Package 'casper'

October 19, 2011

Version 0.0.0

Date 2010-07-29

Title Characterization of Alternative Splicing based on Paired-End Reads

Author Manuel Kroiss, Camille Stephan-Otto, Almond Strocker, David Rossell

Maintainer David Rossell <david.rossell@irbbarcelona.org>

Depends R (>= 2.10.0), Biobase, IRanges, methods, gtools,GenomicFeatures, GenomicRanges

Suggests multicore

Description The package provides methodology to infer alternative splicing patterns for paired-end based on high-throughput sequencing data. Both quantification of known splicing variants and de novo splice variant discovery are considered. The approach is based on a flexible Bayesian model which estimates the RNA fragment size distribution, allows the user to set part of the data to missing values to remove artifacts, and is computationally efficient.

License GPL (>=2)

LazyLoad yes

biocViews Bioinformatics, GeneExpression, DifferentialExpression,Transcription, RNASeq, HighThroughputSequencing

R topics documented:

calcExp — *Calculate expression of gene variants*

Description

Calculate expression of gene variants

Usage

```
calcExp(distrs, genomeDB, pc)
```

Arguments

distrs	List of fragment distributions as generated by the "getDistrs" function
genomeDB	List of genomic annotations generated by the "procGenome" function
pc	Named vector of exon path counts

Value

Expression set with resulting values

Author(s)

Camille Stephan-Otto Attolini

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.
```

genPlot — *Plot exon structure and aligned reads for a given gene*

Description

Plot exon structure for all transcripts in a given gene and aligned reads

Usage

```
genPlot(goi, genomeDB, reads, exp)
```

Arguments

goi	ENTREZ id of gene of interest
genomeDB	List of annotations produced with the "procGenome" function
reads	RangedData object of aligned reads or fragments
exp	ExpressionSet object with expression values

Value

`gene`	IRangesList object with one IRanges per transcript and its exons
`exp`	Named integer vector with transcript expression for the gene of interest

Author(s)

Camille Stephan-Otto Attolini

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.
```

`getDistrs` *Compute fragment start and fragment length distributions*

Description

This function calculates fragment start distributions for reads aligned to genes with only one annotated variant and fragment length distribution for fragments aligned to long exons (>1000nt)

Usage

```
getDistrs(txs, exons, frags, mc.cores)
```

Arguments

`txs`	GRanges object with known transcripts
`exons`	GRangesList with annotated exons per transcript
`frags`	RangedData object defined by start and end of whole fragments
`mc.cores`	Number of cores to use in parallel processing (multicore package required)

Value

`stDis`	Numeric vector with relative fragment start positions
`lenDis`	Table with fragment counts for each existing length

Author(s)

Camille Stephan-Otto Attolini

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.
```

pathCounts *Compute exon path counts*

Description

Compute counts for exon paths visited by aligned reads

Usage

```
pathCounts(reads, exons)
```

Arguments

reads	RangeData object with aligned reads
exons	RangedData object with non-overlapping exons

Value

Named integer vector with counts of exon paths. Names are character strings built as ".exon1.exon2-exon3.exon4.", with dashes making the split between exons visited by left and right-end reads correspondingly.

Author(s)

Camille Stephan-Otto Attolini

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.
```

procBam *Process SAM/BAM files*

Description

Process paired-end data stored in SAM or BAM formats and read into RangedData objects. The Samtools package is required if files are in BAM format. For large files, sequencial processing by chromosome is possible in order to minimize RAM needs.

Usage

```
procBam(bamFileName, samtools, chrom, bam, parallel)
```

Arguments

bamFileName	Absolute path to SAM/BAM file
samtools	Absolute path to the directory where the samtools executable is
chrom	Chromosome to be processed, empty string ("") will process the complete genome
bam	0 for SAM format, 1 for BAM format
parallel	Set to TRUE if sequencial processing by chromosome is needed. TRUE will overlook the chrom option and process the complete genome.

Value

reads	A RangedData object with reads from fragments with both ends correctly aligned after splitting them by the corresponding CIGAR. Unique identifiers by fragment are stored.
frags	A RangedData objects with start and end of fragments with both reads aligned.

Author(s)

Camille Stephan-Otto Attolini

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.
```

---

procGenome *Download and format annotations for a given genome*

Description

Download USCS annotations for a given genome

Usage

```
procGenome(genome, mc.cores = mc.cores)
```

Arguments

genome	Genome version from UCSC (e.g. "hg19", "dm3")
mc.cores	Number of cores to use in parallel processing (multicore package required)

Details

This function generates all necessary annotation objects for subsequent functions.

Value

`gene`	Named chracter vector with mapping from gene ENTREZ id to transcript id.
`exonsNI`	RangedData object containing non overlapping exons.
`exons`	GRangesList object with original exon coordinates. List elements correspond to a known transcripts.
`txs`	GRanges object with original transcripts annotation.
`newTxs`	Named list mapping transcript id's to non-overlapping exons.
`exonmap`	Named list mapping original exon id's to non-overlapping exon id's.

Author(s)

Camille Stephan-Otto Attolini

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.
```

Index