter `p.value`. Adjusted p-values are calculated using the base R function `p.adjust()` and this argument is directly passed to the argument `method` of that function.
verbose	(Default TRUE) Logical value indicating if progress should be reported through the execution of the code.
BPPARAM	An object of a `BiocParallelParam` subclass to configure the parallel execution of the code. By default, a `SerialParam` object is used, which does not use any parallelization, with the flag `progress=TRUE` to show progress through the calculations.

isSpliceCompatibleJunction
: (Default FALSE) Logical value indicating if spliced alignments overlapping a transcript in a "splice compatible" way should be included in the BAM file. For paired-end reads, one or both alignments must have one or more splice sites compatible with splicing. See OverlapEncodings.

isSpliceCompatibleExonic
: (Default FALSE) Logical value indicating if alignments without a splice site, but that overlap a transcript in a "splice compatible" way, should be included in the BAM file. For paired-end reads, none of the alignments must be spliced, and each pair can be in different exons (or in the same one), as long as they are "splice compatible". See OverlapEncodings.

isInStrandedWindow
: (Default FALSE) Logical value indicating whether an alignment occurs in a stranded windows. More concretely, for each alignment, strandedness will be tested with respect to the rest of the alignments occurring in overlapping windows. This filter will assign a TRUE value when those tests are passed.

isIntronic
: (Default FALSE) Logical value indicating if alignments mapping to introns should be included in the BAM file.

isIntergenic
: (Default FALSE) Logical value indicating if alignments aligned to intergenic regions should be included in the BAM file.

flag
: A value from a call to the function 'filterBAMtxFlag()'.

value
: A character vector with the name of a flag.

Value

A vector of output filename paths.

Examples

```
library(gDNAinRNAseqData)

library(TxDb.Hsapiens.UCSC.hg38.knownGene)
txdb <- TxDb.Hsapiens.UCSC.hg38.knownGene

# Getting the 'gDNAx' object
bamfiles <- LiYu22subsetBAMfiles()
bamfiles <- bamfiles[c(1,7)] # using a subset of samples
gdnax <- gDNAdx(bamfiles, txdb, singleEnd=FALSE, strandMode=NA)

# Filtering splice-compatible alignments and writing them into new BAM files
fbf <- filterBAMtxFlag(isSpliceCompatibleJunction=TRUE,
                       isSpliceCompatibleExonic=TRUE)
dir <- tempdir()
fstats <- filterBAMtx(gdnax, path=dir, txflag=fbf)
list.files(dir, pattern="*.bam$")


# Filtering splice-compatible alignments and writing them into new BAM files
fbf <- filterBAMtxFlag(isSpliceCompatibleJunction=FALSE,
```

```
                        isSpliceCompatibleExonic=FALSE,
                        isInStrandedWindow=FALSE,
                        isIntronic=FALSE,
                        isIntergenic=FALSE)

testBAMtxFlag(fbf, "isSpliceCompatibleJunction")
```

---

gDNAdx *Calculate gDNA diagnostics*

Description

Calculate diagnostics for assessing the presence of genomic DNA (gDNA) in RNA-seq data over a subset of the alignments in the input BAM files.

Plot diagnostics calculated with gDNAdx()

Using the output from gDNAdx(), plot the genomic origin of the alignments.

Plot fragments length distributions estimated with gDNAdx()

Usage

```
gDNAdx(
  bfl,
  txdb,
  singleEnd,
  strandMode,
  stdChrom = TRUE,
  yieldSize = 100000L,
  exonsBy = c("gene", "tx"),
  minnaln = 2e+05,
  useRMSK = FALSE,
  verbose = TRUE,
  BPPARAM = SerialParam(progressbar = verbose)
)

## S4 method for signature 'gDNAx,ANY'
plot(x, group = 1L, labelpoints = FALSE, ...)

plotAlnOrigins(x, group = 1L)

plotFrgLength(x)
```

Arguments

bfl	A BamFile or BamFileList object, or a character string vector of BAM filenames.

txdb — A character string of a TxDb package, or a TxDb object, with gene and transcript annotations. For accurate calculations, it is important that the version of these annotations matches the version of the annotations used to inform the alignment of spliced reads, by the short-read aligner software that generated the input BAM files.

singleEnd — (missing by default) Logical value indicating if reads are single (TRUE) or paired-end (FALSE). When this argument is missing (default), its value will be estimated automatically from the BAM files.

strandMode — (missing by default) Integer value either 0, 1, 2 or NA, indicating how the strand of a pair of read alignments should be inferred from the strand of the first and last alignments in the pair. See the GAlignmentPairs class for further details. If singleEnd = TRUE, then strandMode is ignored. For unstranded RNA-seq libraries, use NA. When this argument is missing (default), its value will be estimated automatically from the BAM files.

stdChrom — (Default TRUE) Logical value indicating whether only alignments in the 'standard chromosomes' should be used. Consult the help page of the function keepStandardChromosomes from the package GenomeInfoDb for further information.

yieldSize — (Default 1e5) Number of records to read from each input BAM file to calculate the diagnostics.

exonsBy — (Default 'gene') Character string, either gene or tx, respectively specifying whether exon annotations should be grouped by gene or by transcript, in the estimation of strandedness values. Consult the help page of the function exonsBy() from the package GenomicFeatures for further information.

minnaln — (Default 200000) Minimum number of read alignments overlapping exonic regions considered necessary for a reliable estimation of strandedness values. A warning message is given if the number of such available alignments is smaller than the one given through this parameter. This parameter only applies if no argument is given for the strandMode parameter.

useRMSK — (Default FALSE) Logical value indicating if RepeatMasker annotations should be used when building intergenic and intronic genomic ranges for gDNA estimation. If useRMSK=TRUE, then UCSC RepeatMasker annotations will be downloaded as AnnotationHub resources, and intergenic and intronic genomic ranges will exclude them.

verbose — (Default TRUE) Logical value indicating if progress should be reported through the execution of the code.

BPPARAM — An object of a BiocParallelParam subclass to configure the parallel execution of the code. By default, a SerialParam object is used, which does not use any parallelization, with the flag progress=TRUE to show progress through the calculations.

x — A 'gDNAx' object.

group — A string character vector or a factor, with as many values as BAM files analyzed in 'x', whose values define groups among those BAM files.

labelpoints — (Default FALSE) A logical indicator that labels points in those plots where each point represents a BAM file. Labels correspond to the index number of the BAM file in 'x'.

... Named arguments to be passed to plot.

Value

A gDNAx object.

Examples

```
library(gDNAinRNAseqData)

library(TxDb.Hsapiens.UCSC.hg38.knownGene)
txdb <- TxDb.Hsapiens.UCSC.hg38.knownGene

# Retrieving BAM files
bamfiles <- LiYu22subsetBAMfiles()
bamfiles <- bamfiles[c(1,4,7)] # using a subset of samples

# Getting information about the gDNA concentrations of each BAM file
pdat <- LiYu22phenoData(bamfiles)

gdnax <- gDNAdx(bamfiles, txdb, singleEnd=FALSE, strandMode=NA)
gdnax

# plot gDNA diagnostic measures
plot(gdnax, group=pdat$gDNA, pch=19)

# plot origin of alignments per sample
plotAlnOrigins(gdnax, group=pdat$gDNA)

# plot fragments length distributions
plotFrgLength(gdnax)
```

---

gDNAtx *Remove gDNA contamination from RNA-seq data*

Description

Remove gDNA contamination from RNA-seq data by filtering read alignments in BAM files that putatively have a gDNA origin. This is currently a wrapper with convenient default values for the function filterBAMtx(), please use that function if you need greater control on how to filter RNA-seq alignments.

Usage

```
gDNAtx(
  x,
  path = ".",
  sbparam = NULL,
```

```
  yieldSize = 1000000L,
  verbose = TRUE,
  BPPARAM = SerialParam(progressbar = verbose)
)
```

Arguments

x	gDNAx object obtained with the function gDNAdx().
path	Directory where to write the filtered BAM files.
sbparam	Either NULL (default) or a ScanBamParam object. The NULL value implies that internally a ScanBamParam object is built with the following flags: isUnmappedQuery=FALSE, isProperPair=!singleEnd(x), isSecondaryAlignment=FALSE, isNotPassingQualityControls=FA isDuplicate=FALSE.
yieldSize	(Default 1e6) Number of records in the input BAM file to yield each time the file is read. The lower the value, the smaller memory consumption, but in the case of large BAM files, values below 1e6 records may decrease the overall performance.
verbose	(Default TRUE) Logical value indicating if progress should be reported through the execution of the code.
BPPARAM	An object of a BiocParallelParam subclass to configure the parallel execution of the code. By default, a SerialParam object is used, which does not use any parallelization, with the flag progress=TRUE to show progress through the calculations.

Value

A data.frame object with the number of filtered read alignments tallied by their origin.

Examples

```
library(gDNAinRNAseqData)

library(TxDb.Hsapiens.UCSC.hg38.knownGene)
txdb <- TxDb.Hsapiens.UCSC.hg38.knownGene

## fetch sample BAM files
bamfiles <- LiYu22subsetBAMfiles()
bamfiles <- bamfiles[c(1,7)] # using a subset of samples

## diagnose gDNA contamination
gdnax <- gDNAdx(bamfiles, txdb, singleEnd=FALSE, strandMode=NA)

## remove gDNA contamination
dir <- tempdir()
fstats <- gDNAtx(gdnax, path=dir)
fstats
list.files(dir, pattern="*.bam$")
```

gDNAx-class *gDNAx class*

Description

This is a class for storing the results of a call to the 'gDNAdx()' function.

Usage

```
## S4 method for signature 'gDNAx'
getDx(x)

## S4 method for signature 'gDNAx'
show(object)

## S4 method for signature 'gDNAx'
getIgc(x)

## S4 method for signature 'gDNAx'
getInt(x)

## S4 method for signature 'gDNAx'
singleEnd(x)

## S4 method for signature 'gDNAx'
strandMode(x)

## S4 replacement method for signature 'gDNAx'
strandMode(x) <- value

## S4 method for signature 'gDNAx'
allStrandModes(x)

## S4 method for signature 'gDNAx'
strandedness(x, ...)
```

Arguments

x	A gDNAx object.
object	A gDNAx object.
value	Integer value either 0, 1, 2 or NA, indicating how the strand of a pair of read alignments should be inferred from the strand of the first and last alignments in the pair. See the GAlignmentPairs class for further detail.
...	Further arguments when x is not a gDNAx object; see the help page of gDNAdx().

Value

`getDx()`: A `data.frame` object with the following columns:

- `IGC`: percentage of read alignments fully contained in intergenic regions.
- `INT`: percentage of read alignments fully contained in intronic regions.
- `SCJ`: percentage of splice-compatible junction read alignments. These are alignments compatible with a transcript in the given annotation, for which the aligned read, or at least one of the two aligned reads in the case of a paired-end layout, spans one or more exon-exon junctions over two or more exons of that transcript.
- `SCE`: percentage of splice compatible exonic alignments. These are alignments compatible with a transcript in the given annotation, but which differently to SCJ alignments, do not include an exon-exon junction in the alignment.
- `JNC`: percentage of alignments that include one or more junctions, irrespective of whether those alignments are compatible with an spliced transcript in the given annotation.
- `*FLM`: estimation of the fragment length mean in the alignments of the corresponding type (`IGC`, `SCJ`, or `SCE`).
- `STRAND`: strandedness values, with `NA` in the case that the data is unstranded.

`getIgc()`: A `GRanges` object with intergenic ranges.

`getInt()`: A `GRanges` object with intron ranges.

`singleEnd()`: Logical value indicating whether the `gDNAx` object contains data from a single-end (`TRUE`) or a paired-end (`FALSE`) RNA-seq experiment.

`strandMode()`: Integer value indicating whether the `gDNAx` object contains data from an unstranded (`NA`), stranded with the first mate read indicating the real strand (`1`), or stranded with the last mate read indicating the real strand (`2`) from an RNA-seq experiment.

Vector of strand modes for each BAM file in the gDNAx object.

A `data.frame` object with strandedness values for each BAM file in the gDNAx object.

Slots

`bfl` A `BamFileList` object.

`txdbpkg` A `TxDb` object.

`singleEnd` Logical value indicating if reads are single (`TRUE`) or paired-end (`FALSE`).

`strandMode` Integer value either 0, 1, 2 or `NA`, indicating how the strand of a pair of read alignments should be inferred from the strand of the first and last alignments in the pair. See the `GAlignmentPairs` class for further detail.

`allStrandModes` Vector of integer values each of them corresponding to a `strandMode` value estimated from a BAM file.

`suppSpeciesInAnnot` Logical value indicating whether the species metdata in the input annotations is supported by the information available at the Bioconductor GenomeInfoDb package.

`stdChrom` Logical value indicating whether only alignments in the 'standard chromosomes' should be used. Consult the help page of the function `keepStandardChromosomes` from the package `GenomeInfoDb` for further information.

`readLength` Integer value storing the read length.

`yieldSize` Integer value storing the number of alignments employed by the function `gDNAdx()`.

`diagnostics` A 'data.frame' object storing the diagnostics calculated by the function 'gDNAdx()'.

`strandedness` A 'data.frame' object storing the estimated values of strandedness, calculated when the argument `strandMode` is missing in the call to the function 'gDNAdx()'.

`igcfrglen` A 'list' object storing the fragment lengths derived from alignments in intergenic regions.

`intfrglen` A 'list' object storing the fragment lengths derived from alignments in intronic regions.

`scjfrglen` A 'list' object storing the fragment lengths derived from spliced-compatible junction alignments in transcripts.

`scefrglen` A 'list' object storing the fragment lengths derived from spliced-compatible exonic alignments in transcripts.

`sicfrglen` A 'list' object storing the fragment lengths derived from splice-incompatible alignments in transcripts.

`intergenic` A 'GRanges' object storing the intergenic feature annotations.

`intronic` A 'GRanges' object storing the intronic feature annotations.

`transcripts` A 'GRangesList' object storing the transcript annotations.

`tx2gene` A string character vector storing the correspondence between transcripts and genes according to an 'TxDb' object.

Examples

```
library(gDNAinRNAseqData)
library(TxDb.Hsapiens.UCSC.hg38.knownGene)
txdb <- TxDb.Hsapiens.UCSC.hg38.knownGene
bamfiles <- LiYu22subsetBAMfiles() # Retrieving BAM files
## one could simply call 'gDNAx(bamfiles, txdb)' but we give the arguments
## below to reduce time and verbosity when running this example
gdnax <- gDNAdx(bamfiles, txdb, singleEnd=FALSE, strandMode=NA,
                useRMSK=FALSE, verbose=FALSE)
gdnax


# Getting statistics
dx <- getDx(gdnax)
head(dx)

gdnax

igc <- getIgc(gdnax)
head(igc, n=3)

int <- getInt(gdnax)
head(int, n=3)

singleEnd(gdnax)
```

```
strandMode(gdnax)

strandMode(gdnax) <- NA

allStrandModes(gdnax)

strandedness(gdnax)
```

`identifyStrandMode` *Identify strandMode*

Description

THIS FUNCTION HAS BEEN DEPRECATED, HAS BEEN REPLACED BY `strandedness()` AND WILL WILL BE DEFUNCT AT THE NEXT RELEASE.

Identify `strandMode` (strandedness) in RNA-seq data samples based on Given strandedness values calculated assuming either the same or the opposite strand of gene annotations, classify them into an strand mode according to cutoff values specified in the parameters.

Compute strandedness for each feature in RNA-seq data samples based on the proportion of reads aligning to the same strand as feature annotations in relation to the total number of reads aligning to that feature.

Usage

```
identifyStrandMode(
  bfl,
  txdb,
  singleEnd,
  stdChrom = TRUE,
  exonsBy = c("gene", "tx"),
  minnaln = 2e+05,
  verbose = TRUE,
  BPPARAM = SerialParam(progressbar = verbose)
)

## S4 method for signature 'character'
strandedness(
  x,
  txdb,
  singleEnd,
  stdChrom = TRUE,
  exonsBy = c("gene", "tx"),
  minnaln = 2e+05,
  verbose = TRUE,
  BPPARAM = SerialParam(progressbar = verbose)
)
```

```
## S4 method for signature 'BamFileList'
strandedness(
  x,
  txdb,
  singleEnd,
  stdChrom = TRUE,
  exonsBy = c("gene", "tx"),
  minnaln = 2e+05,
  verbose = TRUE,
  BPPARAM = SerialParam(progressbar = verbose)
)

classifyStrandMode(
  strnessdat,
  strcutoff = 0.9,
  weakstrcutoff = 0.6,
  warnweakstr = TRUE
)

strnessByFeature(
  bfl,
  features,
  singleEnd = TRUE,
  strandMode = 1L,
  yieldSize = 1000000L,
  ambiguous = FALSE,
  p = 0.6,
  verbose = TRUE,
  BPPARAM = SerialParam(progressbar = verbose)
)
```

Arguments

bfl	A BamFile or BamFileList object, or a character string vector of BAM filenames.
txdb	A character string of a TxDb package, or a TxDb object, with gene and transcript annotations. For accurate calculations, it is important that the version of these annotations matches the version of the annotations used to inform the alignment of spliced reads, by the short-read aligner software that generated the input BAM files.
singleEnd	(Default FALSE) Logical value indicating if reads are single (TRUE) or paired-end (FALSE).
stdChrom	(Default TRUE) Logical value indicating whether only alignments in the 'standard chromosomes' should be used. Consult the help page of the function keepStandardChromosomes() from the package GenomeInfoDb for further information.

exonsBy	(Default 'gene') Character string, either gene or tx, respectively specifying whether exon annotations should be grouped by gene or by transcript, in the estimation of strandedness values. Consult the help page of the function exonsBy() from the package GenomicFeatures for further information.
minnaln	(Default 200000) Minimum number of read alignments overlapping exonic regions considered necessary for a reliable estimation of strandedness values. A warning message is given if the number of such available alignments is smaller than the one given through this parameter.
verbose	(Default TRUE) Logical value indicating if progress should be reported through the execution of the code.
BPPARAM	An object of a BiocParallelParam subclass to configure the parallel execution of the code. By default, a SerialParam object is used, which does not use any parallelization, with the flag progress=TRUE to show progress through the calculations.
x	A BamFileList object.
strnessdat	A data.frame object obtained with the function strandedness().
strcutoff	(Default 0.9) Minimum cutoff above which a strandedness value is considered to strongly support that read alignments originate from a specific strand.
weakstrcutoff	(Default 0.6) Minimum cutoff above which a strandedness value is considered to weakly support that read alignments originate from a specific strand.
warnweakstr	(Default TRUE) Logical value indicating whether to warn the user when strandedness values only provide a weakly support for a specific strand.
features	A GRanges or GRangesList object with annotations of features (e.g. genes, transcripts, etc.).
strandMode	(Default 1L) Numeric vector which can take values 0, 1, or 2. The strand mode is a per-object switch on GAlignmentPairs objects that controls the behavior of the strand getter. See GAlignmentPairs class for further detail. If singleEnd = TRUE, then strandMode is ignored.
yieldSize	(Default 5e5) Field inherited from BamFile. The BAM is read by chunks. yieldSize represents the number of records to read for each chunk.
ambiguous	(Default FALSE) Logical value indicating if reads that overlap a region with features annotated to both strands should be included in the strandedness value computation.
p	(Default 0.6) Numeric value for the exact binomial test performed for the strandedness value of each feature, representing the hypothesized probability of success (i.e. the strandedness value expected for a non-stranded dataset).

Details

Identify strandMode (strandedness) in RNA-seq data samples based on the proportion of reads aligning to the same or opposite strand as transcripts in the annotations.

If the value in the "strandMode1" column is > 0.90, strandMode is set to 1L. If "strandMode2" column is > 0.90, strandMode is set to 2L. If "strandMode1" and "strandMode2" are comprised between 0.40 and 0.60, strandMode is set to NA If none of the three previous criteria are met,

strandMode is set to "ambiguous". This criteria can be conservative in some cases (e.g. when there is genomic DNA contamination), for this reason we recommend to check the data.frame with strandedness values.

In case of single-end data, the same criteria are used, but the interpretation of strandMode = 1L and strandMode = 2L changes: when strandMode = 1L the strand of the read is concordant with the reference annotations, when strandMode = 2L the correct read strand is the opposite to the one of the read.

A subset of 200,000 alignments overlapping gene annotations are used to compute strandedness.

Strandedness is computed for each feature and BAM file according to the strandMode specified in case of paired-end data. For single-end, the original strand of reads is used. All alignments from the BAM file(s) are considered to compute the strandedness.

The p value should be close to 0.5, representing the strandedness expected for a non-stranded RNA-seq library.

Value

A list object with two elements:

- "strandMode": the strandMode of the sample(s) following GAlignmentPairs class definition. If all samples have the same strandMode, the length of the vector is 1. It can take values: NA (library is not strand-specific), 1 (strand of pair is strand of its first alignment), 2 (strand of pair is strand of its second alignment) or "ambiguous" (additional category used here for samples not fitting any of the three previous categories). See "Details" section below to know the classification criteria, as well as to how interpret results for single-end data.
- "Strandedness": data.frame with one row per sample and 3 columns. "strandMode1": proportion of alignments aligned to the same strand than a transcript according to the strand of its first alignment. "strandMode2": proportion of alignments aligned to the same strand than a transcript according to the strand of its second alignment. "ambiguous": alignments aligned to regions with transcripts in both strands.

A vector of integer values, NA, 1, or 2,

A SummarizedExperiment with three assays:

- "strness": contains strandedness values for each feature and sample.
- "counts": number of reads aligning to each feature on the same strand (according to strandMode).
- "counts_invstrand": number of reads aligning to each feature but on the opposite strand (according to strandMode).

Examples

```
library(gDNAinRNAseqData)

library(TxDb.Hsapiens.UCSC.hg38.knownGene)
txdb <- TxDb.Hsapiens.UCSC.hg38.knownGene

# Retrieving BAM files
bamfiles <- LiYu22subsetBAMfiles()
bamfiles <- bamfiles[c(1,7)] # using a subset of samples
```

```
strandM <- identifyStrandMode(bamfiles, txdb, singleEnd=FALSE)
strandM$strandMode
head(strandM$Strandedness)


strnessdat <- data.frame(strandMode1=c(0.91, 0.92, 0.93),
                         strandMode2=c(0.09, 0.08, 0.07))
classifyStrandMode(strnessdat)

features <- range(exonsBy(txdb, by="gene"))
features <- features[1:100]
sByFeature <- strnessByFeature(bamfiles, features, singleEnd=FALSE,
                               strandMode=2L)
sByFeature
```

Index

∗ **package**
    gDNAx-package, 2

allStrandModes *(*gDNAx-class*)*, 10
allStrandModes,gDNAx-method *(*gDNAx-class*)*, 10
AnnotationHub, *7*

BamFile, *14, 15*
BamFileList, *11*, *14, 15*
BiocParallelParam, *4*, *7*, *9*, *15*

classifyStrandMode, *2*
classifyStrandMode *(*identifyStrandMode*)*, 13

exonsBy, *7*, *15*

filterBAMtx, 3, *3*, *8*
filterBAMtxFlag, *3*
filterBAMtxFlag *(*filterBAMtx*)*, 3

GAlignmentPairs, *7*, *10, 11*, *15, 16*
gDNAdx, *2*, 6, *9, 10*, *12*
gDNAtx, 8
gDNAx, *8–11*
gDNAx *(*gDNAx-package*)*, 2
gDNAx-class, 10
gDNAx-package, 2
getDx, *2*
getDx *(*gDNAx-class*)*, 10
getDx,gDNAx-method *(*gDNAx-class*)*, 10
getIgc *(*gDNAx-class*)*, 10
getIgc,gDNAx-method *(*gDNAx-class*)*, 10
getInt *(*gDNAx-class*)*, 10
getInt,gDNAx-method *(*gDNAx-class*)*, 10

identifyStrandMode, 13

keepStandardChromosomes, *7*, *11*, *14*

list, *16*

OverlapEncodings, *5*

plot, *2*, *8*
plot,gDNAx,ANY-method *(*gDNAdx*)*, 6
plotAlnOrigins *(*gDNAdx*)*, 6
plotFrgLength *(*gDNAdx*)*, 6

ScanBamParam, *9*
SerialParam, *4*, *7*, *9*, *15*
show *(*gDNAx-class*)*, 10
show,gDNAx-method *(*gDNAx-class*)*, 10
singleEnd *(*gDNAx-class*)*, 10
singleEnd,gDNAx-method *(*gDNAx-class*)*, 10
strandedness, *2*, *15*
strandedness *(*gDNAx-class*)*, 10
strandedness,BamFileList-method *(*identifyStrandMode*)*, 13
strandedness,character-method *(*identifyStrandMode*)*, 13
strandedness,gDNAx-method *(*gDNAx-class*)*, 10
strandMode *(*gDNAx-class*)*, 10
strandMode,gDNAx-method *(*gDNAx-class*)*, 10
strandMode<- *(*gDNAx-class*)*, 10
strandMode<-,gDNAx-method *(*gDNAx-class*)*, 10
strnessByFeature *(*identifyStrandMode*)*, 13
SummarizedExperiment, *16*

testBAMtxFlag *(*filterBAMtx*)*, 3
TxDb, *11*