Package 'chimera'

October 15, 2018

Type Package

Title A package for secondary analysis of fusion products

Version 1.22.0

Date 31 March 2015

Author Raffaele A Calogero, Matteo Carrara, Marco Beccuti, Francesca Cordero

Maintainer Raffaele A Calogero <raffaele.calogero@unito.it>

Depends Biobase, GenomicRanges (>= 1.13.3), Rsamtools (>= 1.13.1), GenomicAlignments, methods, AnnotationDbi, BSgenome.Hsapiens.UCSC.hg19, TxDb.Hsapiens.UCSC.hg19.knownGene, Homo.sapiens

Suggests BiocParallel, geneplotter

Enhances Rsubread, BSgenome.Mmusculus.UCSC.mm9, TxDb.Mmusculus.UCSC.mm9.knownGene, BSgenome.Mmusculus.UCSC.mm10, TxDb.Mmusculus.UCSC.mm10.knownGene, Mus.musculus, BSgenome.Hsapiens.NCBI.GRCh38, TxDb.Hsapiens.UCSC.hg38.knownGene

Description

This package facilitates the characterisation of fusion products events. It allows to import fusion data results from the following fusion finders: chimeraScan, bellerophontes, deFuse, FusionFinder, FusionHunter, mapSplice, tophat-fusion, FusionMap, STAR, Rsubread, fusionCatcher.

biocViews Infrastructure

SystemRequirements STAR, TopHat, bowtie and samtools are required for some functionalities

License Artistic-2.0

git_url https://git.bioconductor.org/packages/chimera

git_branch RELEASE_3_7

git_last_commit a334c3d

git_last_commit_date 2018-04-30

Date/Publication 2018-10-15

R topics documented:

`chimera-package` *A package for secondary analysis of fusion products*

Description

The package imports fusion results from tophat-fusion, tophat-fusion-post, mapSplice, deFuse, fusionmap, bellerophontes, fusionfinder, fusionhunter, STAR, Rsubread, fusionCatcher. The package was design to facilitate the characterisation of fusion products events. Data upload: outputs for the above indicated fusion detection tools can be imported using `importFusionData` in a list of `fSet` objects. `fSet-class` offers various methods to extract information from the fSet objects. The fusion names can be extracted with `fusionName` function. The number of reads supporting a fusion event cam be extracted with the `supportingReads` function.

Filtering: The imported fusion list can be filtered using `filterList`

Annotation: Oncofuse can be installed in chimera with the function `oncofuseInstallation` Various information on the fusions location, on structural and functional domains affected by the fusion event as well as a prediction of the putative functional effect of the fusion on the cell can be obtained by using `oncofuseRun`.

`chimeraSeqs` generates the nucleotide sequence of a fusion transcript described in an fSet object. `chimeraSeqSet` does the same but on a list of `fSet` objects.

`fusionPeptides` allows to investigate if the fusion events generate also a fusion at protein level.

`subreadRun` allows to remap reads on the fused transcripts reconstructed with `chimeraSeqs`

Validation GapFiller is a seed-and-extend local assembler capable to produce (in-silico) longer and highly accurate sequences from a collection of Next Generation Sequencing reads. It can be installed in chimera with the function `gapfillerInstallation`. The function `gapfillerRun` allows to check if reads mapped by `subreadRun` over a fused transcript generated with `chimeraSeqs` are able to reconstruct by de novo assembly the fusion break-point.

Export The function `prettyPrint` converts the information stored in a list of `fSet` objects in a dataframe structure that is saved as tab delimited file.

Details

Package:	chimera
Type:	Package
Version:	1.0
Date:	2014-31-07
License:	Artistic-2.0

Author(s)

Raffaele A Calogero Maintainer: Raffaele A Calogero <raffaele.calogero@unito.it>

References

Beccuti M, et al. The structure of state-of-art gene fusion-finder algorithms. OA Bioinformatics 2013;1(1):2. Carrara, M., et al. State of art fusion-finder algorithms are suitable to detect transcription-induced chimeras in normal tissues? BMC bioinformatics 2013;14 Suppl 7:S2. Carrara, M., et al. State-of-the-Art Fusion-Finder Algorithms Sensitivity and Specificity. BioMed research international 2013;2013:340620. Shugay, M., et al. Oncofuse: a computational framework for the prediction of the oncogenic potential of gene fusions. Bioinformatics 2013;29(20):2539-2546. Nadalin, F., et al. GapFiller: a de novo assembly approach to fill the gap within paired reads. BMC bioinformatics 2012;13 Suppl 14:S8.

`bam2fastq` *A function to extract pair end reads from the bam file generated with subreadRun function*

Description

A function to extract pair end reads from the bam file generated with subread function. The output files are ready to be used for fusion validation with gapfiller

Usage

```
bam2fastq(bam, filename="ready4gapfiller",ref,parallel=FALSE)
```

Arguments

bam	name of the bam file to be used for PE reads extraction
filename	base name for the PE fastq output data
ref	name of the fusion sequence that was used as reference
parallel	option that allow the use of BioParallel package

Value

PE fastq files

Author(s)

Raffaele A Calogero

Examples

```
#if(require(Rsubread)){
#  subreadRun(ebwt=paste(find.package(package="chimera"),"/examples/SULF2_ARFGEF2.fa",sep=""),
#    input1=paste(find.package(package="chimera"),"/examples/mcf7_sample_1.fq",sep=""),
#    input2=paste(find.package(package="chimera"),"/examples/mcf7_sample_2.fq",sep=""),
#    outfile.prefix="accepted_hits", alignment="se", cores=1)
#  ref.name <- names(readDNAStringSet(paste(find.package(package="chimera"),"/examples/SULF2_ARFGEF2.fa",se
# bam2fastq(bam="accepted_hits.bam", filename="ready4gapfiller", ref=ref.name, parallel=F)
#}
```

---

breakpointOverlaps *A function to extract the reads overlapping to fusion breakpoint.*

Description

A function to extract the reads overlapping to fusion breakpoint.

Usage

```
breakpointOverlaps(fset, plot=FALSE, ylim=NULL)
```

Arguments

fset	An fSet object. The slots fusionRNA and fusionGA needs to be loaded
plot	If FALSE plot is not printed
ylim	If NULL it uses the full fusion transcript coverage to define the ylim of the plot. If setted it can be used to zoom in the plot to better see the structure of the coverage at the brek point

Value

An object of GAlignment class. A plot of the fusion trascript coverage in blue and of the reads spanning over the break point in yellow.

Author(s)

Raffaele A Calogero

Examples

```
load(paste(find.package(package="chimera"), "/examples/fset_ARFGEF2-SULF2.rda", sep=""))
my.seq <- chimeraSeqs(my.fset)
my.fset <- addRNA(my.fset, my.seq)
tmp <- breakpointOverlaps(my.fset)
```

---

chimeraSeqs *A function to generate the nucleotide sequences of a fusion event*

Description

A function generating the nucleotide sequences of a chimera.

Usage

```
chimeraSeqs(fset, extend=1000, type="transcripts")
```

Arguments

fset	A fSet object.
extend	number of nucleotides used to extend a genomic region that is not an annotated gene. Default is 1000 nts
type	Chimera can be build at transcript level, i.e. transcript level will generate multiple fusion transcripts depending of the number of transcripts associated to each of the two genes involved in the fusion.

Value

A DNAStringSet encompassing the fusions generated using all the isoforms for each gene involved in the fusion.The name of each element of the DNAStringSet has the following format: gene1-lengthOfGeneFragment:gene2-lengthOfGeneFragment. In case the fusion junction is located in an intronic sequence, a warning is provided. The presence of a partial intron in the fusion is an indication that the fusion does not generate an active chimeric peptide.

Author(s)

Raffaele A Calogero

See Also

fusionName, chimeraSeqSet

Examples

```
tmp <- importFusionData("fusionmap", paste(find.package(package="chimera"),"/examples/mcf7.FMFusionReport",
fusion.names <- fusionName(tmp)
fusion.names
myset <- tmp[[13]]
tmp.seq <- chimeraSeqs(myset, type="transcripts")
#writeXStringSet(tmp.seq, paste(sub(":","_",fusion.names[[13]]),".fa",sep=""), format="fasta")
```

`chimeraSeqSet` *A function to generates DNAStringSet encompassing fusion sequences*

Description

A function generating the nucleotide sequences of chimeras described in a list of fSet, i.e. the list generated using importFusionData function.

Usage

```
chimeraSeqSet(list, parallel=FALSE)
```

Arguments

`list`	A list of fSet objects.
`parallel`	If TRUE uses the BioParallel package

Value

A DNAStringSet encompassing the fusions described in a list of fSet objects. This object represents the ideal reference to remap reads over detected fusions. Remapping is required to validate fusions using GapFiller de novo reconstruction.

Author(s)

Raffaele A Calogero

See Also

`fusionName`, `importFusionData`, `gapfillerInstallation`, `gapfillerRun`

Examples

```
tmp <- importFusionData("fusionmap", paste(find.package(package="chimera"),"/examples/mcf7.FMFusionReport",
fusion.names <- fusionName(tmp)
fusion.names
myset <- tmp[1:3]
tmp.seq <- chimeraSeqSet(myset, parallel=FALSE)
#   sapply(tmp.seq, function(x){writeXStringSet(x, "detected.fusions.fa", format="fasta", append=TRUE)})
```

`defuseTPTN` *A function that genererate a list of fSet objects encopassing 60 experimentally validated fusions and 61 false fusions*

Description

The function uses the file defuse_TP_FP.txt located in the examples folder of chimera to generate a list of fSet object encompassing 60 TP fusions and 61 TN fusions extracted from supplementary table 8 of the deFuse paper McPherson et al. (2011) PLoS Comput Biol 7(5): e1001138. The fset object can be used to evaluate the efficacy of filtering approaches available in chimera package.

Usage

```
defuseTPTN()
```

Value

An list of fSet objects: Experimentally validated fusions 1-60, fish validated fusions 1-14, false fusions 61-121.

Author(s)

Raffaele A Calogero

Examples

```
tptn <-defuseTPTN()
```

`filterList` *A function to filter a list of fSet objects*

Description

A function filtering a list of fSet objects on the basis of various parameters.

Usage

```
filterList(x,type=c("spanning.reads","fusion.names", "intronic", "annotated.genes", "read.through
```

Arguments

`x`	a list of fSet object
`type`	filtering can be performed on the basis of spanning.read: a minimal number of spanning reads, fusion.names: a vector list of user defined fusion names, intronic: only fusion encompassing exon data are retained annotated.genes: only fusion encompassing two annotated genes are retained read.through: only fusion with different gene names are retained oncofuse: using the output of oncofuseRun various filtering option can be applied using oncofuse.type and oncofuse.threshold

oncofuse.output	The output generated with oncofuseRun
query	query is a number. In case spanning.reads is selected or a vector of fusion names if the case fusion.names is selected. In case type is intronic query is NULL. In the latter case fusion having one of the elements located in an intronic region are discarded. In the case of oncofuse.type equal to passenger.prob or expression.gain query is the threshold number to be used for the filtering
oncofuse.type	This refers to the filtering based on the output of oncofuse that can be generated using the oncofuseRun function. g5CDS selects only fusions having the breakpoint in 5' end gene CDS, g3CDS selects only fusions having the breakpoint in 3' end gene CDS, g5g3CDS selects only fusions having the breakpoint in both gene CDSs, g5exon selects only fusions having the breakpoint in an exon of 5' end gene, g3exon selects only fusions having the breakpoint in an exon of 3' end gene, g5g3exon selects only fusions having the breakpoint in an exon of both genes. In the case of oncofuse.type equal to driver.prob the filter will use the probability of the fusion to be a tumor driver. In the case of oncofuse.type equal to expression.gain the filter will use the score that suggests the presence of a gain in expression
parallel	option to run a parallel version of the function, default FALSE

Author(s)

Raffaele A Calogero

Examples

```
tmp <- importFusionData("fusionmap", paste(find.package(package="chimera"),"/examples/mcf7.FMFusionReport",
fusion.names <- fusionName(tmp)
tmp1 <- filterList(tmp, type="fusion.names", fusion.names[c(1,3,7)], parallel=FALSE)
tmp2 <- filterList(tmp, type="spanning.reads", query=2, parallel=FALSE)
#tmp3 <- filterList(tmp, type="intronic")
#tmp4 <- filterList(tmp, type="annotated.genes", parallel=FALSE)
#tmp5 <- filterList(tmp, type="read.through", parallel=FALSE)
#csdf.of <- oncofuseRun(csdf.e, tissue="EPI")
#tmp6 <- filterList(csdf.e[1:100],oncofuse.output=csdf.of, type="oncofuse", oncofuse.type="g5g3CDS", paralle
#tmp7 <- filterList(csdf.e[1:100],oncofuse.output=csdf.of, type="oncofuse", oncofuse.type="passenger.prob",
```

---

filterSamReads *A function to filter SAM or BAM files*

Description

A function to filter SAM or BAM files using picard-tools

Usage

```
filterSamReads(input, output, filter=c("includeAligned","excludeAligned"))
```

Arguments

input	SAM/BAM file to be validated
output	file name in which to save the filtered reselts
filter	type of filter

Value

A filtered SAM/BAM.

Author(s)

Raffaele A Calogero

See Also

picardInstallation

Examples

```
# filterSamReads(input="kd2_accepted_hits2.sam", output="kd2_accepted_hits2_mapped.sam", filter="includeAli
```

---

fSet *Class fSet, a class represent fusion data, and methods for processing it*

Description

This is class representation for a fusion event.

Slots

fusionInfo A list: fusionTool: the tool that has generated the fusions UniqueCuttingPositionCount: the number of unique cutting positions detected for the fusion. SeedCount: the number of reads overlapping the break-point, i.e. spanning reads (FusionMap, FusionHunter, mapSplice, Tophat-fusion, ChimeraScan, STAR, Rsubread, FusionCatcher). Both spanning and encompassing reads (Bellerophontes, FusionFinder). Encompassing reads, i.e. one read of a pair on gene 1, and the other on gene2 (deFuse). RescuedCount: the number of reads overlapping the break-point rescued after identification of the break point (FusionMap). Encompassing reads (Tophat-fusionm Tophat-fusion-post, FusionCatcher, STAR). Both spanning and encompassing reads (ChimeraScan, Rsubread). SplicePattern: the splice pattern for a fusion junction FusionGene: the name of the fusion gene in the format gene1 -> gene2. frameShift: frameshift at break-point

fusionLoc A GRangesList encompassing fusion locations for gene 1 and 2

fusionRNA A DNAStringSet encompassing the fusion transcript that can be generated by chimeraSeqs function.

fusionGA A GAlignments object encompassing positions for all reads mapping on the DNAStringSet located in fusionRNA slot

Methods

Standard generic methods:

fusionData(fSet) An accessor function used to retrieve information for a fusion

fusionGRL(fSet) An accessor function used to retrieve GRangesList encompassing fusion locations for gene 1 and 2

fusionRNA(fSet) An accessor function used to extract the DNAStringSet

addRNA(fSet, rna) An accessor function used to add the DNAStringSet to the fset object

fusionGA(fSet) An accessor function used to extract the GAlignments object

addGA(fSet, bam) An accessor function used to add the GAlignments object to the fSet object

Constructor

newfSet `newfSet(fusionInfo = list(), fusionLoc = GRangesList(), fusi`
Creates a fSet object.

`fusionInfo` A list of the fusin characteristics see above slot fusionInfo

`fusionLoc` A GRangesList encompassing the fusion break points

`fusionRNA` A DNAStringSet encompassing the fusion transcript

`fusionGA` A GAlignments encompassing the location of reads mapping on the fusion transcript

Author(s)

Raffaele A Calogero

See Also

`chimeraSeqs`, `oncofuseRun`, `fusionPeptides`, `prettyPrint`

Examples

```
#creating a fusion report from output of fusionMap
tmp <- importFusionData("fusionmap", paste(find.package(package="chimera"),"/examples/mcf7.FMFusionReport",
#extracting the fusions names
fusion.names <- fusionName(tmp)
fusion.names
#extracting the fSet object ofr one of the fusions
myset <- tmp[[13]]
#constructing the fused sequence(s)
trs <- chimeraSeqs(myset, type="transcripts")
#adding the sequences to the fSet object
myset <- addRNA(myset , trs)
#extracting sequences from an fSet object
tmp.seq <- fusionRNA(myset)
#adding reads mapped on the fusion generated using tophatRun function
myset <- addGA(myset, paste(path.package(package="chimera"),"/examples/mcf7_trs_accepted_hits.bam",sep=""))
#extracting the GAlignments from an fSet object
ga <- fusionGA(myset)
```

---

fusionName *A function to extract fusion names for a list of fSet object*

Description

A function allowing extract fusion names from a list of fSet objects.

Usage

```
fusionName(list, parallel=FALSE)
```

Arguments

list	a list of fSet object
parallel	option to run a parallel version of the function, default FALSE

Author(s)

Raffaele A Calogero

Examples

```
tmp <- importFusionData("fusionmap", paste(find.package(package="chimera"),"/examples/mcf7.FMFusionReport",
fusion.names <- fusionName(tmp)
fusion.names
```

---

fusionPeptides *A function to investigate the peptides involved in the fusion event.*

Description

A function extracting the donor and the acceptor peptides involved in the fusion.

Usage

```
fusionPeptides(chimeraSeq.output, annotation="hsUCSC")
```

Arguments

chimeraSeq.output	DNAStringSet encompassing the fusion event of interest, generated by chimeraSeq function
annotation	The annotation used to retrieve the UCSC names of the transcripts involved in the fusion

Value

An list encompassing:

AAStringSet	encompassing: fusion sequence, peptide from p1 and peptide from p2. In case the peptides are not in frame the AAStringSet will not contain the fusion sequence
DNAStringSet	encompassing the fusion transcript

Author(s)

Raffaele A Calogero

Examples

```
tmp <- importFusionData("fusionmap", paste(find.package(package="chimera"),"/examples/mcf7.FMFusionReport",
fusion.names <- fusionName(tmp)
fusion.names
myset <- tmp[1:3]
tmp.seq <- chimeraSeqSet(myset, parallel=FALSE)
tmpx <- lapply(tmp.seq,fusionPeptides)
```

gapfillerInstallation *A function to download a compiled version of GapFiller*

Description

A function allowing the download and installation of a compiled version of GapFiller in chimera package folder. The source code of GapFiller can be retrieved at http://sourceforge.net/projects/gapfiller/files/?source=nav . The paper describing the tool is Nadalin F, Vezzi F, Policriti A. GapFiller: a de novo assembly approach to fill the gap within paired reads. BMC Bioinformatics. 2012;13 Suppl 14:S8.

Usage

```
gapfillerInstallation(os=c("mac64","unix64"))
```

Arguments

os	The supported operating systems

Author(s)

Raffaele A Calogero

See Also

gapfillerRun

Examples

```
#gapfillerInstallation(os="mac64")
```

gapfillerRun *A function to confirm fusion break point by de novo assembly*

Description

A function that uses GapFiller to confirm, by de novo assembly, the presence of the fusion break point. The function needs as input the fusion transcript generated by chimeraSeqs function and two fastq files corresponding to the reads mapping over the fusion transcript.

Usage

```
gapfillerRun(fusion.fa, seed1,  seed2, gapfiller=NULL, seed.ins=200, seed.var=50,
block.length=5, overlap=20, max.length=5000, slack=30,
k=6, global.mismatch=5)
```

Arguments

fusion.fa	fasta file with the fusin transcript
seed1	The R1 fastq of a pair-end
seed2	The R2 fastq of a pair-end
gapfiller	path to GapFiller executable program
seed.ins	seed reads insert size
seed.var	seed reads insert variation
block.length	length of perfect match
overlap	minimum suffix-prefix overlap
max.length	print only contigs <= max-length long
slack	number of overlaps: suffix-prefix overlap range is given by overlap, overlap + slack
k	length of the word used to hash
global.mismatch	maximum number of mismatches between mate and contig

Value

The program will write in a temporary directory contigs.fasta and contig.stats, which are used to evaluate if the de novo assembly allows the identification of the fusion break point. The function returns a list of three objects. The list is returned only in case that some of de novo assemblies cover the breakpoint junction. The list is made of:

contigs	which is a PairwiseAlignments object
junction.contigs	which is a DNAStringSet encompassing the sequences present in the contigs object
fusion	which is a DNAStringSet object encompassing the fusion transcript

Author(s)

Raffaele A Calogero

See Also

chimeraSeqs, gapfillerInstallation

Examples

```
# tmp <- gapfillerRun(fusion.fa=paste(path.package("chimera", quiet = FALSE),"/examples/uc002xvp.1-243_uc002
```

gapfillerWrap *A function to prepare files and to run gapfiller*

Description

A function that uses GapFiller to confirm, by de novo assembly, the presence of the fusion break point. The function needs as input a list of fusion transcript generated by chimeraSeqSet function and the bam file containing the reads remapped over the fusion transcripts made using subreadRun.

Usage

```
gapfillerWrap(chimeraSeqSet.out, bam, parallel=c(FALSE,TRUE))
```

Arguments

chimeraSeqSet.out	a list of DNAStringSet output from chimeraSeqSet
bam	bam file containing the reads remapped over the fusion transcripts using Rsubread
parallel	if FALSE FALSE no parallelization, if TRUE TRUE full paralleization, if FALSE TRUE only parallelization for internal funtions

Value

The program will write in a temporary directory contigs.fasta and contig.stats, which are used to evaluate if the de novo assembly allows the identification of the fusion break point. The function returns for each fusion a list of three objects. The list is returned only in case that some of de novo assemblies cover the breakpoint junction. The list is made of:

contigs	which is a PairwiseAlignments object
junction.contigs	which is a DNAStringSet encompassing the sequences present in the contigs object
fusion	which is a DNAStringSet object encompassing the fusion transcript

Author(s)

Raffaele A Calogero

See Also

chimeraSeqs, gapfillerInstallation, gapfillerRun

Examples

```
#tmp <- importFusionData("star", "Chimeric.out.junction", org="hg19", min.support=100)
#myset <- tmp[1:4]
#tmp.seq <- chimeraSeqsSet(myset, type="transcripts")
#tmp <- gapfillerWrap(chimeraSeqSet.out=trsx, bam="accepted_hits_mapped.bam", parallel=c(FALSE,TRUE))
```

`importFusionData` *A function to import fusion data detected by different fusion finders*

Description

A function to import in a list fusions data detected by bellerophontes, defuse, fusionfinder, fusionhunter, mapsplice, tophat-fusion, fusionmap, chimerascan, STAR, Rsubread, fusionCatcher. In the case of chimerascan and STAR output it is possible to load the data applying a filter on the minimal number of reads supporting a specific fusion. Both chimerascan and STAR accept data generated using human hg19, hg38, mouse mm9 and mm10 reference. IMPORTANT: please note that the it is important that the genome reference version used for the alignment is the same of that used for by chimera for annotation. Expecially between hg38 and hg19 there are shifts in gene location.

Usage

```
importFusionData(format, filename, ...)
```

Arguments

`format`	Format allows to select the data structure to be imported. One of the following keyword need to be associated to format parameter: bellerophontes, defuse, fusionfinder, fusionhunter, mapsplice, tophat-fusion, fusionmap, chimerascan, star, rsubread, fusioncatcher.
`filename`	The file generated by one of the fusion finders defined in format
`...`	Additional parameters: In case of rsubread min.distance, which indicates the minimal distance to consider a fusion within the same chromosome, is set to 700000 as default. In case of rsubread, chimerascan and STAR min.support, which indicates the minimal number of reads supporting a specific fusion, is set to 10 as default. Please remember that using low values as min.support, e.g. 1, will significantly increase the time for data import. In this case is strongly suggested to run the function as batch and, when available, using the parallel option, see below. In case of chimerascan, STAR, Rsubread, fusionmap, fusioncatcher, tophat-fusion, tophat-fusion-post, mapsplice, bellerophontes org, which indicates the organism to be used for annotation, has the following options: hg19, hg38, mm9, mm10. In case of STAR and Rsubread the upload can be parallelized using parallel=TRUE

Author(s)

Raffaele A Calogero

Examples

```
tmp <- importFusionData("fusionmap", paste(find.package(package="chimera"),"/examples/mcf7.FMFusionRepor
#min.support allow to retrieve only the subset of fusions supported by a user defined minimal number of jur
#tmp <- importFusionData("chimerascan", "edgren_cs10.bedpe", min.support=10, org="hg19")
#download Edgren Chimeric.out.junction. This file encompass the results obtaines combined all cell lines u
#download.file("http://sourceforge.net/projects/ochguiextras/files/chimera/Chimeric.out.junction.zip/do
 #unzip("Chimeric.out.junction.zip")
#tmp <- importFusionData("star", "Chimeric.out.junction", org="hg19", min.support=100)
```

is.fSet *A function to evaulate if an object belongs to fSet class or not*

Description

A function to evaulate if an object belongs to fSet class or not.

Usage

```
is.fSet(x)
```

Arguments

x	an object

Value

If the object belons to fSet class it returns TRUE, else it returned FALSE

Author(s)

Raffaele A Calogero

Examples

```
tmp <- importFusionData("fusionmap", paste(find.package(package="chimera"),"/examples/mcf7.FMFusionReport",
is.fSet(tmp[[1]])
```

MHmakeRandomString *A function generating a random string*

Description

A function generating a random string.

Usage

```
MHmakeRandomString()
```

Value

a string

Author(s)

Raffaele A Calogero

Examples

```
tmp.file <- paste(MHmakeRandomString(),".fa", sep="")
```

newfSet *A constructor for fSet class objects*

Description

A function to create a new fSet object

Usage

```
newfSet(fusionInfo = list(fusionTool = NA,
                                      UniqueCuttingPositionCount = 0,
                                      SeedCount = 0,
                                      RescuedCount = 0,
                                      SplicePattern = NA,
                                      FusionGene = NA,
                                      frameShift = NA),
            fusionLoc = GRangesList(),
                fusionRNA = DNAStringSet(),
                fusionGA = GAlignments())
```

Arguments

fusionInfo	A list of the fusion characteristics see fSet class slot fusionInfo
fusionLoc	A GRangesList encompassing the fusion break points
fusionRNA	A DNAStringSet encompassing the fusion transcript
fusionGA	A GAlignments encompassing the location of reads mapping on the fusion transcript

Value

An object of fSet class

Author(s)

Raffaele A Calogero

Examples

```
tmp <- newfSet()
tmp
```

`oncofuseInstallation` *A function to download oncofuse*

Description

A function allowing the download of oncofuse in the chimera folder. Oncofuse requires java

Usage

```
oncofuseInstallation()
```

Author(s)

Raffaele A Calogero

See Also

oncofuseRun

Examples

```
#oncofuseInstallation()
```

`oncofuseRun` *A function to annotate fusions with Oncofuse. Oncofuse is a naive bayesian classifier. Its goal is to identify those fusion sequences with higher probability of being driver than passenger events*

Description

A list of fSet objects can be annotated using the Oncofuse java application.

Usage

```
oncofuseRun(listfSet, tissue=c("EPI","HEM","MES","AVG"), org=c("hg19","hg38"), threads=1, plot=FA
```

Arguments

`listfSet`	A list of fSets
`tissue`	Type of tissue in which the fusion was detected. EPI epithelial, HEM ematological, MES mesenchimal, AVG average expression
`org`	Supported genome assembly version. IMPORTANT: the genome version used for the fusion detection and for the oncofuse analysis need to be same
`threads`	number of threads that Oncofuse will use
`plot`	plotting the expression gain score versus the passenger mutation probability. For more info please see Shugay et al. Bioinformatics 2013:29,2539
`type`	listfSet a list of fSet objects or coordDf a dataframe of coordinates of fusions in the format required by Oncofuse

Value

The output is a dataframe containing the output of Oncofuse. For more info on Oncofuse please see Shugay et al. Bioinformatics 2013, 29, 2539.

Author(s)

Raffaele A Calogero

Examples

```
#tmp <- importFusionData("fusionmap", paste(find.package(package="chimera"),"/examples/mcf7.FMFusionReport"
#installOncofuse()
#of.out <- oncofuseRun(tmp, tissue="EPI")
```

---

picardInstallation *A function to download picard-tools*

Description

A function allowing the download of picard-tools in the chimera folder. Picard tools require java

Usage

```
picardInstallation()
```

Author(s)

Raffaele A Calogero

Examples

```
#picardInstallation()
```

---

plotCoverage *A function to plot the coverage of a fusion gene*

Description

A function to plot the coverage of a fusion gene.

Usage

```
plotCoverage(fset, plot.type=c("exons","junctions"), junction.spanning=20, fusion.only=FALSE, xla
```

Arguments

fset	A fSet object
plot.type	exons plot exons coverage as junctions plot coverage of junction between exons
junction.spanning	number of nucleotides located upstream and downstream the junction/fusion location
fusion.only	if TRUE only fusion coverage is plotted
xlab	x-axis label
ylab	y-axis label
main	Plot title
col.box1	color of the box describing the first gene
col.box2	color of the box describing the second gene
ybox.lim	y range defining the height of the box representing the exons

Author(s)

Raffaele A Calogero

See Also

fusionName, chimeraSeqs

Examples

```
tmp <- importFusionData("fusionmap", paste(find.package(package="chimera"),"/examples/mcf7.FMFusionReport",
fusion.names <- fusionName(tmp)
fusion.names
myset <- tmp[[13]]
trs <- chimeraSeqs(myset, type="transcripts")
myset <- addRNA(myset , trs)
tmp.seq <- fusionRNA(myset)
myset <- addGA(myset, paste(path.package(package="chimera"),"/examples/mcf7_trs_accepted_hits.bam",sep=""))
ga <- fusionGA(myset)
plotCoverage(myset, plot.type="exons", col.box1="red", col.box2="green", ybox.lim=c(-4,-1))
plotCoverage(myset, plot.type="junctions", col.box1="red", col.box2="yellow", ybox.lim=c(-4,-1))
plotCoverage(myset, fusion.only=TRUE, col.box1="red", col.box2="yellow", ybox.lim=c(-4,-1))
```

---

prettyPrint *A function to represent a list of fSet as a dataframe*

Description

A function reorganizing a list of fSet in a dataframe structure. The dataframe is then saved in a tab delimited file

Usage

```
prettyPrint(list, filename, fusion.reads=c("all","spanning"))
```

Arguments

list	a list of fSet object
filename	the name of the file in which the dataframe is printed
fusion.reads	it allows to extract spanning reads associated to the SeedCount slot or all the detected reads associate to the RescuedCount

Author(s)

Raffaele A Calogero

Examples

```
#tmp <- importFusionData("fusionmap", paste(find.package(package="chimera"),"/examples/mcf7.FMFusionReport"
#fusion.names <- fusionName(tmp)
#tmp1 <- filterList(tmp, type="fusion.names", fusion.names[c(1,3,7)], parallel=FALSE)
#prettyPrint(tmp1, "tmp1.df.txt", fusion.reads="spanning")
```

---

removingErrorLine *A function to remove a line stopping SAM to BAM conversion*

Description

A function to remove a line stopping SAM to BAM conversion

Usage

```
removingErrorLine(line.number, filenameIn, filenameOut)
```

Arguments

line.number	line number to be removed
filenameIn	input file name
filenameOut	output file name

Value

SAM file without the error line

Author(s)

Raffaele A Calogero

Examples

```
#removingErrorLine(14680066,"kd2_accepted_hits.sam","kd2_accepted_hits1.sam")
```

starInstallation *A function to download STAR*

Description

A function allowing the download and installation of STAR (Dobin et al. Bioinformatics 2012) in chimera package folder. The function also creates soft links in the user bin folder to allow the call of the above mentioned program.

Usage

```
starInstallation(binDir, os=c("unix","mac"))
```

Arguments

binDir	The user bin folder
os	The supported operating systems

Author(s)

Raffaele A Calogero

Examples

```
#starInstallation(binDir="/somewhere/inyourpc/bin", os="mac")
```

starReads *A function to extract reads info from STAR fusion output*

Description

A function producing a GRangeList for the reads information, involved in a fusion event.

Usage

```
starReads(fusion.report, parallel=FALSE)
```

Arguments

fusion.report	STAR fusion output file
parallel	option to run a parallel version of the function, default FALSE

Author(s)

Raffaele A Calogero

Examples

```
#tmp <- starReads("Chimeric.out.junction", parallel=FALSE)
```

starRun *A function to generate a bam file for fusions coverage evaluation*

Description

A function mapping reads to a chimera sequence set. The bam produced by this remapping on a putative fusion will be used to plot the coverage data for all the fused constructs. The function assumes that STAR is installed and located in the path.

Usage

```
starRun(input1, input2, cores=1, star= "STAR", samtools="samtools", fa, alignment=c("se","pe"), ch
```

Arguments

input1	The R1 fastq of a pair-end
input2	The R2 fastq of a pair-end
cores	number of cores to be used by tophat with program name, e.g. /somewhere/tophat
star	full path of STAR with program name, e.g. /somewhere/STAR
samtools	full path of samtools
fa	full path and name of the fasta file of one of the set of fusions of interest, to be used to build the STAR database. The fusion nucleotide sequences was generated with the function chimeraSeqs
alignment	se means that both fastq files from the pair-end run are concatenated, pe means that tophat will use fastq files in pair-end mode
chimSegmentMin	STAR fusion parameter, see STAR manual
chimJunctionOverhangMin	STAR fusion parameter, see STAR manual

Value

The function create a folder called chimeraDB_time, where time is the time when the folder was created. STAR output will be located in the folder output_time, where time is the time when the folder was created. The bam file of interest is accepted_hits.bam.

Author(s)

Raffaele A Calogero

See Also

chimeraSeqs

Examples

```
#starRun(input1=paste(find.package(package="chimera"),"/examples/mcf7_sample_1.fq",sep=""), input2=paste(fi
```

subreadRun *A function to generate a bam file for fusions coverage evaluation*

Description

A function mapping reads to a chimera sequence set. The bam produced by this remapping on a putative fusion will be used to plot the coverage data for all the fused constructs. The function uses Rsubread aligner for MAC and UNIX OS. In case WINDOWS OS Rbowtie is used.

Usage

```
subreadRun(ebwt,input1, input2, outfile.prefix="accepted_hits", alignment=c("se","pe"),cores=1)
```

Arguments

ebwt	Full path and name of the fasta file of one of the set of fusions of interest, to be used to build the index database. The fusion nucleotide sequences can be generated with the function chimeraSeqs
input1	The R1 fastq of a pair-end
input2	The R2 fastq of a pair-end
outfile.prefix	Prefix of the output bam file. Default is accepted_hits
alignment	se means that both fastq files from the pair-end run are concatenated, pe means that tophat will use fastq files in pair-end mode
cores	Number of cores to be used by the aligner

Value

Standard bam file output. The bam file name by default is accepted_hits.bam.

Author(s)

Raffaele A Calogero

See Also

chimeraSeqs

Examples

```
if(require(Rsubread)){
 subreadRun(ebwt=paste(find.package(package="chimera"),"/examples/SULF2_ARFGEF2.fa",sep=""),
   input1=paste(find.package(package="chimera"),"/examples/mcf7_sample_1.fq",sep=""),
   input2=paste(find.package(package="chimera"),"/examples/mcf7_sample_2.fq",sep=""),
   outfile.prefix="accepted_hits", alignment="se", cores=1)
}
```

supportingReads *A function to extract supporting reads values from a list of fSet object*

Description

A function extracting supporting reads values from a list of fSet objects. Please note that not all outputs of supported tools provides both spanning reads, i.e. pair-end reads having one of the two mates spanning over the break point, and encompassing reads, i.e. pair-end reads having the two mates mapping on the exons of the two transcripts involved in the fusion. The presence of both type of reads is mandatory to provide the full number of reads covering the junction region. To know which information are provided by the supported tool please check fSet

Usage

```
supportingReads(list, fusion.reads=c("encompassing","spanning"), parallel=FALSE)
```

Arguments

list	a list of fSet objects
fusion.reads	it allows to extract spanning reads associated to the SeedCount slot or encompassing reads associated to the RescuedCount
parallel	option to run a parallel version of the function, default FALSE

Author(s)

Raffaele A Calogero

Examples

```
tmp <- importFusionData("fusionmap", paste(find.package(package="chimera"),"/examples/mcf7.FMFusionReport",
supporting.reads <- supportingReads(tmp, fusion.reads="spanning")
supporting.reads
```

tophatInstallation *A function to download tophat, bowtie and samtools*

Description

A function allowing the download and installation of tophat, bowtie and samtools in chimera package folder. The function also creates soft links in the user bin folder to allow the call of the above mentioned programs.

Usage

```
tophatInstallation(binDir, os=c("unix","mac"))
```

Arguments

binDir	The user bin folder
os	The supported operating systems

Author(s)

Raffaele A Calogero

Examples

```
#tophatInstallation(binDir="/somewhere/inyourpc/bin", os="mac")
```

---

tophatRun *A function to generate a bam file for fusions coverage evaluation*

Description

A function mapping reads to a chimera sequence set. The bam produced by this remapping on a putative fusion will be used to plot the coverage data for all the fused constructs. The function assumes that tophat is installed and located in the path. To run TopHat a softlink to bowtie or bowtie2 need to located in the user bin dir

Usage

```
tophatRun(input1, input2, output,cores=1, bowtie= c("bowtie","bowtie2"), tophat= "tophat",ebwt=pa
```

Arguments

input1	The R1 fastq of a pair-end
input2	The R2 fastq of a pair-end
output	Folder in which tophat will generate the output
cores	number of cores to be used by tophat with program name, e.g. /somewhere/tophat
bowtie	selecting bowtie or bowtie2 aligner
tophat	full path of tophat
ebwt	full path and name of the fasta file of one of the set of fusions of interest, to be used to build the bowtie database. The fusion nucleotide sequences was generated with the function chimeraSeqs
alignment	se means that both fastq files from the pair-end run are concatenated, pe means that tophat will use fastq files in pair-end mode

Value

TopHat standard output. The bam file of interest is accepted_hits.bam. The bam file will be then loaded in the slot fusionsLoc of the fSetSummary object from which fusions were retrieved.

Author(s)

Raffaele A Calogero

See Also

chimeraSeqs

Examples

```
#tophatRun(input1=paste(find.package(package="chimera"),"/examples/mcf7_sample_1.fq",sep=""), input2=paste(
```

validateSamFile *A function to validate SAM or BAM files*

Description

A function to validate SAM or BAM files using picard-tools

Usage

```
validateSamFile(input, output, mode=c("VERBOSE", "SUMMARY"), max.output="100")
```

Arguments

input	SAM/BAM file to be validated
output	file name in which to save the validation information
mode	Mode of output. Default value: VERBOSE
max.output	max number of reported error lines

Value

Validation information referring to a SAM/BM file.

Author(s)

Raffaele A Calogero

See Also

picardInstallation

Examples

```
#validateSamFile(input=paste(find.package(package="chimera"),"/examples/mcf7_trs_accepted_hits.bam",sep="")
```

Index