Chimera

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

May 28 2013

1 Introduction

The discovery of novel gene fusions can lead to a better comprehension of cancer progression and development. The emergence of deep sequencing of transcriptome, known as RNAseq, has opened many opportunities for the identification of this class of genomic alterations, leading to the discovery of novel chimeric transcripts in melanomas, breast cancers and lymphomas. Nowadays, various computational approaches have been developed for the detection of chimeric transcripts. Although all of these computational methods allow to detect fusions events, each one producing its own type of output. Outputs generated by fusion finders do not follow any standard structure and, as far as we know, no tools are available to analyse and manipulate, these output. Thus, we have developed *chimera*, which is a package for downstream processing of data obtained by the following fusion detection tools:

1. bellerophontes,
2. deFuse,
3. FusionFinder,
4. FusionHunter,
5. mapSplice,
6. tophat-fusion,
7. FusionMap
8. chimeraScan

1.1 Fusion finder tools

We have categorised the fusion detection algorithms into two classes:

1. Fragment-based approach
2. Pseudo-reference based approach

In the fragment based approach tools split the input reads into fragments, which are aligned with respect to a genome or whole transcriptome reference. The mapped fragments are then used to build a list of putative fusion events that are selected using several additional pieces of information or filter steps. This category includes the following tools: *FusionFinder, FusionMap, MapSlice, deFuse, chimeraScan*. Pseudo-reference based approaches are characterised by a combination of candidate fusion events, obtained after the first mapping phase, which are then used to generate a pseudo reference for chimeras detection. The candidate fusion events are filtered following several policies chosen by the authors. *TopHat-Fusion*, *deFuse* and *FusionHunter* are the tools included in this category.

2 Data structure

The chimera finder output is loaded in R a s list, made of objects of the class *fSet* for each fusion event.

An object of the *fSet* class is characterised by the following slots:

fusionInfo : a list embedding various characteristics of the fusion. The most interesting one is the *SeedCount* slot, which contains the number of reads supporting the fusion junction.

fusionLoc : embedding a GRangesList containing two GRanges objects, one for each gene involved in the fusion. Furthermore each Granges object also contains various informations about the fusion in the *elementMetadata* slot, as *KnownGene* referring to the gene involved in the fusion and the *FusionJunctionSequence*, which is the sequence involved in the fusion.

fusionRNA : embedding a DNAStringSet encompassing the possible fusion events obtainable using the gene isoforms encompassing the exons involved in the fusion.

fusionGA : a GAlignments object containing all the positions of the reads mapping over the fusion, which is useful to generate coverage information supporting the fusion.

2.1 fSet methods

The method *fusionData*, given a fSet object, returns information for a fusion, depending on the tool used only some of the fusion description information are available (fusionTool: used for the analysis, SeedCount: number of reads supporting the fusion junction, RescuedCount: number of reads supporting the fusion globally, i.e. spanning and encompassing reads, SplicePattern: spice patter used, FusionGene: transcripts involved in the fusion, frameShift: presence of a frame shift in the fusion event) The method *fusionGRL*, given a fSet object, returns the GRangesList with the information on the genomic location of the fusion boundaries for the two genes involved in the fusion. The method *fusionRNA* given a fSet object, returns the DNAStringSet of the putative fusions encompassing the exons involved in the fusion. The method *addDNA*, given a fSet object, allows to add the DNAStringSet of the putative fusions to an fSet object. The DNAStringSet of the putative fusions can be generated using the function *chimeraSeqs*, see section below. The method *fusionGA* given a fSet object, returns the GAlignments object of the putative fusions encompassing the exons involved in the fusion. The method *addGA*, given a fSet object, allows to add the GAlignments of the putative fusions to an fSet object. The DNAStringSet of the putative fusions can be generated using the function *tophatRun*, see section below. The function *addGA* sorts, indexes and loads the bam generated by TopHat as a GAlignments object.

```
> #creating a fusion report from output of fusionMap
> library(chimera)
> tmp <- importFusionData("fusionmap", paste(find.package(package="chimera"),
+ "/examples/mcf7.FMFusionReport", sep=""), org="hs")
> #extracting the fSet object for 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)
```

3 Functions

The function *importFusionData* allows to import in a list outputs generated by bellerophontes, defuse, fusionfinder, fusionhunter, mapsplice, tophat-fusion, fusionmap, chimerascan.

The function *supportingReads* allows to extract the number of reads supporting each of the detected fusions. It is notable that the same fusion gene is detected with a different number of supporting read depending on the fusion tool used. Furthermore, each tool detects a different number of fusions when querying the same data set. same apply to the supporting reads.

```
> supporting.reads <- supportingReads(tmp)
> supporting.reads

 [1] 1 1 1 1 1 1 1 1 1 1 1 1 2
```

The function *fusionName* allows to extract fusion names from a list of fSet objects.

```
> fusion.names <- fusionName(tmp)
> fusion.names

 [1] "HMGN2:ESYT1"           "CC2D1B:DTYMK"          "NOS1AP:C1orf226"
 [4] "GREB1:GREB1"           "RYBP:YAF2"             "SLC30A5:AZIN1"
 [7] "TUBB:KRT80"            "EEF1A1:GHITM"          "HNRNPK:AATF"
[10] "NDUFA1:SYNJ2BP-COX16" "FAM208B:FAM208B"       "YLPM1:ITPK1"
[13] "SULF2:ARFGEF2"
```

The function *chimeraSeqs* allows to generate the chimera nucleotide sequence. The output is a DNAStringSet object encompassing the fusions generated using all the isoforms for each gene involved in the fusion.

```
> myset <- tmp[[13]]
> trs <- chimeraSeqs(myset, type="transcripts")
```

The function *subreadRun* allows to map reads to a chimera sequence set generated by *chimeraSeqs*. The function produces a standard output bam file (accepted_hits.bam). The bam produced by this remapping on putative fusions can be used to generate the coverage plot for all the fused constructs.

Using *write.XStringSet* function from Biostring package DNAStringSet can be saved in fast format. The function *tophatRun* maps reads to a chimera sequence set, eg. *trs*.

```
> #the DNAStringSet of transcript fusions sequences is saved as fast file
> #write.XStringSet(trs, paste("SULF2_ARFGEF2.fa",sep=""), format="fasta")
>    if(require(Rsubread)){
+             subreadRun(ebwt=paste(find.package(package="chimera"),"/examples/SULF2
+              input1=paste(find.package(package="chimera"),"/examples/mcf7_sample_1
+              input2=paste(find.package(package="chimera"),"/examples/mcf7_sample_2
+              outfile.prefix="accepted_hits", alignment="se", cores=1)
+    }
```

The function *filterList* allows to filter a list of fSet objects on the basis of supporting reads or fusion names or presence of intronic sequence in the fusion. The rational of filtering on the basis of supporting reads is that biological effect also depends on the amount of the expressed mRNA, thus highly expressed fusions, i.e. fusions with an high number of junction-spanning reads, might have a more important role in cancer physiology. On the other hand the presence e of an intron in a fusion generates a very large transcript, which do not produce in frame transcripts. Furthermore it is possible to remove read-through events, i.e. fusion in which different exons of the same gene are recognised as a fusion being separated by extremely long introns. It is also possible to retain only fusions in which only annotated names are considered, eg. PID1:TP53 is retained and PID1:chr2:133038625-133038655 is removed.

```
> tmp1 <- filterList(tmp, type="fusion.names", fusion.names[c(1,3,7)])
> tmp2 <- filterList(tmp, type="supporting.reads", 2)
> #tmp3 <- filterList(tmp, type="intronic")
```

Coverage can be visualised with the function *plotCoverage*. Below three examples of the visualisation of a fusion based on exon coverage, junctions coverage or coverage at the fusion boudaries.

```
> tmp <- importFusionData("fusionmap", paste(find.package(package="chimera"),
+ "/examples/mcf7.FMFusionReport", sep=""), org="hs")
> fusion.names <- fusionName(tmp)
> 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=""))
> pdf("coverage1.pdf")
> plotCoverage(myset, plot.type="exons", col.box1="red",
+ col.box2="green", ybox.lim=c(-4,-1))
> dev.off()

null device
          1

> pdf("coverage2.pdf")
> plotCoverage(myset, plot.type="junctions", col.box1="red",
+ col.box2="yellow", ybox.lim=c(-4,-1))
> dev.off()

null device
          1
```

```
> pdf("coverage3.pdf")
> plotCoverage(myset, junction.spanning=100, fusion.only=TRUE, col.box1="red",
+ col.box2="yellow", ybox.lim=c(-4,-1))
> dev.off()

null device
          1
```

The coverage at the fusion boundary, can be generated using the parameter fusion.only sets on TRUE. To extend the region around the fusion location, it is possible to change the number of nucleotides spanning around the fusion location, using the parameter junction.spanning, which is set to 20 nts as default.

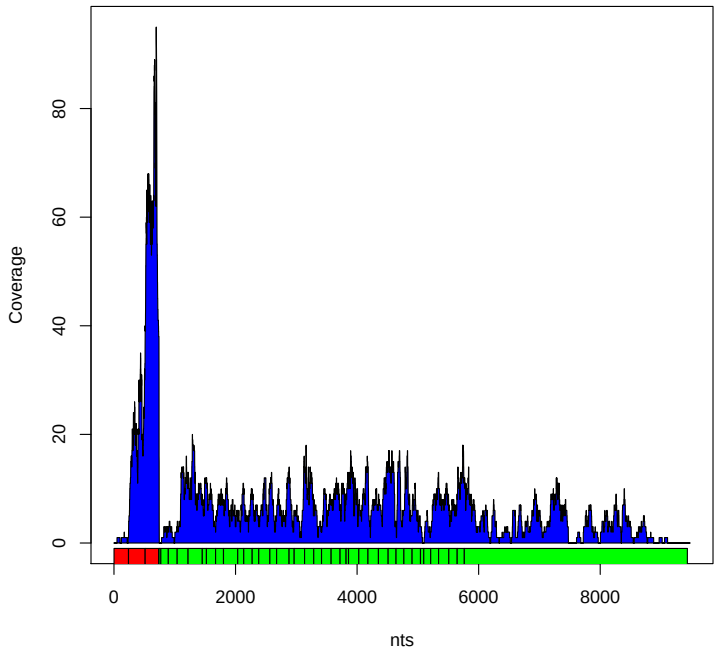

Figure 1: Exons coverage plotted with respect to the fused transcript structure. Exons of gene 1 are shown in red, exons of gene 2 are shown in yellow.

It is also possible to reconstruct the protein sequence involved in the fusion using the function *fusionPeptides*. The function returns the donor and acceptor transcripts and the corresponding peptides as part of a list.

```
> mypeps <- fusionPeptides(fset=myset, which.isoform=1,
+ donor.up=200, acceptor.down=200)
```

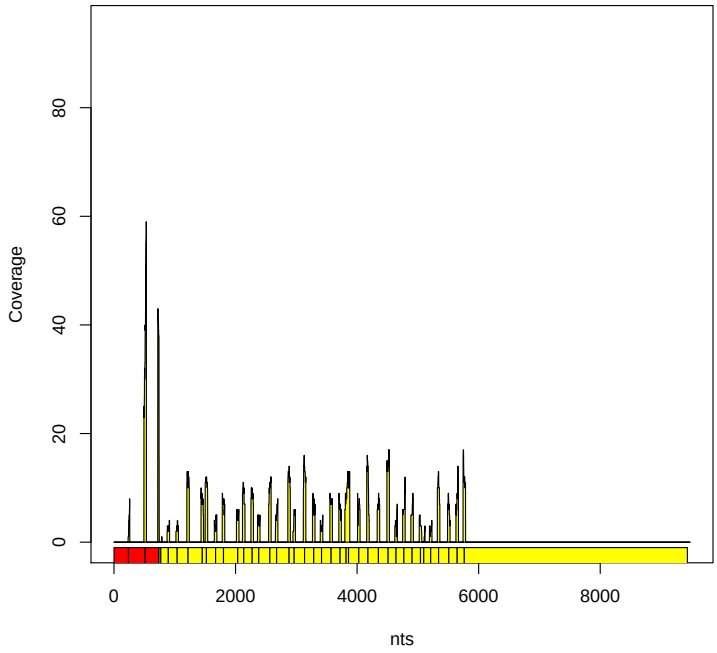

Figure 2: Exon/Exon junctions coverage plotted with respect to the fused transcript structure. Exons of gene 1 are shown in red, exons of gene 2 are shown in yellow.

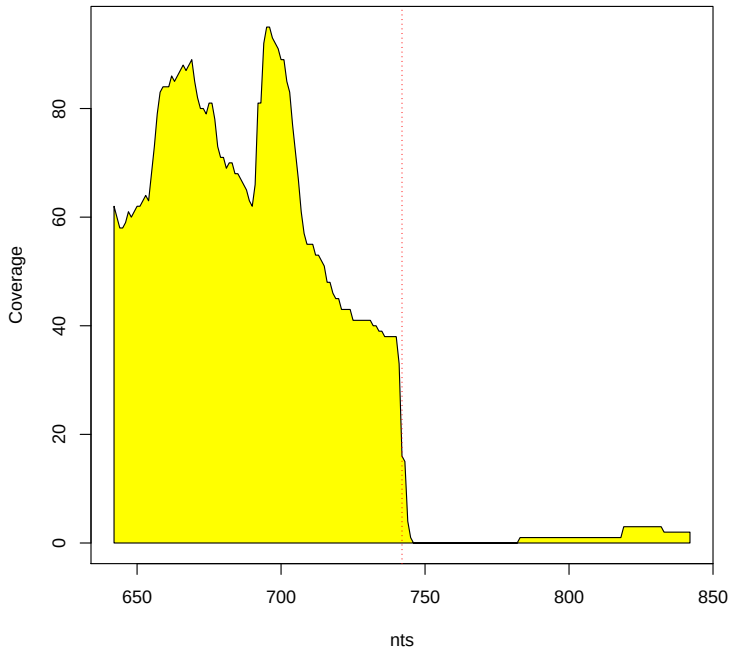

Figure 3: Fusion boundaries coverage plotted at the level at the fusion location. In this case it is clear that reads spanning over the fusion boundary (dashed red line) are not detected

```
CDS of gene1 is on frame 2

CDS of gene2 is on frame 3

> mypeps

$selected.alignment
[1] 1

$transcript1
  745-letter "DNAString" instance
seq: GAGCGAGAGTGTGTCGAGTGAGTGTGCGTCTGTGTG...TTGCCGTGTACCTCAATAGCACTGGCTACCGGACAG

$pep1
  248-letter "AAString" instance
seq: ERECVE*VCVCVSRRGCALGAGSAASRVRRHREVES...NHNTYTNNENCSSPSWQAQHESRTFAVYLNSTGYRT

$frame.pep1
[1] 2

$transcript2
  8732-letter "DNAString" instance
seq: GATGAAATTAAAGCAGAAATAGAAAAGCAGAGGCTT...TGATCAGGTGGTACATCAATAAAATTTTAAAAAGTA

$pep2
  2910-letter "AAString" instance
seq: DEIKAEIEKQRLGTAAPPKANFIEADKYFLPFELAC...YNMKCYVKFSM*KYYV*YY*NIPCFDQVVHQ*NFKK

$frame.pep2
[1] 3

$fusion.status
[1] "In frame"

$validation.seq
[1] "AGCAGGGCGGGGCGCACTTCATCAACGCCTTCGTGACCACACCCATGTGCTGCCCCTCACGCTCCTCCATCCTCACTGGCA

$junction.ga
GappedAlignments with 0 alignments and 0 metadata columns:
   seqnames strand       cigar    qwidth     start       end     width
      <Rle>  <Rle> <character> <integer> <integer> <integer> <integer>
        ngap
   <integer>
```

```
---
seqlengths:
 uc002xto.3:uc002xtx.4
                  9477
```

Furthermore, the output of the *fusionPeptides*, provides a DNA region encompassing the fusion that can be used to be upload in Primer3, `http://simgene.com/Primer3`, to design the PCR primers to validated the fusion event or to obtain the region encompassing the fusion that can be cloned and sequenced with Sanger method. The parameter *donor.end*, *acceptor.end* allow to define the extension of the DNA fragment to be extracted, respectively upstream to the donor end and down stream the acceptor start.

4 FusionMap

FusionMap aligns fusion reads directly to the genome without prior knowledge of potential fusion regions. FusionMap can detect fusion events in both single- and paired-end datasets from either RNA-Seq or gDNA-Seq studies and characterize fusion junctions at base-pair resolution: `http://www.omicsoft.com/fusionmap` The software splits reads into smaller fragments and finds fusion candidates aligning these fragments to genes annotated on genomic reference. The read alignment is performed by GSPN algorithm, integrated in the tool, with up to 2 base mismatches of tolerance. Two 25 bp seeds at each side of unmapped read are extracted and aligned to the reference. Putative fusions are reported only if both seeds align. Fusions events characterized by fusion boundaries distant less than 5 bp are aggregated and used for junction refinement. A scoring system based on canonical splicing patterns is then used to define the position of the fusion boundary. False positives are removed using the following four steps: (i) reads with break point score above a threshold are removed; (ii) fusions nearer than 5kb (partial removal of read-through events) are ignored; (iii) the fusion source sequences are concatenated, creating a pseudo-reference to be used to align unmapped reads. Fusion candidates with no reads aligned to the pseudo-reference are removed; and (iv) fusions with less than 2 reads aligned on distinct regions (PCR artifact removal) are removed.

5 FusionHunter

FusionHunter is an open-source software tool, which reliably identifies fusion transcripts from transcriptional analysis of paired-end RNA-seq `http://bioen-compbio.bioen.illinois.edu/FusionHunter/`. FusionHunter maps the input paired-end reads against a reference genome using Bowtie. The mapped reads are used to identify the fusion candidates. These candidates are combined to generate a pseudo reference used to identify junction-spanning reads. Two genomic regions are marked as fusion candidate if

there are two transcripts enriched by input reads and the fusion junction between them is supported by at least two different paired-end reads. Each fusion candidate composed by two genes sharing significant homology is removed. Candidates are analyzed to estimate their orientation and are concatenated into a pseudo-reference. Unmapped reads are split in segments and mapped on the pseudo-reference. If one segment is correctly aligned, the tool searches for the nearest canonical splicing junction and aligns the other part of the original read with that region. Several filters are applied to (i) remove reads aligned on the break point if anchored to one gene with less than 6 bp; (ii) remove fusion events with no reads aligned on the break point: (iii) keep only one read in case multiple reads are stacked (PCR artifact removal); and (iv) remove read-through events if not available in the human EST database.

6 FusionFinder

FusionFinder is a Perl-based software designed to automate the discovery of candidate gene fusion partners from single-end (SE) or paired-end (PE) RNA-Seq read data. `http://bioinformatics.childhealthresearch.org.au/software/fusionfinder/`. FusionFinder splits reads into fragments and reports fusion candidates when these fragments align to genes annotated on genomic reference. The main differences with respect to FusionMap are the tools used for alignment and the filter implementation. Bowtie is used to align reads with respect to coding reference transcriptome. Two fragments at each side of unmapped read, namely pseudo-Paired-End (PE) reads, are extracted. The pseudo-PE reads are aligned with Bowtie with respect to the coding reference transcriptome considering up to two mismatches. A list of exons involved in the fusion can be reported by the identification of the closest ENSEMBL exons. Finally, multiple filtering steps are used to refine the results: (i) pairs of seeds mapping on the same gene are removed; (ii) pairs on the same chromosome but on opposite strands (antisense transcript removal) are removed; (iii) the pairs are mapped on the genome. Pairs mapped at a distance higher than read length are removed; and (iv) all possible artifacts caused by sequence similarity are also removed.

7 deFuse

deFuse is a computational method for fusion discovery in tumor RNA-Seq data. Unlike existing methods that use only unique best-hit alignments and consider only fusion boundaries at the ends of known exons, deFuse considers all alignments and all possible locations for fusion boundaries. `http://sourceforge.net/apps/mediawiki/defuse/index.php?title=DeFuse`. deFuse uses reads pairs with discordant alignments to define putative fusion events on the basis of two conditions: the region covered by different reads must overlap and the shift between overlapping reads must be coherent with the fragment length. Each paired-end read with discordant alignment is then assigned to a

putative fusion in order to minimize the number of reported events. For each putative fusion an estimation of fusion boundary position is used to detect encompassing reads and to map the fusion boundary position at nucleotide level. This information is also used to discard read pairs aligned located at a distance not compatible with the expected distribution of sequenced fragments distance.

8 mapSplice

MapSplice can be applied to both short (<75?bp) and long reads (?75?bp). MapSplice is not dependent on splice site features or intron length, consequently it can detect novel canonical as well as non-canonical splices. MapSplice leverages the quality and diversity of read alignments of a given splice to increase accuracy. `http://www.netlab.uky.edu/p/bioinfo/MapSplice`. MapSplice starts splitting each read in a set of consecutive segments, having size smaller than half of the read size. The exon alignment of segments is performed using Bowtie and BWA or SOAP2, BFAST and MAQ, with an input specified mismatch tolerance. For each read, MapSplice aligns segments not mapped in the previous step, exploiting the information derived by the other aligned segments. Finally, the splice junction quality of fusion events is evaluated according with two statistical measures: the ?anchor significance?, determined by an alignment that maximizes significance as a result of long anchors on each side of the splice junction, and the ?entropy? measured by the diversity of splice junction positions.

9 TopHat-fusion

TopHat-Fusion is an algorithm designed to discover transcripts representing fusion gene products, which result from the breakage and re-joining of two different chromosomes, or from rearrangements within a chromosome. TopHat-Fusion is an enhanced version of TopHat, an efficient program that aligns RNA-seq reads without relying on existing annotation. Because it is independent of gene annotation, TopHat-Fusion can discover fusion products deriving from known genes, unknown genes and unannotated splice variants of known genes. `http://tophat.cbcb.umd.edu/fusion_index.html`. TopHat-Fusion uses Bowtie to detect all reads aligning entirely within exons, and creates a set of partial exons from these alignments. Then, hypothetical intron boundaries are created between the partial exons, and Bowtie is used to re-align the initially unmapped reads and find those that define introns. Each read is split into segments of 25 bp and each segment mapped on the genome. Putative fusions are reported if segments map in a way consistent with fusions (using TopHat with relaxed parameters). Several filters are applied after candidate definition to (i) remove candidate fusions on multi-copy genes or repetitive sequences; (ii) remove reads anchored with less than 13 bp on either side of the fusion; and (iii) remove candidate fusions from regions nearer than 100 kb (read-through events removal). TopHat-Fusion trims 22 bp segments flanking each fusion point, it con-

structs spliced fusion contigs and builds an index for them. Each segment is re-mapped on the new contigs and the results are stitched together to produce the full read alignment. The algorithm evaluates then the contradicting reads, i.e. the reads fully mapped on a single part of the fusion and overlapping the fusion boundary. Finally, the tool removes fusion events after the junction definition, if both sides are not annotated.

10 Bellerophontes

Bellerophontes is a fully automated framework for the detection of novel fusion transcripts in paired end RNA-Seq data `http://eda.polito.it/bellerophontes/index.html`. It detects putative fusion genes by searching those reads that discordantly matches on different genes. Then, the tool applies several modular filters in order to select those fused genes matching an accurate gene fusion model based on experimental evidences reported in recent literature. Bellerophontes runs on top of TopHat and Cufflinks tools. The analysis is based on the results of TopHat alignment and Cufflinks transcript isoform detection.

11 chimeraScan

ChimeraScan uses Bowtie to align paired-end reads to a combined genome-transcriptome reference. read pairs that could not be aligned concordantly are trimmed into smaller segments (default = 25 bp) and realigned. Trimming increases the chance that neither read alignment spans a chimeric junction, thereby improving sensitivity for nominating chimeras. the trimmed alignments are scanned for evidence of discordant read pairs, or reads that align to distinct references or distant genomic locations (as determined by the fragment size range) of the same reference. Reads aligning to overlapping transcripts are not considered discordant. ChimeraScan clusters the discordant reads and produces a list of putative transcript pairs that serve as chimera candidates. after spanning reads are incorporated, ChimeraScan filters chimeras with few supporting reads and chimeras with fragment sizes far outside the range of the distribution. When isoforms of the same gene support a fusion ChimeraScan only retains the isoform(s) with highest coverage. ChimeraScan produces a tabular text file describing each chimera.

12 STAR

Spliced Transcripts Alignment to a Reference (STAR) software is based on a previously un-described RNA-seq alignment algorithm which utilizes sequential maximum mappable seed search in uncompressed suffix arrays followed by seed clustering and stitching procedure. STAR can discover non-canonical splices and chimeric (fusion) transcripts. If the best scoring alignment window does not cover the entire read, STAR reports chimeric

connections to the other windows that cover portions of the read not covered by the main window. These chimeric connections between windows can span long distance on the same strand, or different strands on the same chromosome, or different chromosomes. The *importFusionData* allows to import the file Chimeric.out.junction. Furthermore, STAR can be installed in the chimera folder using the function *starInstallation*, and STAR can be run using *starRun*.

13 example

The package comes with a set of data, present in the example folder, representing all the information needed to create an fSet object for the fusion SULF2:ARFGEF2 detected by FusionMap in the MCF7 dataset published by Edgren et al. Genome Biology 2011, 12:R6.

```
> dir(paste(find.package(package="chimera"),"/examples/",sep=""))

[1] "SULF2_ARFGEF2.fa"
[2] "mcf7.FMFusionReport"
[3] "mcf7_sample_1.fq"
[4] "mcf7_sample_2.fq"
[5] "mcf7_trs_accepted_hits.bam"
[6] "mcf7_trs_accepted_hits.bam_sorted.bam"
[7] "mcf7_trs_accepted_hits.bam_sorted.bam.bai"

mcf7.FMFusionReport is the output of FusionMap and can be imported using   importFus
mcf7_sample_1.fq and mcf7_sample_2.fq are fastq files to be used for  coverage estim
using tophatRun on  the fusion SULF2_ARFGEF2.fa.
mcf7_trs_accepted_hits.bam, mcf7_trs_accepted_hits.bam_sorted.bam and
mcf7_trs_accepted_hits.bam_sorted.bam.bai are the output of tophatRun
and they can be used to generate the GAlignments object to be loaded in the fSet obj
using addGA function.
```

14 R-Bioconductor information

```
> sessionInfo()

R version 3.0.1 (2013-05-16)
Platform: i386-w64-mingw32/i386 (32-bit)

locale:
[1] LC_COLLATE=C
[2] LC_CTYPE=English_United States.1252
```

```
[3] LC_MONETARY=English_United States.1252
[4] LC_NUMERIC=C
[5] LC_TIME=English_United States.1252

attached base packages:
[1] parallel  stats     graphics  grDevices utils     datasets  methods
[8] base

other attached packages:
 [1] chimera_1.2.6
 [2] TxDb.Hsapiens.UCSC.hg19.knownGene_2.9.2
 [3] GenomicFeatures_1.12.2
 [4] BSgenome.Hsapiens.UCSC.hg19_1.3.19
 [5] BSgenome_1.28.0
 [6] org.Mm.eg.db_2.9.0
 [7] org.Hs.eg.db_2.9.0
 [8] RSQLite_0.11.4
 [9] DBI_0.2-7
[10] AnnotationDbi_1.22.6
[11] Rsamtools_1.12.3
[12] Biostrings_2.28.0
[13] GenomicRanges_1.12.4
[14] IRanges_1.18.1
[15] Biobase_2.20.0
[16] BiocGenerics_0.6.0

loaded via a namespace (and not attached):
[1] RCurl_1.95-4.1     XML_3.96-1.1       biomaRt_2.16.0     bitops_1.0-5
[5] rtracklayer_1.20.2 stats4_3.0.1       tools_3.0.1        zlibbioc_1.6.0
```