Package ‘SGSeq’

March 24, 2015

Type Package

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

Version 1.1.19

Author Leonard Goldstein

Maintainer Leonard Goldstein <goldstel@geneNcom>

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 splice events (e.g. inclusion or skipping of a cassette exon). The SGSeq package enables the prediction, quantification and visualization of splice 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

NeedsCompilation no

R topics documented:

analyzeFeatures ................................................. 2
analyzeVariants ................................................. 4
annotate ......................................................... 5
assays .................................................................. 6
convertToSGFeatures ............................................ 8
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,
max_complexity = 20, verbose = FALSE, 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
analyzeFeatures

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, terminal exons are processed. For terminal exons sharing a splice site with an internal exon, minimum overhang required for terminal exons to be included. For remaining terminal exons overlapping other exons, minimum overhang required to suppress trimming. Use NA to remove all terminal exons sharing a splice site with an internal exon and trim all remaining terminal exons overlapping other exons. Use NULL to disable processing (disabling processing is useful if results are subsequently merged with other predictions and processing is postponed until after the merging step).

annotation TxFeatures object used for annotation

max_complexity Maximum allowed complexity. If a locus exceeds this threshold, it is skipped, resulting in a warning message. Here complexity is defined as the maximum number of unique filtered splice junctions overlapping a given position in a locus. High complexity regions are often due to spurious read alignments and can significantly slow down processing. To disable this filter, set to NA.

verbose If TRUE, print messages indicating progress.

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
analyzeVariants

Author(s)
Leonard Goldstein

Examples

```r
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

```r
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

```r
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)
**Description**

Accessor and replacement functions for assay data.

**Usage**

```r
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)
```
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
convertToSGFeatures

Examples

```r
x <- counts(sgf)
y <- FPKM(sgf)
```

Description

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

Usage

```r
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

```r
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**  
If `x` is a GRangesList, transcript names and gene names can be specified as character vectors in elementMetadata columns txName and geneName, respectively. If missing, transcript names are based on names(x).

**Value**  
A TxFeatures object

**Author(s)**  
Leonard Goldstein

**Examples**

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

---

**exportFeatures**  
*Export to BED format*

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

**Usage**  
`exportFeatures(features, file)`
findTxVariants

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)

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
getBamInfo

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, counts_only = FALSE, 
cores_per_sample = 1, verbose = FALSE, 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
- **counts_only**: Logical indicating only counts should be returned
- **cores_per_sample**: Number of cores per sample
- **verbose**: If TRUE, print messages indicating progress.
- **BPPARAM**: BiocParallelParam for processing samples in parallel, defaults to MulticoreParam(1)

Value

An SGFeatureCounts object or integer matrix of counts if counts_only = TRUE

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 representative counts.

Usage
getTxVariantCounts(object, variants, cores = 1)

Arguments
- object  SGFeatureCounts object
- variants  TxVariants object
- cores  Number of cores available for parallel processing

Value
A TxVariantCounts object

Author(s)
Leonard Goldstein

Examples
txvc <- getTxVariantCounts(sgfc, txv)

makeSGFeatureCounts  Create SGFeatureCounts object

Description
Create SGFeatureCounts object from rowData, colData and counts.

Usage
makeSGFeatureCounts(rowData, colData, counts)

Arguments
- rowData  An SGFeatures object
- colData  Data frame with sample information
- counts  Integer matrix of counts
Value

An SGFeatureCounts object

Author(s)

Leonard Goldstein

Examples

```r
sgfc <- makeSGFeatureCounts(sgf, si, matrix(0L, length(sgf), nrow(si)))
```

makeVariantNames  

Create interpretable transcript variant names

Description

This function creates interpretable transcript variant names taking the format GENE_EVENT_VARIANT/ORDER_TYPE. GENE is based on geneName if available, and geneID otherwise. EVENT and VARIANT enumerate events and variants for the same gene and event, respectively. ORDER indicates the total number of variants in the same event (e.g. 1/2 refers to the first out of two transcript variants in the event). TYPE is based on variantType.

Usage

```r
makeVariantNames(variants)
```

Arguments

variants  

TxVariants object

Value

Character vector with transcript variant names

Author(s)

Leonard Goldstein

Examples

```r
makeVariantNames(txv)
```


### mergeTxFeatures

**Merge redundant features**

**Description**

Merge features, typically after feature prediction in multiple samples.

**Usage**

```r
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**

```r
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, track = NULL, track_color = "darkblue", track ylim = NULL, track ypos = c(0.2, 0.1), track nbins = 400, track_summary = mean, 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) }, Rowl = 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
track RLeList containing nucleotide-level scores or a GRangesList to be plotted with the splice graph
track_color Color used for plotting tracks
track ylim y-axis range used for plotting scores
track ypos Numeric vector of length two, indicating the vertical position and height of the track panel, specified as fractions of the height of the plotting region
track nbins Number of bins for plotting scores
track_summary Function used to calculate per-bin score summaries
main Plot title
plotFeatures

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

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

## End(Not run)```

plotSpliceGraph

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, track = NULL, track_color = "darkblue",
track ylim = NULL, track ypos = c(0.2, 0.1), track nbin = 400,
track_summary = mean, 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
plotSpliceGraph

track  RleList containing nucleotide-level scores or a GRangesList to be plotted with the splice graph
track_color  Color used for plotting tracks
track_ylim  y-axis range used for plotting scores
track_ypos  Numeric vector of length two, indicating the vertical position and height of the track panel, specified as fractions of the height of the plotting region
track_nbin  Number of bins for plotting scores
track_summary  Function used to calculate per-bin score summaries
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

```r
## 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,
  track = NULL, track_color = "darkblue", track_ylim = NULL,
  track_ypos = c(0.2, 0.1), track_nbin = 400, main = NULL, cexMain = 1,
  tx_view = FALSE, tx_dist = 0.1, tx_cex = 1, transform = function(x) {
    x }, Roww = 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 = c(0, 1), 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

track      RleList containing nucleotide-level scores or a GRangesList to be plotted with
            the splice graph

track_color Color used for plotting tracks

track_ylim y-axis range used for plotting scores
plotVariants

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

Value

Return value of plotSpliceGraph

Author(s)

Leonard Goldstein

Examples

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

## End(Not run)
```
predictTxFeatures  \hspace{1cm} Transcript feature prediction from BAM files

Description

Transcript features are predicted for each sample and merged across samples. Subsequently, terminal exons are filtered and trimmed (if applicable). For details, see the help pages for \texttt{predictTxFeaturesPerSample}, \texttt{mergeTxFeatures}, and \texttt{processTerminalExons}.

Usage

\begin{verbatim}
predictTxFeatures(sample_info, which = NULL, alpha = 2, psi = 0,
beta = 0.2, gamma = 0.2, min_junction_count = NULL,
max_complexity = 20, min_n_sample = 1, min_overhang = NA,
verbose = FALSE, cores_per_sample = 1, BPPARAM = MulticoreParam(1))
\end{verbatim}

Arguments

- \texttt{sample_info}  \hspace{1cm} 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 \texttt{getBamInfo}.
- \texttt{which}  \hspace{1cm} GRanges of genomic regions to be considered for feature prediction, passed to \texttt{ScanBamParam}.
- \texttt{alpha}  \hspace{1cm} Minimum FPKM required for a splice junction to be included. Internally, FPKMs are converted to counts, requiring arguments \texttt{read_length}, \texttt{frag_length} and \texttt{lib_size}. \texttt{alpha} is ignored if argument \texttt{min_junction_count} is specified.
- \texttt{psi}  \hspace{1cm} Minimum splice frequency required for a splice junction to be included
- \texttt{beta}  \hspace{1cm} Minimum relative coverage required for an internal exon to be included
- \texttt{gamma}  \hspace{1cm} Minimum relative coverage required for a terminal exon to be included
- \texttt{min_junction_count}  \hspace{1cm} Minimum fragment count required for a splice junction to be included. If specified, argument \texttt{alpha} is ignored.
- \texttt{max_complexity}  \hspace{1cm} Maximum allowed complexity. If a locus exceeds this threshold, it is skipped, resulting in a warning message. Here complexity is defined as the maximum number of unique filtered splice junctions overlapping a given position in a locus. High complexity regions are often due to spurious read alignments and can significantly slow down processing. To disable this filter, set to NA.
- \texttt{min_n_sample}  \hspace{1cm} Minimum number of samples a feature must be observed in to be included
- \texttt{min_overhang}  \hspace{1cm} After merging, terminal exons are processed. For terminal exons sharing a splice site with an internal exon, minimum overhang required for terminal exons to be included. For remaining terminal exons overlapping other exons, minimum overhang required to suppress trimming. Use NA to remove all terminal exons sharing a splice with an internal exon and trim all remaining terminal exons overlapping other exons. Use NULL to disable processing (disabling processing is useful if results are subsequently merged with other predictions and processing is postponed until after the merging step).
processTerminalExons

   verbose  If TRUE, print messages indicating progress.
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)

processTerminalExons  Process predicted terminal exons

Description

Prediction of transcript starts and ends from RNA-seq data is imprecise. Due to the employed prediction strategy, further processing of predicted terminal exons is required to avoid common artifacts.

Usage

processTerminalExons(features, min_overhang = NA)

Arguments

   features  TxFeatures object
   min_overhang  For terminal exons sharing a splice site with an internal exon, minimum overhang required for terminal exons to be included. For remaining terminal exons overlapping other exons, minimum overhang required to suppress trimming. Use NA to remove all terminal exons sharing a splice with an internal exon and trim all remaining terminal exons overlapping other exons.
Details

Processing of terminal exon predictions is done in two steps: (1) terminal exons that share a splice site with an internal exon are filtered, and (2) remaining terminal exons that overlap other exons are trimmed.

`predictTxFeatures` predicts flanking terminal exons for each identified splice junction. This ensures that each splice junction has a flanking exon after merging with `mergeTxFeatures`. This approach results in many predicted terminal exons that share a splice site with predicted internal exons (often contained within them or with a short overhang due to incorrect alignments). Most of these are not real terminal exons and are filtered before further analysis. Filtering based on the overhang is possible via argument `min_overhang`.

Some of the remaining predicted terminal exons overlap other exons such that their unspliced boundary shows a short overhang with respect to a spliced boundary of the overlapping exon. Often these exon extensions into an intron are due to incorrect alignments. Terminal exons with overhang smaller than `min_overhang` are trimmed such that their trimmed unspliced boundary coincides with the spliced boundary of the overlapping exon.

Value

`TxFeatures` object with processed features

Author(s)

Leonard Goldstein

Examples

```r
  txf_processed <- processTerminalExons(txf)
```

---

**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
slots

```r
## 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 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
TxFeatures

Constructor function for S4 class TxFeatures

Description

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

Usage

```r
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).

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

Value

A TxFeatures object

Author(s)

Leonard Goldstein

Examples

```r
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**  
```r
TxVariantCounts(x)
```

**Arguments**  
- `x`  
  SummarizedExperiment with `TxVariants` as rowData and appropriate assays

**Value**  
A `TxVariantCounts` object

**Author(s)**  
Leonard Goldstein

**Examples**  
```r
txvc <- TxVariantCounts()
```

---

**TxVariants**  
*Constructor function for S4 class TxVariants*

---

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

**Usage**  
```r
TxVariants(x)
```

**Arguments**  
- `x`  
  GRangesList of `SgFeatures` with appropriate outer elementMetadata columns

**Value**  
A `TxVariants` object
Author(s)

Leonard Goldstein

Examples

txv <- TxVariants()
Index

analyzeFeatures, 2
analyzeVariants, 4
annotate, 3, 5
annotateTxVariants, 10
assays, 6
closed3p (slots), 26
closed3p, TxVariantCounts-method (slots), 26
closed3p, TxVariants-method (slots), 26
closed3p<- (slots), 26
closed3p<-, TxVariantCounts-method (slots), 26
closed3p<-, TxVariants-method (slots), 26
closed5p (slots), 26
closed5p, TxVariantCounts-method (slots), 26
closed5p, TxVariants-method (slots), 26
closed5p<- (slots), 26
closed5p<-, TxVariantCounts-method (slots), 26
closed5p<-, TxVariants-method (slots), 26
eventID (slots), 26
eventID, TxVariantCounts-method (slots), 26
eventID, TxVariants-method (slots), 26
eventID<- (slots), 26
eventID<-, TxVariantCounts-method (slots), 26
eventID<-, TxVariants-method (slots), 26
exportFeatures, 9
featureID (slots), 26
featureID, Counts-method (slots), 26
featureID, Paths-method (slots), 26
featureID, SGFeatures-method (slots), 26
featureID3p (slots), 26
featureID3p, TxVariantCounts-method (slots), 26
featureID3p, TxVariants-method (slots), 26
featureID3p<- (slots), 26
featureID3p<-, TxVariantCounts-method (slots), 26
featureID3p<-, TxVariants-method (slots), 26
featureID5p (slots), 26
featureID5p, TxVariantCounts-method (slots), 26
featureID5p, TxVariants-method (slots), 26
convertToSGFeatures, 3, 8
convertToTxFeatures, 9
counts, SGFeatureCounts-method (assays), 6
counts<-, SGFeatureCounts-method (assays), 6
countsTotal3p (assays), 6
countsTotal3p, TxVariantCounts-method (assays), 6
countsTotal3p<- (assays), 6
countsTotal3p<-, TxVariantCounts-method (assays), 6
countsTotal5p (assays), 6
countsTotal5p, TxVariantCounts-method (assays), 6
countsTotal5p<- (assays), 6
countsTotal5p<-, TxVariantCounts-method (assays), 6
countsVariant3p (assays), 6
countsVariant3p, TxVariantCounts-method (assays), 6
countsVariant3p<- (assays), 6
countsVariant3p<-, TxVariantCounts-method (assays), 6
countsVariant5p (assays), 6
countsVariant5p, TxVariantCounts-method (assays), 6
countsVariant5p<- (assays), 6
countsVariant5p<-, TxVariantCounts-method (assays), 6

37
featureID5p, TxVariants-method (slots), 26
featureID5p<-(slots), 26
featureID5p<-, TxVariantCounts-method (slots), 26
featureID5p<-, TxVariants-method (slots), 26
featureID<-(slots), 26
featureID<-, Paths-method (slots), 26
featureID<-, Paths-counts-method (slots), 26
featureID<-, SGFeatures-method (slots), 26
findTxVariants, 4, 10
FPKM (assays), 6
FPKM, SGFeatureCounts-method (assays), 6
FPKM<-(assays), 6
FPKM<-, SGFeatureCounts-method (assays), 6
from (slots), 26
from, Paths-method (slots), 26
from, TxVariantCounts-method (slots), 26
from<-(slots), 26
from<-, Paths-method (slots), 26
from<-, TxVariantCounts-method (slots), 26
geneID (slots), 26
geneID, Counts-method (slots), 26
geneID, Paths-method (slots), 26
geneID, SGFeatures-method (slots), 26
geneID<-(slots), 26
geneID<-, Counts-method (slots), 26
geneID<-, Paths-method (slots), 26
geneID<-, SGFeatures-method (slots), 26
geneName (slots), 26
geneName, Counts-method (slots), 26
geneName, Features-method (slots), 26
geneName<-(slots), 26
geneName<-, Counts-method (slots), 26
geneName<-, Features-method (slots), 26
geneName<-, Paths-method (slots), 26
getBamInfo, 3, 11
getSGFeatureCounts, 3, 12
getTxVariantCounts, 4, 13
makeSGFeatureCounts, 13
makeVariantNames, 14
mergeTxFeatures, 15, 22
plotFeatures, 15
plotSpliceGraph, 18
plotVariants, 20
predictTxFeatures, 3, 22
predictTxFeaturesPerSample, 22
processTerminalExons, 22, 23
segmentID (slots), 26
segmentID, Paths-method (slots), 26
segmentID, TxVariantCounts-method (slots), 26
segmentID<-(slots), 26
segmentID<-, Paths-method (slots), 26
segmentID<-, TxVariantCounts-method (slots), 26
SGFeatureCounts, 24
SGFeatures, 25
slots, 26
splice3p (slots), 26
splice3p, SGFeatureCounts-method (slots), 26
splice3p<-(slots), 26
splice3p<-, SGFeatures-method (slots), 26
splice3p<-, TxSegments-method (slots), 26
splice5p (slots), 26
splice5p, SGFeatureCounts-method (slots), 26
splice5p, SGFeatures-method (slots), 26
splice5p, TxSegments-method (slots), 26
splice5p<-(slots), 26
splice5p<-, SGFeatures-method (slots), 26
splice5p<-, SGFeatures-method (slots), 26
to (slots), 26
to, TxVariantCounts-method (slots), 26
to<-(slots), 26
to<-, Paths-method (slots), 26
to<-, TxVariantCounts-method (slots), 26
TxFeatures, 34
txName (slots), 26
txName, Counts-method (slots), 26
txName, Features-method (slots), 26
txName, Paths-method (slots), 26
variantFreq (assays), 6
variantFreq-, TxVariantCounts-method (assays), 6
variantFreq<-, (assays), 6
variantFreq<-, TxVariantCounts-method (assays), 6
variantID (slots), 26
variantID, TxVariantCounts-method (slots), 26
variantID<-, (slots), 26
variantID<-, TxVariantCounts-method (slots), 26
variantID<-, TxVariants-method (slots), 26
variantName (slots), 26
variantName, TxVariantCounts-method (slots), 26
variantName, TxVariants-method (slots), 26
variantName<-, (slots), 26
variantName<-, TxVariantCounts-method (slots), 26
variantName<-, TxVariants-method (slots), 26
variantType (slots), 26
variantType, TxVariantCounts-method (slots), 26
variantType, TxVariants-method (slots), 26
variantType<-, (slots), 26
variantType<-, TxVariantCounts-method (slots), 26