Package 'SCANVIS'

February 13, 2019

Type Package

Title SCoring, ANnotating and VISualizing splicing signatures

Version 1.0

Date 2019-02-01

Author Phaedra Agius <pagius@nygenome.org>

Maintainer Phaedra Agius <pagius@nygenome.org>

Depends R(>= 3.1.0)

Description A tool for SCoring, ANnotating and VISualizing splice junctions

Imports IRanges, plotrix, rtracklayer, RCurl

NeedsCompilation no

R topics documented:

`SCANVIS-package` *SCoring, ANnotating and VISualizing splicing signatures*

Description

A tool for SCoring, ANnotating and VISualizing splice junctions

Details

Package:	SCANVIS
Type:	Package
Title:	SCoring, ANnotating and VISualizing splicing signatures
Version:	1.0
Date:	2019-02-01
Author:	Phaedra Agius <pagius@nygenome.org>
Maintainer:	Phaedra Agius <pagius@nygenome.org>
Depends:	R(>= 3.1.0)
Description:	A tool for SCoring, ANnotating and VISualizing splice junctions
Imports:	IRanges, plotrix, rtracklayer, RCurl
License:	Copyright (2019), New York Genome Center. All rights reserved.

Index of help topics:

SCANVIS is a set of tools for SCoring and ANnotating splice junctions using gencode annotation. It also has a VISualization component that allows users to quickly view one or more samples in sashimi style plots, showing splice junctions (SJs) and, optionally, a read coverage profile as well as mutations in one figure. These sashimi style plots are novel in that unannotated splice junctions are highlighted in various colours to delineate various junction types, with line styles indicating whether unannotated junctions are in frame or not.

Author(s)

Phaedra Agius <pagius@nygenome.org>

Maintainer: Phaedra Agius <pagius@nygenome.org>

`SCANVIS.gencode` *assembles gencode annotation into a SCANVIS-compatible format*

Description

This function ftps to the supplied gencode url, downloads gencode data to current directory and assembles the gencode data into an object required for running SCANVIS.R

Usage

```
SCANVIS.gencode(ftp.url)
```

Arguments

`ftp.url`

Value

a gencode object compatible (and required) for use with most SCANVIS functions

Note

Web access required. If variants are available and intended for use with SCANVIS.linkvar, the gencode reference genome must be the same as that used for the variant calls.

Examples

```
gen28=SCANVIS.gencode('ftp://ftp.ebi.ac.uk/pub/databases/gencode/Gencode_human/release_28/')
```

SCANVIS.linkvar *maps variants to SCANVIS scored splice junctions*

Description

This function maps variants to SJs by overlapping the union of gene coordinates that harbor the SJs (optionally, with some gene interval expansion) with variant coordinates

Usage

```
SCANVIS.linkvar(scn, bed, gen, p)
```

Arguments

`scn`	matrix output by SCANVIS.scan
`bed`	matrix with variants in bed format with colnames chr, start, end and with and additional description column (eg. ssSNP for splice site mutations)
`gen`	gencode object as generated by the function SCANVIS.gencode
`p`	expands gene intervals up/downstream by p (default=0, no padding)

Value

Returns the input scn matrix with an additional column showing variants, if any, that occur in/near the listed genes. For instances where multiple variants map to a SJ, the variants are | separated (eg. chr7:145562;A>G|chr7:145592;C>G)

Note

The reference genome used to align RNA-seq reads that generated the initial set of SJs should be the same reference genome used for the variant calls.

See Also

SCANVIS.scan, SCANVIS.gencode, SCANVIS.visual

Examples

```
data(scanvis_examples)
gbm3.scn=SCANVIS.scan(sj=gbm3,gen=gen19,Rcut=5)
### Variant format required (these are toy variants)
head(gbm3.vcf)
gbm3.scnv=SCANVIS.linkvar(gbm3.scn,gbm3.vcf,gen19)
table(gbm3.scnv[,'passedMUT'])
### Expand variant intervals by p
gbm3.scnvp=SCANVIS.linkvar(gbm3.scn,gbm3.vcf,gen19,p=100)
### Observe variant chr6:46820148;Z>AA which was not previously matched to any SJ
table(gbm3.scnvp[,'passedMUT'])
```

`SCANVIS.merge` *merges multiple SCANVIS samples*

Description

With this function, the PSI scores and number of supporting reads across a number of samples are collected into matrices by collecting the union of all SJs. Furthermore, a representative sample is assembled by computing the mean (or median) of PSIs and supporting reads across all samples - this may be used to visualize a cohort in one figure (see SCANVIS.visual).

Usage

```
SCANVIS.merge(scn,method,roi,gen)
```

Arguments

`scn`	list of SCANVIS matrices OR character vector of urls pointing to SCANVIS matrix outputs
`method`	method for computing a PSI/uniq.reads representative, either "mean" or "median" (default="mean")
`roi`	NULL for all SJs OR chromosome name for a query chromosome (eg. chr1) OR 3 bit vector (chr, start, end) indicating region of interest OR a vector with one or more gene names (default=NULL in which case all SJs are merged)
`gen`	gencode object as generated by SCANVIS.gencode which must be supplied if roi is a list of one or more gene names, otherwise NULL (default=NULL)

Value

Returns a list object ready for use in SCANVIS.visual with the following details:

`PSI`	a matrix with PSI scores for each sample (columns) and the union of SJs across all samples (rows)
`NR`	a matrix with number of SJ reads each sample (columns) and the union of SJs across all samples (rows)
`MUTS`	a binary matrix with 1 indicating presence of a mutation (row) in a sample (column), generated only if samples submitted were variant-mapped SJs
`SJ`	a representative sample with mean/median PSI and uniq.reads that can be used in SCANVIS.visual to visualize sample cohort
`roi`	genomic coordinates for region of interest used to derive resulting data

Note

For 50 or more samples, roi cannot be NULL as resulting matrices may be too large. For cohort agglomeration, please consider agglomerating chromosome by chromosome.

See Also

SCANVIS.scan, SCANVIS.linkvar, SCANVIS.visual

Examples

```
data(scanvis_examples)
### merge all SJs across in sample list GBM
GBM.merged=SCANVIS.merge(GBM)
### only merge SJs intersecting with gene PTGDS
GBM.merged=SCANVIS.merge(GBM,'mean','PTGDS',gen19)
```

SCANVIS.scan *SCore, ANnotate and VIsualize splice junctions*

Description

This function annotates and scores splice junctions (SJs) supplied in bed format (coordinates plus read support) and gencode annotation (see SCANVIS.gencode). Each SJ is annotated by gene name and junction type, with unannotated SJs (USJs) falling into one of the following groups: exon.skip, alt3p, alt5p, IsoSwitch, Unknown and NE (Novel Exons) - see below. USJs are also checked and marked for in or out of frame shifts. Each SJ is scored by a Percent Spliced-In (PSI) type score which is dependent on the junction read support of local annotated SJs. This local context is determined by a minimum genomic interval merged over local annotated SJs that intersect with the gene/s hosting the SJ. Novel Exons (NEs) are detected by USJ pairs that coincide in intronic regions and are scored by the mean PSIs of the supporting USJs. NEs are also scored by a read-coverage based PSI (covPSI) if the bam file is supplied.

Usage

```
SCANVIS.scan(sj, gen, Rcut, bam, samtools)
```

Arguments

sj	SJ matrix with colnames chr,start,end,uniq.reads
gen	gencode object as generated by SCANVIS.gencode
Rcut	min read cutoff; only SJs with >=Rcut reads are retained (Default=5)
bam	url to bam file for NE covPSI computation (default=NULL)
samtools	url to samtools function, MUST be specified if bam is supplied (default=NULL)

Value

An extension of the input SJ matrix for relevant SJs, with additional rows for NE junction pairs, as well as the following additional columns:

JuncType	describes junction type as annot for annotated SJs and one of the following for unannotated SJs: exon.skip, alt3p, alt5p, IsoSwitch, Unknown and NE (Novel Exons) where exon.skip refers to SJs that skip an exon present in all isoforms, alt3p refers to an alternative 3 prime acceptor site, alt5p refers to an alternative 5 prime donor sites, IsoSwitch refers to SJs aligning to mutually exclusive isoforms such that a novel unannotated isoform is incurred, Unknown SJs have coordinates that do not align to any exons and NE (Novel Exons) refers to SJ pairs with the start of one SJ and the end coordinate of the other SJ coinciding in an intronic region
gene_name	genes that intersect with the SJ (multiple genes are comma separated
PSI	Percent Spliced-In score defined as x/(x+y) where x is the number of reads of the query junction and y is the median of the number of reads supporting annotated SJs in genomic_interval
genomic_interval	interval used for the PSI computation
FrameStatus	frame shifts induced by unannotated SJs, where INframe indicates no frame-shift in any gene isoforms, OUTframe indicates frame-shifting in ALL gene isoforms and all other entries indicating frame shifts for specified isoforms. FrameStatus is marked NA for annotated SJs)
covPSI	generated for NEs only if bam file is supplied

See Also

SCANVIS.gencode, SCANVIS.linkvar, SCANVIS.visual

Examples

```
data(scanvis_examples)
head(gbm3) #required SJ format
gbm3.scn=SCANVIS.scan(sj=gbm3,gen=gen19,Rcut=5)
head(gbm3.scn)
### to compute coverage-based PSI scores for NEs, run as follows:
#gbm3.scn=SCANVIS.scan(gbm3,gen19,5,bam=<BAM>,samtools=<SAMTOOLS>)
```

SCANVIS.visual *a sashimi-style visualization tool*

Description

This function quickly generates sashimi-style plots for SCANVIS outpus showing SJ details for a query gene or a specific genomic region. Annotated SJs are depicted with grey arcs, while different colors segregate unannotated SJ subtypes. Arc height and thickness correspond to the junction read support and PSI score respectively. If the supplied junction file is output from SCANVIS.linkvar output, then variants are also plotted. If the bam file is supplied, a normalized read coverage profile is shown as an inverted read profile. Multiple samples may be supplied, in which case the SCANVIS.merge function is used to merge the samples. The resulting output is a sashimi plot of the union of SJs over the submitted sample cohort, with SJs depicted by mean PSI and read support over the samples. This is useful for comparing disease cohorts.

Usage

```
SCANVIS.visual(roi,gen,scn,SJ.special,TITLE,bam,samtools,full.annot)
```

Arguments

roi	gene name OR region of interest (chr,start,end as 3-bit vector)
gen	gencode object as generated by the function SCANVIS.gencode.R
scn	matrix OR list of url/s to output from SCANVIS.scan/linkvar (which will be submitted to SCANVIS.merge) OR output from SCANVIS.merge for a set of samples already merged
SJ.special	3 col matrix indicating chr,start,end of any SJs of interest to be highlighted in cyan (default=NULL)
TITLE	figure name/title (default=NULL)
bam	url to one bam file corresponding to the input scn (not applicable for multiple/merged samples, default=NULL)
samtools	url to samtools which MUST be specified if bam is supplied (default=NULL)
full.annot	TRUE for each isoform listed separately, FALSE for concise format (default=FALSE)

Value

Returns a sashimi-style plot depicting the relevant SJs, as well as an object with the coordinates of the genomic region, the SJs and any variants in the figure

See Also

SCANVIS.scan, SCANVIS.linkvar

Examples

```
data(scanvis_examples)

### exon skip events in PPA2 in two LUSC samples
par(mfrow=c(2,1),mar=c(1,1,1,1))
vis.lusc1=SCANVIS.visual('PPA2',gen19,LUSC[[1]],TITLE=names(LUSC)[1],full.annot=TRUE)
vis.lusc2=SCANVIS.visual('PPA2',gen19,LUSC[[2]],TITLE=names(LUSC)[2],full.annot=TRUE)
### if bam file were available for LUSC1 ...
#vis.lusc1=SCANVIS.visual('PPA2',gen19,LUSC[[1]],TITLE=names(LUSC)[1],..\
#full.annot=TRUE,bam=<BAM4LUSC1>,samtools=<SAMTOOLS>)

### sashimi plots with variants
gbm3.scn=SCANVIS.scan(sj=gbm3,gen=gen19,Rcut=5)
gbm3.scnv=SCANVIS.linkvar(gbm3.scn,gbm3.vcf,gen19)
vis.gbm3=SCANVIS.visual('PTGDS',gen19,gbm3.scnv,TITLE='gbm3')
roi=vis.gbm3$roi
d=diff(as.numeric(roi[2:3]))
roi2=c(roi[1],round(as.numeric(roi[2])+(d*0.1)),round(as.numeric(roi[3])-(d*0.5)))
### Supply exact coordinates instead of gene names ... Zooming in for gbm3
vis.gbm3.zoom=SCANVIS.visual(roi2,gen19,gbm3.scnv)

### plot multiple genes ... PTGDS and neighbors
vis.gbm3.multiple_genes=SCANVIS.visual(c('FBXW5','PTGDS','C9orf142'),gen19,gbm3.scnv,TITLE='gbm3')
```

```
par(mfrow=c(2,1),mar=c(1,1,1,1))
### see PTGDS in merge of 3 GBMs
GBM.PTGDS=SCANVIS.visual('PTGDS',gen19,GBM,TITLE='GBM, merged',full.annot=TRUE)
#### see PTGDS in merge of 3 LUADs ... no exon skips
LUAD.PTGDS=SCANVIS.visual('PTGDS',gen19,LUAD,TITLE='LUAD, merged',full.annot=TRUE)

### NEs in GPR116 in LUAD, but not in GBM
par(mfrow=c(2,1),mar=c(1,1,1,1))
GBM.GPR116=SCANVIS.visual('GPR116',gen19,GBM,TITLE='GBM, merged',full.annot=TRUE)
LUAD.GPR116=SCANVIS.visual('GPR116',gen19,LUAD,TITLE='LUAD, merged',full.annot=TRUE)
```

Index