Package 'SGSeq'

April 10, 2015

Type Package

Title Prediction, quantification and visualization of alternative transcript events from RNA-seq data

Version 1.0.6

Author Leonard Goldstein

Maintainer Leonard Goldstein <goldstel@gene.com>

Description RNA-seq data are informative for the analysis of known and novel transcript isoforms. While the short length of RNA-seq reads limits the ability to predict and quantify full-length transcripts, short read data are well suited for the analysis of individual alternative transcripts events (e.g. inclusion or skipping of a cassette exon). The SGSeq package enables the prediction, quantification and visualization of alternative transcript events from BAM files.

License Artistic-2.0

LazyData yes

Depends BiocParallel, GenomicRanges, IRanges, methods

Imports AnnotationDbi, BiocGenerics, Biostrings, GenomicAlignments, GenomicFeatures, GenomeInfoDb, igraph, parallel, Rsamtools, rtracklayer, S4Vectors (>= 0.2.3)

Suggests BiocStyle, knitr, TxDb.Hsapiens.UCSC.hg19.knownGene

VignetteBuilder knitr

biocViews AlternativeSplicing, RNASeq, Transcription

R topics documented:

`analyzeFeatures` *Analysis of splice graph features from BAM files*

Description

High-level function for the prediction and quantification of splice junctions, exon bins and splice sites from BAM files.

Usage

```
analyzeFeatures(sample_info, which = NULL, features = NULL,
  predict = is.null(features), alpha = 2, psi = 0.1, beta = 0.2,
  gamma = 0.2, min_n_sample = 1, min_overhang = NA, annotation = NULL,
  cores_per_sample = 1, BPPARAM = MulticoreParam(1))
```

Arguments

`sample_info`	`data.frame` with sample information including mandatory character columns "sample_name" and "file_bam".
`which`	`GRanges` of genomic regions to be considered for feature prediction, passed to `ScanBamParam`
`features`	`TxFeatures` or `SGFeatures` object
`predict`	Logical indicating whether transcript features should be predicted from BAM files
`alpha`	Minimum FPKM required for a splice junction to be included
`psi`	Minimum splice frequency required for a splice junction to be included

beta	Minimum relative coverage required for an internal exon to be included
gamma	Minimum relative coverage required for a terminal exon to be included
min_n_sample	Minimum number of samples a feature must be observed in to be included
min_overhang	After merging, all or a subset of terminal exons sharing a splice site with an internal exon are removed. min_overhang specifies the minimum overhang required for a terminal exon to be included. Use NA to remove all terminal exons sharing a splice site with an internal exon. Use NULL to disable filtering (not usually recommended, as this will result in terminal exon predictions for each splice site; disabling filtering is useful if results are subsequently merged with other predictions and filtering is postponed until after the merging step).
annotation	TxFeatures object used for annotation
cores_per_sample	Number of cores per sample
BPPARAM	BiocParallelParam for processing samples in parallel, defaults to MulticoreParam(1)

Details

If alignment information is not included in sample_info, it is obtained directly from BAM files with getBamInfo.

Splice junctions and exons are predicted from BAM files with predictTxFeatures.

Known features can be provided as TxFeatures or SGFeatures via argument features.

If features is not NULL and predict is TRUE, known features are augmented with predictions.

Known and/or predicted transcript features are converted to splice graph features. For details, see convertToSGFeatures.

Optionally, splice graph features can be annotated with respect to a TxFeatures object provided via argument annotation. For details, see the help page for function annotate.

Finally, compatible fragment counts for splice graph features are obtained from BAM files with getSGFeatureCounts.

Value

SGFeatureCounts object

Author(s)

Leonard Goldstein

Examples

```
dir <- system.file("extdata", package = "SGSeq")
si$file_bam <- file.path(dir, "bams", si$file_bam)
sgfc <- analyzeFeatures(si, gr)
```

analyzeVariants *Analysis of transcript variants*

Description

High-level function for the analysis of transcript variants from splice graph features. Transcript variants are identified with `findTxVariants`. Representative counts and estimated variant frequencies are obtained with `getTxVariantCounts`.

Usage

```
analyzeVariants(object, maxnvariant = 20, cores = 1)
```

Arguments

`object`	`SGFeatureCounts` object
`maxnvariant`	If more than `maxnvariant` variants are identified in an event, the gene is skipped, resulting in a warning. Set to `NA` to include all genes.
`cores`	Number of cores available for parallel processing

Value

A `TxVariantCounts` object

Author(s)

Leonard Goldstein

Examples

```
txvc <- analyzeVariants(sgfc)
```

annotate *Annotation with respect to transcript features*

Description

Features in `query` are annotated with respect to transcript features in `subject`.

Usage

```
annotate(query, subject)
```

Arguments

query	SGFeatures, TxVariants, SGFeatureCounts or TxVariantCounts object
subject	TxFeatures object

Details

Annotation happens at two levels: For feature-centric annotation, query features are assigned all transcript names associated with any matching subject features. For gene-centric annotation, query features are assigned all gene names associated with subject features that are part of the same gene (connected component in the splice graph) as any matching query features.

Feature matching is performed as follows: Query splice junctions are matched with identical subject splice junctions. Query splice sites are matched with splice sites implied by subject splice junctions. Query exon bins are matched with overlapping subject exons. Spliced boundaries of query exon bins must match spliced subject exon boundaries. Query exon bins cannot extend across spliced subject exon boundaries.

Value

query with updated txName, geneName column slots

Author(s)

Leonard Goldstein

Examples

```
sgf_annotated <- annotate(sgf, txf)
txv_annotated <- annotate(txv, txf)
```

---

assays *Accessing and replacing assay data*

Description

Accessor and replacement functions for assay data.

Usage

```
FPKM(object)

FPKM(object) <- value

countsVariant5p(object)

countsVariant5p(object) <- value

countsVariant3p(object)
```

```
countsVariant3p(object) <- value

countsTotal5p(object)

countsTotal5p(object) <- value

countsTotal3p(object)

countsTotal3p(object) <- value

variantFreq(object)

variantFreq(object) <- value

## S4 method for signature SGFeatureCounts
counts(object)

## S4 replacement method for signature SGFeatureCounts
counts(object) <- value

## S4 method for signature SGFeatureCounts
FPKM(object)

## S4 replacement method for signature SGFeatureCounts
FPKM(object) <- value

## S4 method for signature TxVariantCounts
countsVariant5p(object)

## S4 replacement method for signature TxVariantCounts
countsVariant5p(object) <- value

## S4 method for signature TxVariantCounts
countsVariant3p(object)

## S4 replacement method for signature TxVariantCounts
countsVariant3p(object) <- value

## S4 method for signature TxVariantCounts
countsTotal5p(object)

## S4 replacement method for signature TxVariantCounts
countsTotal5p(object) <- value

## S4 method for signature TxVariantCounts
countsTotal3p(object)
```

```
## S4 replacement method for signature TxVariantCounts
countsTotal3p(object) <- value

## S4 method for signature TxVariantCounts
variantFreq(object)

## S4 replacement method for signature TxVariantCounts
variantFreq(object) <- value
```

Arguments

object	Object containing assay data
value	Replacement value

Details

Counts objects defined in the SGSeq package contain different types of assay data. For example, class SGFeatureCounts contains assays counts and FPKM.

To facilitate accessing and modifying assays, for each assay there exists a function, with name identical to the assay name, that can be used to access and modify it (see examples).

Value

Assay data for accessor functions, updated object for replacement functions.

Author(s)

Leonard Goldstein

Examples

```
x <- counts(sgfc)
y <- FPKM(sgfc)
```

---

convertToSGFeatures *Convert transcript features to splice graph features*

Description

Convert transcript features, predicted from RNA-seq data or extracted from transcript annotation, to splice graph features.

Usage

```
convertToSGFeatures(x, coerce = FALSE)
```

Arguments

x	TxFeatures object
coerce	Logical indicating whether transcript features should be coerced to splice graph features without disjoining exons and omitting splice donor and acceptor sites

Details

Splice junctions are unaltered. Exons are disjoined into non-overlapping exon bins. Adjacent exon bins without a splice site at the shared boundary are merged. All exon bins are assigned type "E".

Entries for splice donor and acceptor sites (positions immediately upstream and downstream of introns, respectively) are added.

In the returned SGFeatures object, column slots splice5p and splice3p indicate whether compatibility with an exon bin requires a fragment to be spliced at the 5' or 3' boundary, respectively. splice5p (splice3p) is TRUE if the first (last) position of the exon coincides with a splice acceptor (donor), and it is not adjacent to a neighboring exon bin.

Each feature is assigned a unique feature and gene identifier, stored in column slots featureID and geneID, respectively. The latter indicates features that belong to the same gene, represented by a connected component in the splice graph.

Value

An SGFeatures object

Author(s)

Leonard Goldstein

Examples

```
sgf <- convertToSGFeatures(txf)
```

---

convertToTxFeatures *Convert to TxFeatures object*

Description

Convert a TxDb object or a GRangesList of exons grouped by transcripts to a TxFeatures object.

Usage

```
convertToTxFeatures(x)
```

Arguments

x	TxDb object, or GRangesList of exons grouped by transcripts

Details

In the returned TxFeatures object, column slot txName is based on either tx_name in the TxDb object or names(x) if x is a GRangesList.

Value

A TxFeatures object

Author(s)

Leonard Goldstein

Examples

```
gr <- GRanges(c(1, 1), IRanges(c(1, 201), c(100, 300)), c("+", "+"))
grl <- split(gr, 1)
txf <- convertToTxFeatures(grl)
```

---

exportFeatures *Export to BED format*

Description

Export features to BED format. Splice sites are not included.

Usage

```
exportFeatures(features, file)
```

Arguments

features	TxFeatures or SGFeatures object
file	Character string specifying output file

Value

NULL

Author(s)

Leonard Goldstein

Examples

```
## Not run:
  exportFeatures(txf, "txf.bed")
  exportFeatures(sgf, "sgf.bed")

## End(Not run)
```

`filterTerminalExons` *Filter terminal exons that share splice sites with internal exons*

Description

Terminal exons that share their splice site with an internal exon are filtered based on the overhang with respect to the internal exon.

Usage

```
filterTerminalExons(features, min_overhang = NA, return_index = FALSE)
```

Arguments

`features`	`TxFeatures` object
`min_overhang`	For terminal exons sharing a splice site with an internal exon, minimum overhang required for terminal exon to be included. Use `NA` to remove all terminal exons sharing a splice site with an internal exon.
`return_index`	Logical indicating whether indices of retained features should be returned instead of filtered features

Details

`predictTxFeatures` predicts flanking terminal exons for each identified splice junction. Thus splice junctions are guaranteed to have flanking exons, even after filtering exons during merging with `mergeTxFeatures`. However, many predicted terminal exons share a splice site with predicted internal exons and are often contained within them. Many of these predictions are unlikely to be real terminal exons and are excluded before further analysis.

Value

`TxFeatures` object with filtered features, or indices of retained features if `return_index = TRUE`

Author(s)

Leonard Goldstein

Examples

```
txf_filtered <- filterTerminalExons(txf)
```

findTxVariants *Find transcript variants from splice graph*

Description

Find transcript variants from splice graph

Usage

```
findTxVariants(features, maxnvariant = 20, annotate_events = TRUE,
  cores = 1)
```

Arguments

features	SGFeatures object
maxnvariant	If more than maxnvariant variants are identified in an event, the gene is skipped, resulting in a warning. Set to NA to include all genes.
annotate_events	Logical indicating whether identified transcript variants should be annotated in terms of canonical events. For details see help page for annotateTxVariants.
cores	Number of cores available for parallel processing

Value

A TxVariants object

Author(s)

Leonard Goldstein

Examples

```
txv <- findTxVariants(sgf)
```

getBamInfo *Obtain alignment information from BAM files*

Description

Obtain paired-end status, median aligned read length, median aligned insert size and library size from BAM file.

Usage

```
getBamInfo(sample_info, yieldSize = NULL, BPPARAM = MulticoreParam(1))
```

Arguments

sample_info	data.frame with sample information including mandatory character columns "sample_name" and "file_bam".
yieldSize	Number of records used for obtaining alignment information, or NULL for all records
BPPARAM	BiocParallelParam for processing samples in parallel, defaults to MulticoreParam(1)

Details

Alignment information can be inferred from a subset of BAM records by setting the number of records via argument yieldSize. Note that library size can only be obtained if yieldSize is NULL.

Value

sample_info with additional columns "paired_end", "read_length", "frag_length", and "lib_size" if yieldSize is NULL

Author(s)

Leonard Goldstein

Examples

```
dir <- system.file("extdata", package = "SGSeq")
si$file_bam <- file.path(dir, "bams", si$file_bam)
si <- si[, c("sample_name", "file_bam")]
si_complete <- getBamInfo(si)
```

---

getSGFeatureCounts *Compatible counts for splice graph features from BAM files*

Description

Compatible counts are obtained for each sample and combined into an SGFeatureCounts object.

Usage

```
getSGFeatureCounts(sample_info, features, cores_per_sample = 1,
  BPPARAM = MulticoreParam(1))
```

Arguments

sample_info	data.frame with sample information. Required columns are "sample_name", "file_bam", "paired_end", "read_length", "frag_length" and "lib_size". Alignment information can be obtained with function getBamInfo.
features	SGFeatures object
cores_per_sample	Number of cores per sample
BPPARAM	BiocParallelParam for processing samples in parallel, defaults to MulticoreParam(1)

Value

An SGFeatureCounts object

Author(s)

Leonard Goldstein

Examples

```
dir <- system.file("extdata", package = "SGSeq")
si$file_bam <- file.path(dir, "bams", si$file_bam)
sgfc <- getSGFeatureCounts(si, sgf)
```

---

getTxVariantCounts *Representative counts and frequency estimates for transcript variants*

Description

For transcript variants, obtain counts of compatible fragments extending across the start and/or end of each variant. Variant frequencies are estimated based on representive counts.

Usage

```
getTxVariantCounts(object, variants)
```

Arguments

object	SGFeatureCounts object
variants	TxVariants object

Value

A TxVariantCounts object

Author(s)

Leonard Goldstein

Examples

```
txvc <- getTxVariantCounts(sgfc, txv)
```

`mergeTxFeatures` *Merge redundant features*

Description

Merge features, typically after feature prediction in multiple samples.

Usage

```
mergeTxFeatures(..., min_n_sample = 1)
```

Arguments

`...`	one or more `TxFeatures` objects, or a single list of `TxFeatures` objects
`min_n_sample`	Minimum number of samples a feature must be observed in to be included

Details

Merged features are the union of splice junctions and internal exons. For terminal exons with shared spliced boundary, the longest exon is retained.

Value

`TxFeatures` object with merged features

Author(s)

Leonard Goldstein

Examples

```
txf_merged <- mergeTxFeatures(txf, txf)
```

plotFeatures *Plot splice graph and heatmap of expression values*

Description

Plot splice graph and heatmap of expression values

Usage

```
plotFeatures(x, geneID = NULL, geneName = NULL, which = NULL,
  toscale = c("exon", "none", "gene"), color = "grey",
  color_novel = "red", color_alpha = 0.8, color_labels = FALSE,
  border = "fill", cexLab = 1, cexExon = 1, score = NULL,
  score_color = "darkblue", score_ylim = NULL, score_ypos = c(0.2, 0.1),
  score_nbin = 400, main = NULL, cexMain = 1, tx_view = FALSE,
  tx_dist = 0.1, tx_cex = 1, assay = "FPKM", include = c("junctions",
  "exons", "both"), transform = function(x) log2(x + 1), Rowv = NULL,
  distfun = dist, hclustfun = hclust, margin = 0.2,
  RowSideColors = NULL, square = FALSE, cexRow = 1, cexCol = 1,
  labRow = colnames(x), col = colorRampPalette(c("black", "gold"))(256),
  zlim = NULL, heightTopPanel = 0.3)
```

Arguments

x	SGFeatureCounts object
geneID	Single gene identifier used to subset x
geneName	Single gene name used to subset x
which	GRanges used to subset x
toscale	Controls which parts of the splice graph are drawn to scale. Possible values are "none" (exonic and intronic regions have constant length), "exon" (exonic regions are drawn to scale) and "gene" (both exonic and intronic regions are drawn to scale).
color	Color used for plotting the splice graph. Ignored if features elementMetadata column "color" is not NULL.
color_novel	Features with missing annotation are highlighted in color_novel. Ignored if features elementMetadata column "color" is not NULL.
color_alpha	Controls color transparency
color_labels	Logical indicating whether label colors should be the same as feature colors
border	Determines the color of exon borders, can be "fill" (same as exon color), "none" (no border) or a valid color name
cexLab	Scale factor for feature labels
cexExon	Scale factor for exon height
score	RLeList containing nucleotide-level scores to be plotted with the splice graph

score_color	Color used for plotting scores
score_ylim	y-axis range used for plotting scores
score_ypos	Numeric vector of length two, indicating the vertical position and height of the score panel, specificed as fractions of the height of the plotting region
score_nbin	Number of bins for plotting scores
main	Plot title
cexMain	Scale factor for plot title
tx_view	Plot transcripts instead of splice graph (experimental)
tx_dist	Vertical distance between transcripts as fraction of height of plotting region
tx_cex	Scale factor for transcript labels
assay	Name of assay to be plotted in the heatmap
include	Include "exons", "junctions" or "both" in the heatmap
transform	Transformation applied to assay data
Rowv	Determines order of rows. Either a vector of values used to reorder rows, or NA to suppress reordering, or NULL for hierarchical clustering.
distfun	Distance function used for hierarchical clustering of rows (samples)
hclustfun	Clustering function used for hierarchical clustering of rows (samples)
margin	Width of right-hand margin as fraction of width of the graphics device. Ignored if square is TRUE.
RowSideColors	Character vector (or list of character vectors) with length(s) equal to ncol(x) containing color names for horizontal side bars for sample annotation
square	Logical, if TRUE margins are set such that cells in the heatmap are square
cexRow	Scale factor for row (sample) labels
cexCol	Scale factor for column (feature) labels
labRow	Character vector of row (sample) labels
col	Heatmap colors
zlim	Range of values for which colors should be plotted, if NULL range of finite values
heightTopPanel	Height of top panel as fraction of height of the graphics device

Value

Return value of plotSpliceGraph

Author(s)

Leonard Goldstein

Examples

```
## Not run:
  sgfc_annotated <- annotate(sgfc, txf)
  plotFeatures(sgfc_annotated)

## End(Not run)
```

plotSpliceGraph *Plot splice graph*

Description

Plot splice graph implied by splice junctions and exon bins.

Usage

```
plotSpliceGraph(x, geneID = NULL, geneName = NULL, eventID = NULL,
  which = NULL, toscale = c("exon", "none", "gene"), label = c("id",
  "name", "label", "none"), color = "grey", color_novel = "red",
  color_alpha = 0.8, color_labels = FALSE, border = "fill", cexLab = 1,
  cexExon = 1, score = NULL, score_color = "darkblue",
  score_ylim = NULL, score_ypos = c(0.2, 0.1), score_nbin = 400,
  main = NULL, cexMain = 1, tx_view = FALSE, tx_dist = 0.2,
  tx_cex = 1, asp = 1)
```

Arguments

x	SGFeatures or TxVariants object
geneID	Single gene identifier used to subset x
geneName	Single gene name used to subset x
eventID	Single event identifier used to subset x
which	GRanges used to subset x
toscale	Controls which parts of the splice graph are drawn to scale. Possible values are "none" (exonic and intronic regions have constant length), "exon" (exonic regions are drawn to scale) and "gene" (both exonic and intronic regions are drawn to scale).
label	Format of exon/splice junction labels, possible values are "id" (format E1,... J1,...), "name" (format type:chromosome:start-end:strand), "label" for labels specified in elementMetadata column "label", or "none" for no labels.
color	Color used for plotting the splice graph. Ignored if features elementMetadata column "color" is not NULL.
color_novel	Features with missing annotation are highlighted in color_novel. Ignored if features elementMetadata column "color" is not NULL.
color_alpha	Controls color transparency
color_labels	Logical indicating whether label colors should be the same as feature colors
border	Determines the color of exon borders, can be "fill" (same as exon color), "none" (no border) or a valid color name
cexLab	Scale factor for feature labels
cexExon	Scale factor for exon height
score	RLeList containing nucleotide-level scores to be plotted with the splice graph

score_color	Color used for plotting scores
score_ylim	y-axis range used for plotting scores
score_ypos	Numeric vector of length two, indicating the vertical position and height of the score panel, specificed as fractions of the height of the plotting region
score_nbin	Number of bins for plotting scores
main	Plot title
cexMain	Scale factor for plot title
tx_view	Plot transcripts instead of splice graph (experimental)
tx_dist	Vertical distance between transcripts as fraction of height of plotting region
tx_cex	Scale factor for transcript labels
asp	Aspect ratio of graphics region

Details

By default, splice graph feature color is determined by annotation (see arguments `color`, `color_novel`) and labels are generated automatically (see argument `label`).

Alternatively, colors and labels can be specified via elementMetadata columns "color" and "label", respectively.

Value

`data.frame` with plotting information for exons and splice junctions in the splice graph

Author(s)

Leonard Goldstein

Examples

```
## Not run:
  sgf_annotated <- annotate(sgf, txf)
  plotSpliceGraph(sgf_annotated)

## End(Not run)
## Not run:
  txv_annotated <- annotate(txv, txf)
  plotSpliceGraph(txv_annotated)

## End(Not run)
```

plotVariants *Plot splice graph and heatmap of variant frequencies*

Description

Plot splice graph and heatmap of variant frequencies

Usage

```
plotVariants(x, eventID = NULL, toscale = c("exon", "none", "gene"),
  color = "grey", color_novel = "red", color_alpha = 0.8,
  color_labels = FALSE, border = "fill", cexLab = 1, cexExon = 1,
  score = NULL, score_color = "darkblue", score_ylim = NULL,
  score_ypos = c(0.2, 0.1), score_nbin = 400, main = NULL, cexMain = 1,
  tx_view = FALSE, tx_dist = 0.1, tx_cex = 1, transform = function(x)
  asin(sqrt(x)), Rowv = NULL, distfun = dist, hclustfun = hclust,
  margin = 0.2, RowSideColors = NULL, square = FALSE, cexRow = 1,
  cexCol = 1, labRow = colnames(x), col = colorRampPalette(c("black",
  "gold"))(256), zlim = NULL, heightTopPanel = 0.3,
  expand_variants = FALSE)
```

Arguments

x	TxVariantCounts object
eventID	Single event identifier used to subset x
toscale	Controls which parts of the splice graph are drawn to scale. Possible values are "none" (exonic and intronic regions have constant length), "exon" (exonic regions are drawn to scale) and "gene" (both exonic and intronic regions are drawn to scale).
color	Color used for plotting the splice graph. Ignored if features elementMetadata column "color" is not NULL.
color_novel	Features with missing annotation are highlighted in color_novel. Ignored if features elementMetadata column "color" is not NULL.
color_alpha	Controls color transparency
color_labels	Logical indicating whether label colors should be the same as feature colors
border	Determines the color of exon borders, can be "fill" (same as exon color), "none" (no border) or a valid color name
cexLab	Scale factor for feature labels
cexExon	Scale factor for exon height
score	RLeList containing nucleotide-level scores to be plotted with the splice graph
score_color	Color used for plotting scores
score_ylim	y-axis range used for plotting scores

score_ypos	Numeric vector of length two, indicating the vertical position and height of the score panel, specificed as fractions of the height of the plotting region
score_nbin	Number of bins for plotting scores
main	Plot title
cexMain	Scale factor for plot title
tx_view	Plot transcripts instead of splice graph (experimental)
tx_dist	Vertical distance between transcripts as fraction of height of plotting region
tx_cex	Scale factor for transcript labels
transform	Transformation applied to variant frequencies
Rowv	Determines order of rows. Either a vector of values used to reorder rows, or NA to suppress reordering, or NULL for hierarchical clustering.
distfun	Distance function used for hierarchical clustering of rows (samples)
hclustfun	Clustering function used for hierarchical clustering of rows (samples)
margin	Width of right-hand margin as fraction of width of the graphics device. Ignored if square is TRUE.
RowSideColors	Character vector (or list of character vectors) with length(s) equal to ncol(x) containing color names for horizontal side bars for sample annotation
square	Logical, if TRUE margins are set such that cells in the heatmap are square
cexRow	Scale factor for row (sample) labels
cexCol	Scale factor for column (feature) labels
labRow	Character vector of row (sample) labels
col	Heatmap colors
zlim	Range of values for which colors should be plotted, if NULL range of finite values
heightTopPanel	Height of top panel as fraction of height of the graphics device
expand_variants	Experimental option - leave set to FALSE

Value

Return value of plotSpliceGraph

Author(s)

Leonard Goldstein

Examples

```
## Not run:
  txvc_annotated <- annotate(txvc, txf)
  plotVariants(txvc_annotated)

## End(Not run)
```

predictTxFeatures *Transcript feature prediction from BAM files*

Description

Transcript features are predicted for each sample and subsequently merged across samples. Terminal exons that share a splice site with internal exons are filtered. For details see the help pages for predictTxFeaturesPerSample, mergeTxFeatures and filterTerminalExons.

Usage

```
predictTxFeatures(sample_info, which = NULL, alpha = 2, psi = 0,
  beta = 0.2, gamma = 0.2, min_junction_count = NULL, min_n_sample = 1,
  min_overhang = NA, cores_per_sample = 1, BPPARAM = MulticoreParam(1))
```

Arguments

sample_info	data.frame with sample information. Required columns are "sample_name", "file_bam", "paired_end", "read_length", "frag_length" and "lib_size". Alignment information can be obtained with function getBamInfo.
which	GRanges of genomic regions to be considered for feature prediction, passed to ScanBamParam
alpha	Minimum FPKM required for a splice junction to be included. Internally, FPKMs are converted to counts, requiring arguments read_length, frag_length and lib_size. alpha is ignored if argument min_junction_count is specified.
psi	Minimum splice frequency required for a splice junction to be included
beta	Minimum relative coverage required for an internal exon to be included
gamma	Minimum relative coverage required for a terminal exon to be included
min_junction_count	Minimum fragment count required for a splice junction to be included. If specified, argument alpha is ignored.
min_n_sample	Minimum number of samples a feature must be observed in to be included
min_overhang	After merging, all or a subset of terminal exons sharing a splice site with an internal exon are removed. min_overhang specifies the minimum overhang required for a terminal exon to be included. Use NA to remove all terminal exons sharing a splice site with an internal exon. Use NULL to disable filtering (not usually recommended, as this will result in terminal exon predictions for each splice site; disabling filtering is useful if results are subsequently merged with other predictions and filtering is postponed until after the merging step).
cores_per_sample	Number of cores per sample
BPPARAM	BiocParallelParam for processing samples in parallel, defaults to MulticoreParam(1)

Value

A TxFeatures object

Author(s)

Leonard Goldstein

Examples

```
dir <- system.file("extdata", package = "SGSeq")
si$file_bam <- file.path(dir, "bams", si$file_bam)
txf <- predictTxFeatures(si, gr)
```

SGFeatureCounts *Constructor function for S4 class* SGFeatureCounts

---

Description

Creates an instance of S4 class SGFeatureCounts for storing compatible splice graph feature counts.

Usage

```
SGFeatureCounts(x)
```

Arguments

x	SummarizedExperiment with SGFeatures as rowData and assays "counts", "FPKM"

Value

An SGFeatureCounts object

Author(s)

Leonard Goldstein

Examples

```
sgfc <- SGFeatureCounts()
```

SGFeatures *Constructor function for S4 class* SGFeatures

Description

Creates an instance of S4 class SGFeatures for storing splice graph features.

Usage

```
SGFeatures(x, type = mcols(x)$type, splice5p = mcols(x)$splice5p,
  splice3p = mcols(x)$splice3p, featureID = mcols(x)$featureID,
  geneID = mcols(x)$geneID, txName = mcols(x)$txName,
  geneName = mcols(x)$geneName)
```

Arguments

x	GRanges with known strand ("+", "-")
type	Character vector or factor taking values in J, E, D, A
splice5p	Logical vector indicating whether reads extending across the 5' boundary of an exon bin must be spliced at the boundary
splice3p	Logical vector indicating whether reads extending across the 3' boundary of an exon bin must be spliced at the boundary
featureID	Integer vector of feature IDs
geneID	Integer vector of gene IDs
txName	CharacterList of transcript names or NULL
geneName	CharacterList of gene names or NULL

Details

SGFeatures extend GRanges with column slot type specifying feature type. type is a factor with levels J (splice junction), E (exon bin), D (splice donor), A (splice acceptor).

splice5p and splice3p are logical vectors indicating whether reads extending across the 5' and 3' boundaries of an exon bin must be spliced at the boundary to be considered compatible with the exon bin.

featureID and geneID are integer vectors representing unique identifiers for features and genes (connected components in the splice graph).

txName and geneName are CharacterLists storing transcript and gene annotation, respectively.

Value

An SGFeatures object

Author(s)

Leonard Goldstein

Examples

```
sgf <- SGFeatures()
```

---

slots *Accessing and replacing column slots*

Description

Accessor and replacement functions for column slots.

Usage

```
type(object)

type(object) <- value

txName(object)

txName(object) <- value

geneName(object)

geneName(object) <- value

featureID(object)

featureID(object) <- value

geneID(object)

geneID(object) <- value

splice5p(object)

splice5p(object) <- value

splice3p(object)

splice3p(object) <- value

from(object)

from(object) <- value

to(object)
```

```
to(object) <- value

segmentID(object)

segmentID(object) <- value

variantID(object)

variantID(object) <- value

eventID(object)

eventID(object) <- value

closed5p(object)

closed5p(object) <- value

closed3p(object)

closed3p(object) <- value

variantType(object)

variantType(object) <- value

variantName(object)

variantName(object) <- value

featureID5p(object)

featureID5p(object) <- value

featureID3p(object)

featureID3p(object) <- value

## S4 method for signature Features
type(object)

## S4 method for signature Paths
type(object)

## S4 method for signature Counts
type(object)

## S4 replacement method for signature Features
```

```
type(object) <- value

## S4 replacement method for signature Paths
type(object) <- value

## S4 replacement method for signature Counts
type(object) <- value

## S4 method for signature Features
txName(object)

## S4 method for signature Paths
txName(object)

## S4 method for signature Counts
txName(object)

## S4 replacement method for signature Features
txName(object) <- value

## S4 replacement method for signature Paths
txName(object) <- value

## S4 replacement method for signature Counts
txName(object) <- value

## S4 method for signature Features
geneName(object)

## S4 method for signature Paths
geneName(object)

## S4 method for signature Counts
geneName(object)

## S4 replacement method for signature Features
geneName(object) <- value

## S4 replacement method for signature Paths
geneName(object) <- value

## S4 replacement method for signature Counts
geneName(object) <- value

## S4 method for signature SGFeatures
featureID(object)

## S4 method for signature Paths
```

```
featureID(object)

## S4 method for signature Counts
featureID(object)

## S4 replacement method for signature SGFeatures
featureID(object) <- value

## S4 replacement method for signature Paths
featureID(object) <- value

## S4 replacement method for signature Counts
featureID(object) <- value

## S4 method for signature SGFeatures
geneID(object)

## S4 method for signature Paths
geneID(object)

## S4 method for signature Counts
geneID(object)

## S4 replacement method for signature SGFeatures
geneID(object) <- value

## S4 replacement method for signature Paths
geneID(object) <- value

## S4 replacement method for signature Counts
geneID(object) <- value

## S4 method for signature SGFeatures
splice5p(object)

## S4 method for signature TxSegments
splice5p(object)

## S4 method for signature SGFeatureCounts
splice5p(object)

## S4 replacement method for signature SGFeatures
splice5p(object) <- value

## S4 replacement method for signature TxSegments
splice5p(object) <- value

## S4 replacement method for signature SGFeatureCounts
```

```
splice5p(object) <- value

## S4 method for signature SGFeatures
splice3p(object)

## S4 method for signature TxSegments
splice3p(object)

## S4 method for signature SGFeatureCounts
splice3p(object)

## S4 replacement method for signature SGFeatures
splice3p(object) <- value

## S4 replacement method for signature TxSegments
splice3p(object) <- value

## S4 replacement method for signature SGFeatureCounts
splice3p(object) <- value

## S4 method for signature Paths
segmentID(object)

## S4 method for signature TxVariantCounts
segmentID(object)

## S4 replacement method for signature Paths
segmentID(object) <- value

## S4 replacement method for signature TxVariantCounts
segmentID(object) <- value

## S4 method for signature Paths
from(object)

## S4 method for signature TxVariantCounts
from(object)

## S4 replacement method for signature Paths
from(object) <- value

## S4 replacement method for signature TxVariantCounts
from(object) <- value

## S4 method for signature Paths
to(object)

## S4 method for signature TxVariantCounts
```

```
to(object)

## S4 replacement method for signature Paths
to(object) <- value

## S4 replacement method for signature TxVariantCounts
to(object) <- value

## S4 method for signature TxVariants
eventID(object)

## S4 method for signature TxVariantCounts
eventID(object)

## S4 replacement method for signature TxVariants
eventID(object) <- value

## S4 replacement method for signature TxVariantCounts
eventID(object) <- value

## S4 method for signature TxVariants
variantID(object)

## S4 method for signature TxVariantCounts
variantID(object)

## S4 replacement method for signature TxVariants
variantID(object) <- value

## S4 replacement method for signature TxVariantCounts
variantID(object) <- value

## S4 method for signature TxVariants
closed5p(object)

## S4 method for signature TxVariantCounts
closed5p(object)

## S4 replacement method for signature TxVariants
closed5p(object) <- value

## S4 replacement method for signature TxVariantCounts
closed5p(object) <- value

## S4 method for signature TxVariants
closed3p(object)

## S4 method for signature TxVariantCounts
```

```
closed3p(object)

## S4 replacement method for signature TxVariants
closed3p(object) <- value

## S4 replacement method for signature TxVariantCounts
closed3p(object) <- value

## S4 method for signature TxVariants
variantName(object)

## S4 method for signature TxVariantCounts
variantName(object)

## S4 replacement method for signature TxVariants
variantName(object) <- value

## S4 replacement method for signature TxVariantCounts
variantName(object) <- value

## S4 method for signature TxVariants
variantType(object)

## S4 method for signature TxVariantCounts
variantType(object)

## S4 replacement method for signature TxVariants
variantType(object) <- value

## S4 replacement method for signature TxVariantCounts
variantType(object) <- value

## S4 method for signature TxVariants
featureID5p(object)

## S4 method for signature TxVariantCounts
featureID5p(object)

## S4 replacement method for signature TxVariants
featureID5p(object) <- value

## S4 replacement method for signature TxVariantCounts
featureID5p(object) <- value

## S4 method for signature TxVariants
featureID3p(object)

## S4 method for signature TxVariantCounts
```

```
featureID3p(object)

## S4 replacement method for signature TxVariants
featureID3p(object) <- value

## S4 replacement method for signature TxVariantCounts
featureID3p(object) <- value
```

Arguments

object	Object containing column slot
value	Replacement value

Details

S4 classes defined in the SGSeq package contain columns with information for each element in the object. For example, class TxFeatures contains a column type that indicates feature type. The specific columns contained in an object depend on its class.

To facilitate accessing and modifying columns, for each column there exists a function, with name identical to the column name, that can be used to access and modify it (see examples).

Value

Column value for accessor functions, updated object for replacement functions.

Author(s)

Leonard Goldstein

Examples

```
head(type(txf))
head(type(sgf))
```

TxFeatures *Constructor function for S4 class* TxFeatures

Description

Creates an instance of S4 class TxFeatures for storing transcript features.

Usage

```
TxFeatures(x, type = mcols(x)$type, txName = mcols(x)$txName,
  geneName = mcols(x)$geneName)
```

Arguments

x	GRanges with known strand ("+", "-")
type	Character vector or factor taking values in J, I, F, L, U
txName	CharacterList of transcript names or NULL
geneName	CharacterList of gene names or NULL

Details

TxFeatures extend GRanges with column slot type specifying feature type. type is a factor with levels J (splice junction), I (internal exon), F (5' terminal exon), L (3' terminal exon), U (unspliced transcript).

txName and geneName are CharacterLists storing transcript and gene annotation, respectively.

Value

A TxFeatures object

Author(s)

Leonard Goldstein

Examples

```
gr <- GRanges(1, IRanges(101, 200), "+")
txf <- TxFeatures(gr, type = "J")
```

TxVariantCounts *Constructor function for S4 class* SGFeatureCounts

---

Description

Creates an instance of S4 class TxVariantCounts for storing representative transcript variant counts.

Usage

```
TxVariantCounts(x)
```

Arguments

x	SummarizedExperiment with TxVariants as rowData and appropriate assays

Value

A TxVariantCounts object

Author(s)

Leonard Goldstein

Examples

```
txvc <- TxVariantCounts()
```

TxVariants *Constructor function for S4 class* TxVariants

Description

Creates an instance of S4 class TxVariants for storing transcript variants.

Usage

```
TxVariants(x)
```

Arguments

x	GRangesList of SGFeatures with appropriate outer elementMetadata columns

Value

A TxVariants object

Author(s)

Leonard Goldstein

Examples

```
txv <- TxVariants()
```

Index