**SFyNCS User Manual**

**Version 0.14**

**I. Introduction**

Somatic Fusions involving Non-Coding Sequences (SFyNCS) detects fusions involving both protein-coding genes and non-coding sequences from paired-end RNA-seq data. It uses discordant read pairs and split reads to detect the fusions and refine fusion breakpoints to the basepair resolution.

**II. Prerequisites**

1. **Software environment**
   1. Unix/Linux system
   2. Perl (v5.010 or above), can be downloaded at <https://www.perl.org/get.html> or installed with conda (<https://anaconda.org/conda-forge/perl>).
   3. Ptyhon2 (tested v2.7.5, v2.7.11), can be downloaded at <https://www.python.org/downloads> or installed with conda (<https://anaconda.org/anaconda/python>, please specify version when installing).
   4. Perl (v5.010 or above), can be downloaded at <https://www.perl.org/get.html> or installed with conda (<https://anaconda.org/conda-forge/perl>).
   5. BLAT (tested v35, v36), can be downloaded at http://hgdownload.soe.ucsc.edu/admin/exe or installed with conda (<https://anaconda.org/bioconda/blat>).
   6. Bedtools (tested v2.25.0, v2.26.0, v2.27.1), can be downloaded at <https://github.com/arq5x/bedtools2/releases> or installed with conda (<https://anaconda.org/bioconda/bedtools>).
   7. Bowtie2 (tested v2.1.0, v2.2.5, v2.2.9, v2.3.0, v2.3.2, v2.3.4.3), can be downloaded at <https://sourceforge.net/projects/bowtie-bio/files/bowtie2/> or installed with conda (<https://anaconda.org/bioconda/bowtie2>).
   8. STAR (v2.6.1a or above, tested v2.6.1a, v2.6.1d, v2.7.0f), can be downloaded at <https://github.com/alexdobin/STAR/releases> or installed with conda (<https://anaconda.org/bioconda/star>).
   9. Samtools (tested v0.1.19, v1.1, v1.3.1, v1.5), can be downloaded at <https://github.com/samtools/samtools/releases> or installed with conda (<https://anaconda.org/bioconda/samtools>).
   10. TopHat2 (must be v2.1.0), can be downloaded at http://ccb.jhu.edu/software/tophat/downloads or installed with conda (<https://anaconda.org/bioconda/tophat>, please specify version when installing).

Note that TopHat2 needs to be v2.1.0 (2.1.1 or 2.0.13 won't work). Follow these steps to install:

wget <http://ccb.jhu.edu/software/tophat/downloads/tophat-2.1.0.Linux_x86_64.tar.gz>

tar -zxvf tophat-2.1.0.Linux\_x86\_64.tar.gz

export PATH=$PWD/tophat-2.1.0.Linux\_x86\_64:$PATH

chmod u+x tophat-2.1.0.Linux\_x86\_64/\*

* 1. It will take 14 minutes to install all above tools using conda:

conda create --no-default-packages -n SFyNCS

source activate SFyNCS

conda install -c conda-forge perl=5.26.2

conda install -c anaconda python=2.7.11

conda install -c bioconda blat=36

conda install -c bioconda bedtools=2.27.1

conda install -c bioconda bowtie2=2.3.4.3

conda install -c bioconda star=2.7.0f

conda install -c bioconda samtools=1.3.1

conda install -c bioconda tophat=2.1.0

1. **Reference, index and annotation files**
   1. Reference genome sequence in fasta format
   2. STAR index of the reference genome (can be skipped if Chimeric.out.junction is available). The STAR index can be built with the following command. We suggest using chr1-chr22, chrX, chrY and chrM only. Please specify the number\_of\_thread in the command below.

mkdir -p /path\_to/star\_index\_dir

STAR --runThreadN number\_of\_thread --runMode genomeGenerate --genomeDir /path\_to/star\_index\_dir --genomeFastaFiles /path\_to/genome.fasta

* 1. TopHat2/Bowtie2 index of the reference genome. The TopHat2/Bowtie2 index can be built with the following command. Please specify file\_prefix in the commands below.

mkdir -p /path\_to/tophat\_index\_dir

bowtie2-build /path\_to/genome.fasta /path\_to/tophat\_index\_dir/file\_prefix

ln -s /path\_to/genome.fasta /path\_to/tophat\_index\_dir/file\_prefix.fa

* 1. Fai index of the reference genome (automatically generated by samtools)
  2. Gene annotation of the reference genome in Gene Predictions Extended (gpe) format (please refer to <https://genome.ucsc.edu/FAQ/FAQformat.html#format9> for more detail). A header is needed (can be any artifact header). X chromosome should be chrX and Y chromosome should be chrY. Gencode\_v29\_hg38.gpe is provided with the package.

Example of gene annotation:

Column 1: Transcript ID (e.g., ENST00000485503.1)

Column 2: Chromosome (e.g., chr7)

Column 3: Strand (e.g., +)

Column 4: Transcript start position (e.g., 55192810)

Column 5: Transcript end position (e.g., 55200802)

Column 6: Coding region (CDS) start position (e.g., 55200802)

Column 7: Coding region end position (e.g., 55200802)

Column 8: Number of exons (e.g., 3)

Column 9: Exon start positions (e.g., 55192810,55198716,55200315)

Column 10: Exon end positions (e.g., 55192841,55198863,55200802)

Column 11: Score (e.g., 0)

Column 12: Gene symbol (e.g., EGFR)

Column 13: Status of CDS start annotation (none, unknown, incomplete, or complete. e.g., none)

Column 14: Status of CDS end annotation (none, unknown, incomplete, or complete. e.g., none)

Column 15: Exon frame offsets (e.g., -1,-1,-1)

**3. Input files**

1. Two paired-end fastq.gz files or fastq files. SFyNCS uses STAR to align reads and use Chimeric.out.junction generated by STAR to call fusion candidates. If users have Chimeric.out.junction prior to running SFyNCS, the STAR alignment step can be skipped, and it can speed up the process significantly.

Chimeric.out.junction file can be produced by STAR with following command, please specify /path\_to/star\_index\_dir, number\_of\_thread and /path\_to/star\_output\_dir in the command:

STAR --genomeDir /path\_to/star\_index\_dir \

--readFilesIn 1.fastq.gz 2.fastq.gz \

--readFilesCommand zcat \

--runThreadN number\_of\_thread \

--outFileNamePrefix /path\_to/star\_output\_dir \

--outReadsUnmapped None \

--twopassMode Basic \

--outSAMstrandField intronMotif \

--outSAMunmapped Within \

--chimSegmentMin 12 \

--chimJunctionOverhangMin 12 \

--chimOutJunctionFormat 1 \

--alignSJDBoverhangMin 10 \

--alignMatesGapMax 100000 \

--alignIntronMax 100000 \

--alignSJstitchMismatchNmax 5 -1 5 5 \

--outSAMattrRGline ID:GRPundef \

--chimMultimapScoreRange 10 \

--chimMultimapNmax 10 \

--chimNonchimScoreDropMin 10 \

--peOverlapNbasesMin 12 \

--peOverlapMMp 0.1 \

--outSAMtype BAM SortedByCoordinate \

--genomeLoad NoSharedMemory

1. Fusion breakpoints from normal samples. They are used to filter out germline events as well as artifacts. We provide fusion breakpoint files from TCGA normal samples (<https://drive.google.com/drive/folders/1KatN9WevL_N9QGT9UVtSVRo5BKn47MGx?usp=sharing>). Users can generate their own if RNAseq data from normal samples are available:
2. Generate Chimeric.out.junction for each normal sample (refer to 3.1/II).
3. Format split reads and discordant read pairs information in Chimeric.out.junction.

perl /path/to/format\_STAR\_chimeric\_file.pl Chimeric.out.junction >format\_chimeric.tsv

1. Remove duplicated reads and get the number of supporting reads for each breakpoint.

perl /path/to/get\_juntions\_in\_normal\_sample.pl format\_chimeric.tsv >no\_duplication\_junction\_read\_count.tsv

Example of normal breakpoint file (no\_duplication\_normal\_junctions.tsv):

Column 1: Chromosome of breakpoint 1 (e.g., chr1)

Column 2: Position of breakpoint 1 (e.g., 10005917)

Column 3: Strand of breakpoint 1 (e.g., +)

Column 4: Chromosome of breakpoint 2 (e.g., chr1)

Column 5: Position of breakpoint 2 (e.g., 10005936)

Column 6: Strand of breakpoint 2 (e.g., +)

Column 7: Breakpoint supported read count (e.g., 1)

1. Rename each sample’s no\_duplication\_junction\_read\_count.tsv and put them under the same directory. Replace “sample\_1” with each sample id.

mv no\_duplication\_junction\_read\_count.tsv sample\_1.tsv

1. Optional: each normal junction file can be compressed.

gzip sample\_1.tsv

**III. Example**

Note: all files under the example directory are for testing only.

1. Download and decompress SFyNCS (the link is not available yet):

wget [https://github.com/yanglab-computationalgenomics/SFyNCS/](https://github.com/yanglab-computationalgenomics/SFyNCS/archive/refs/heads/main.zip)

mv master.zip SFyNCS.zip

unzip SFyNCS.zip

git clone https://github.com/yanglab-computationalgenomics/SFyNCS.git

1. Decompress example.tar.gz and enter example directory:

cd SFyNCS

tar -zxvf example.tar.gz

cd example

1. Build STAR index:

mkdir star\_index

STAR --runThreadN 1 --runMode genomeGenerate --genomeDir star\_index --genomeFastaFiles toy\_reference\_genome\_sequence.fasta

1. Build TopHat2/Bowtie2 index:

mkdir -p tophat\_index

bowtie2-build toy\_reference\_genome\_sequence.fasta tophat\_index/tophat

ln -s $PWD/toy\_reference\_genome\_sequence.fasta $PWD/tophat\_index/tophat.fa

1. Run SFyNCS:
   1. Start from fastq.gz files:

bash ../run\_SFyNCS.sh -p 1 -o demo\_output -a toy\_gene\_annotation.gpe -g toy\_reference\_genome\_sequence.fasta -s star\_index -t tophat\_index/tophat -d toy\_normal\_directory toy\_pair\_end\_reads\_1.fastq.gz toy\_pair\_end\_reads\_2.fastq.gz

* 1. Start from Chimeric.out.junction produced by STAR and fastq.gz files:

bash ../run\_SFyNCS.sh -p 1 -c Chimeric.out.junction -o demo\_output -a toy\_gene\_annotation.gpe -g toy\_reference\_genome\_sequence.fasta -t tophat\_index/tophat -d toy\_normal\_directory toy\_pair\_end\_reads\_1.fastq.gz toy\_pair\_end\_reads\_2.fastq.gz

It will take 1 minute to run the example. Users should get fusions.tsv.gz and fusions\_abridged.tsv.gz under demo\_output, please make sure the content in these two output files is the same as the files under example.

**IV. Workflow**

1. **Running SFyNCS**

SFyNCS can be run by the following commands. Users can further provide "-p thread\_numbers" to speed up the steps of STAR and TopHat2.

* 1. Start from fastq.gz files:

/path/to/run\_SFyNCS.sh -o /path\_to/output\_dir -a /path\_to/gene\_annotation.gpe -g /path\_to/genome.fasta -s /path\_to/star\_index\_dir -t /path\_to/tophat\_index\_dir/file\_prefix -d /path/to/normal\_junction\_directory 1.fastq.gz 2.fastq.gz

* 1. Start from Chimeric.out.junction produced by STAR and fastq.gz files:

/path/to/run\_SFyNCS.sh -c /path\_to/Chimeric.out.junction -o /path\_to/output\_dir -a /path\_to/gene\_annotation.gpe -g /path\_to/genome.fasta -t /path\_to/tophat\_index\_dir/file\_prefix -d /path/to/normal\_junction\_directory 1.fastq.gz 2.fastq.gz

* 1. run\_SFyNCS.sh options. Note that the default parameters are optimized for TCGA data.

-a --annotation\_file STR Gene annotation gpe file (section 2.5/II)

-g --genome\_fasta STR Reference genome fasta file

-o --output\_directory STR Output directory [default: current directory]

-p --thread\_number INT Number of threads [default: 1]. Multiple threads can speed up the steps of STAR and TopHat2

-s --star\_index STR Path to STAR index. This option can be skipped if "-c" is provided

-t --tophat\_index STR Path to TopHat2 index. It includes the name of any of index files up to but not including the first period

-c --chimeric\_file STR Chimeric.out.junction file generated by STAR

-d --normal\_junction\_dir STR Directory contains normal samples' junctions (section 3.2/II)

--adjust\_adjacent\_distance INT Breakpoints within this distance will be adjusted [default: 5]

--cluster\_distance INT Split reads and read pairs within this distance will be clustered together [default: 1000000]

--min\_split\_reads INT Minimal number of split reads for a fusion transcript to be identified [default: 1]

--min\_read\_pairs INT Minimal number of discordant read pairs for a fusion transcript to be identified [default: 1]

--min\_total\_reads INT Minimal number of total reads (split reads and discordant read pairs combined) for a fusion transcript to be identified [default: 3]

--overhang\_length INT Breakpoint overhang length for TopHat2 [default: 5]

--read\_pair\_distance INT Maximal distance between the TopHat2 alignment of read pairs and breakpoint [default: 10000]

--motif\_searching\_length\_in\_blat INT Splice site motifs (GT in the donor, AAG/CAG/TAG in the acceptor) are searched within this window size of breakpoints [default: 5]

--outside\_length\_in\_blat INT Specify the window of breakpoint flanking sequence for artificial reference [default: 1000000]

--inside\_length\_in\_blat INT Specify the window of breakpoint flanking sequence for artificial reference [default: 100]

--length\_for\_identity\_in\_blat INT Flanking sequences size of both fusion breakpoints to calculate sequence identity [default: 10]

--align\_percentage\_in\_blat FLOAT Minimal percentage of bases of the whole read that is alignable by BLAT for split reads [default: 0.9]

--max\_split\_read\_blat\_distance INT Distance between BLAT and STAR alignments of split reads [default: NA, filter not applied]

--max\_sequence\_identity\_in\_blat FLOAT Maximal sequence identity between flanking sequences of two fusion breakpoints [default: 0.8]

--filter\_by\_canonical\_splice\_motif STR Filter by canonical splice site motif [default: Y]

--length\_in\_sd INT Breakpoint flanking size to calculate standard deviation [default: 100]

--sd\_cutoff FLOAT Standard deviation cutoff to filter fusions [default: 0.15]

--filter\_in\_the\_same\_gene STR Filter fusions in the same gene [default: Y]

--normal\_adjacent\_distance INT Window size to search for breakpoints in normal samples [default: 10000]

--normal\_read\_count\_cutoff INT Filter fusions if numbers of reads (discordant pairs or split reads) in normal samples are equal to or more than the specified value [default: 2]

--deletion\_like\_distance INT Minimal distance allowed between fusion breakpoints for deletion-like fusions [default: 500000]

--duplication\_like\_and\_inversion\_like\_distance INT Minimal distance allowed between breakpoints for duplication-like and inversion-like fusions [default: 50000]

“adjust\_adjacent\_distance” controls how to adjust adjacent breakpoints. A read might be aligned to a locus in different ways, STAR will not always choose the same position for reads aligned to the locus (**Figure 1**), resulting in multiple nearby breakpoints. Since they are likely from the same breakpoint, SFyNCS will adjust them to use the same breakpoint if their breakpoints are within the defined distance. That is, split reads with same orientations and within the range defined by this parameter were considered to support the same fusion. Users can use the default parameter.

Graphical user interface, application

Description automatically generated with medium confidence

**Figure 1**. Alternative alignments of split read.

“cluster\_distance” controls how to cluster split reads and discordant read pairs. Based on GENCODE V29 gene annotation, 99% of genes are longer than 400Kbp (99% of protein-coding genes are longer than 617Kbp), so the parameter should be greater than 400000. A bigger value is acceptable, but it will impact the run time, and the standard deviation cutoff. Default value should be fine for most cases.

“overhang\_length” controls the alignment difference between STAR and TopHat2. Split read support breakpoint if it was aligned by TopHat2 within the value defined by this parameter (**Figure 2**).

Timeline

Description automatically generated with low confidence

**Figure 2**. The fusion filtration by BLAT.

“outside\_length\_in\_blat” and “inside\_length\_in\_blat” control the construction of artifact reference which was used by BLAT (**Figure 3**).

“max\_split\_read\_blat\_distance” controls the alignment difference between BLAT and STAR. Split read support breakpoint if it was aligned by BLAT within the value defined by this parameter (**Figure 3**).

Timeline

Description automatically generated

**Figure 3**. The fusion filtration by BLAT.

“length\_in\_sd” and “sd\_cutoff” filter false fusion transcripts based on read alignment around the fusion breakpoints. Artifacts can arise from repetitive regions of the genome. In these cases, there are many candidate fusion clusters supporting different fusions. True fusions often have large SD and artifacts usually have small SD. “length\_in\_sd” controls the window size to search for candidate fusion clusters. See 2.7/IV below for how SD is calculated.

* 1. Output. There are two output files named fusions.tsv.gz and fusions\_abridged.tsv.gz. The second one contains subset columns of the first file and is much smaller. The columns of fusions.tsv.gz are:

Column 1: Chromosome of breakpoint 1 (e.g., chr1)

Column 2: Position of breakpoint 1 (e.g., 46389)

Column 3: Strand of breakpoint 1 (e.g., +)

Column 4: Chromosome of breakpoint 2 (e.g., chr2)

Column 5: Position of breakpoint 2 (e.g., 16337)

Column 6: Strand of breakpoint 2 (e.g., -)

Column 7: Split read count reported by STAR (e.g., 1)

Column 8: Read pair count reported by STAR (e.g., 4)

Column 9: Split read count supported by TopHat2 (e.g., 1)

Column 10: Potential split read count not supported by TopHat2 (e.g., 0)

Column 11: Read pair count reported by TopHat2 (e.g., 3)

Column 12: Split read count supported by BLAT (e.g., 1)

Column 13: Split read count supported by BLAT (considering split read supported by TopHat2 only, e.g., 1)

Column 14: Minimal distance between read pair and breakpoint 1 after aligning by TopHat2 (e.g., 13)

Column 15: Minimal distance between read pair and breakpoint 2 after aligning by TopHat2 (e.g., 22)

Column 16: Sequence identity (e.g., 0.52)

Column 17: Minimal distance between split read and breakpoint 1 after aligning by BLAT (e.g., 0)

Column 18: Minimal distance between split read and breakpoint 2 after aligning by BLAT (e.g., 0)

Column 19: Minimal distance between split read and breakpoint 1 after aligning by BLAT (considering split read supported by TopHat2 only, e.g., 0)

Column 20: Minimal distance between split read and breakpoint 2 after aligning by BLAT (considering split read supported by TopHat2 only, e.g., 0)

Column 21: Presence of canonical splice site (Y or N)

Column 22: Cluster count around breakpoint 1 (e.g., 1)

Column 23: Cluster count around breakpoint 2 (e.g., 1)

Column 24: Cluster count around breakpoint 1 and breakpoint 2 (e.g., 1)

Column 25: Percentage of split reads and read pairs supported fusion transcript’s cluster around breakpoint 1 and breakpoint 2 (e.g., 0.75)

Column 26: Standard deviation for candidate fusion clusters around breakpoint 1 (e.g., NA, NA for only one cluster)

Column 27: Standard deviation for candidate fusion clusters around breakpoint 2 (e.g., NA, NA for only one cluster)

Column 28: Whether fusion transcript locates in the same gene (Y or N)

Column 29: Fusion type (non-coding or protein-coding)

Column 30: Overlapped gene in breakpoint 1 (e.g., C9/DAB2, genes were separated by “/”)

Column 31: Gene type in breakpoint 1 (non-coding\_gene, protein-coding\_gene or unknown, different genes’ type were separated by “/”)

Column 32: Gene strand in breakpoint 1 (e.g., +)

Column 33: Breakpoint 1’s location (intron, exon, splice\_site or unknown)

Column 34: Breakpoint 1’s region type (5’UTR, 3’UTR, CDS, non-coding or unknown)

Column 35: Exon frame after breakpoint 1 (0, 1, 2 or N, N means unknown)

Column 36: Overlapped gene in breakpoint 2 (e.g., EML4)

Column 37: Gene type in breakpoint 2 (non-coding\_gene, protein-coding\_gene or unknown, different genes’ type were separated by “/”)

Column 38: Gene strand in breakpoint 2 (e.g., -)

Column 39: Breakpoint 2’s location (intron, exon, splice\_site or unknown)

Column 40: Breakpoint 2’s region type (5’UTR, 3’UTR, CDS, non-coding or unknown)

Column 41: Exon frame before breakpoint 2 (0, 1, 2 or N, N means unknown)

Column 42: Fusion frame (in-frame, out-frame, or unknown)

Column 43: Split reads reported by STAR

Column 44: Read pairs reported by STAR

Column 45: Split reads supported by TopHat2 (e.g., read\_23)

Column 46: Potential split reads not supported by TopHat2 (e.g., NA)

Column 47: Read pairs supported by TopHat2 (e.g., read\_38,read\_70,read\_8)

Column 48: Split reads supported by BLAT (e.g., read\_23)

Column 49: Distance between each read pair and breakpoint 1 after aligning by TopHat2 (e.g., 740,23,13)

Column 50: Distance between each read pair and breakpoint 2 after aligning by TopHat2 (e.g., 292,2592,22)

Column 51: Distance between each split read and breakpoint 1 after aligning by BLAT (e.g., 0)

Column 52: Distance between each split read and breakpoint 2 after aligning by BLAT (e.g., 0)

Column 53: Distance between each split read and breakpoint 1 after aligning by BLAT (considering split read supported by TopHat2 only, e.g., 0)

Column 54: Distance between each split read and breakpoint 2 after aligning by BLAT (considering split read supported by TopHat2 only, e.g., 0)

Column 55: Needleman-Wunsch alignment of sequence used to calculate sequence identity in breakpoint 1 (e.g., --AGTGGGCCAGGTAG-GGCTGG)

Column 56: Needleman-Wunsch alignment of sequence used to calculate sequence identity in breakpoint 2 (e.g., CCACT—GCCAGG-AGAACCTCA)

Column 57: Discordant read pair cluster IDs around breakpoint 1 (e.g., 1)

Column 58: Discordant read pair cluster IDs around breakpoint 2 (e.g., 1)

Column 59: Supporting reads count in each cluster around breakpoint 1 (e.g., 4)

Column 60: Supporting reads count in each cluster around breakpoint 2 (e.g., 3)

1. **Details of run\_SFyNCS.sh**

run\_SFyNCS.sh is a bash script that integrates several scripts to identify split reads and read pairs and filter false fusion transcripts. Advanced users can customize it to optimize the pipeline. Below are the detailed steps in run\_SFyNCS.sh.

1. Step 1. Generate a file named Chimeric.out.junction that contains split reads and read pairs by running STAR (refer to section 3 in II for STAR parameters, and refer to <https://github.com/alexdobin/STAR/blob/master/doc/STARmanual.pdf> for more detail). Each file line contains the alignment information of a split read or a read pair. Following 14 columns of Chimeric.out.junction is needed:

Column 1: Chromosome of the donor

Column 2: First base of the intron of the donor (1-based)

Column 3: Strand of the donor

Column 4: Chromosome of the acceptor

Column 5: First base of the intron of the acceptor (1-based)

Column 6: Strand of the acceptor

Column 7: Breakpoint type: -1=encompassing breakpoint (between the mates), 1=GT/AG, 2=CT/AC, 0=any other motif

Column 8: Repeat length to the left of the breakpoint

Column 9: Repeat length to the right of the breakpoint

Column 10: Read name

Column 11: First base of the first segment (on the + strand)

Column 12: CIGAR of the first segment

Column 13: First base of the second segment

Column 14: CIGAR of the second segment

1. Step 2. Process STAR alignment information. This step includes formatting split reads and read pairs alignment, removing split reads and read pairs with multiple alignments, removing duplicates, and adjusting alignment for adjacent split reads. The scripts include:
2. format\_STAR\_chimeric\_file.pl

perl format\_STAR\_chimeric\_file.pl Chimeric.out.junction >format\_chimeric.tsv

1. remove\_multiple\_mapped\_reads.pl

perl remove\_multiple\_mapped\_reads.pl format\_chimeric.tsv >no\_multiple\_mapped.tsv

1. remove\_duplicate\_reads.pl

perl remove\_duplicate\_reads.pl no\_multiple\_mapped.tsv >temp\_no\_duplicate.tsv

Note: there would be two breakpoint lines if both read 1 and read 2 are spilt read and have the same breakpoint. Use following command to remove duplicates:

head -n 1 temp\_no\_duplicate.tsv >no\_duplicate.tsv

sort temp\_no\_duplicate.tsv | uniq | grep -v “Chr\_breakpoint\_1” >>no\_duplicate.tsv

1. adjust\_ajacent\_breakpoints.pl

perl adjust\_ajacent\_breakpoints.pl [options] no\_duplicate.tsv >adjust\_ajacent.tsv

-a --adjacent\_distance INT Breakpoints within this distance will be adjusted [default: 5]

1. Step 3 cluster\_discordant\_reads.pl. Cluster split reads and read pairs if breakpoints are aligned to the same chromosomes, the same strands and the distance of breakpoints are less than or equal to the defined distance (-w parameter).

Perl cluster\_discordant\_reads.pl [options] merge\_ajacent.tsv >cluster.tsv

-w --window\_size INT Split reads and read pairs within this distance will be clustered together [default: 1000000]

1. Step 4. Generate fusion transcript candidates.

perl identify\_fusion\_candidates\_from\_cluster\_reads.pl cluster.tsv >preliminary\_candidates.tsv

Note: fusion candidates are further filtered by the number of split reads, read pairs, total reads, fusion transcript related to mitochondria, and fusion distance:

awk -v min\_split\_reads=$min\_split\_reads -v min\_read\_pairs=$min\_read\_pairs -v min\_total\_reads=$min\_total\_reads ‘NR==1 || ($9>=min\_split\_reads && $10>=min\_read\_pairs && ($9+$10)>=min\_total\_reads)’ preliminary\_candidates.tsv | grep -v chrM >temp.tsv

mv temp.tsv preliminary\_candidates.tsv

awk -v deletion\_like\_distance =$ deletion\_like\_distance -v duplication\_like\_and\_inversion\_like\_distance=$duplication\_like\_and\_inversion\_like\_distance ‘NR==1{print $0;} $1!=$4{print $0;} $1==$4{if($3==”+” && $6==”-“){if(($5-$2)>= deletion\_like\_distance) print $0;} else{if(($5-$2)>=duplication\_like\_and\_inversion\_like\_distance ) print $0;}}’ preliminary\_candidates.tsv >temp.tsv

mv temp.tsv preliminary\_candidates.tsv

1. Step 5. Confirm split reads and discordant read pairs by TopHat2 processe\_by\_tophat.pl. The distance between read pairs and breakpoints also calculated.

Perl processe\_by\_tophat.pl [options] tophat.bam preliminary\_candidates.tsv >processed\_with\_tophat.tsv

-o --overhang\_length INT Breakpoint overhang length (1.3/IV) [default: 5]

-w --window\_size INT Maximal distance between TopHat2 alignment and breakpoint [default: 1000000]

-d --max\_read\_pair\_distance INT Maximal distance between the TopHat2 alignment of read pairs and breakpoint [default: 10000]

-p --min\_split\_reads INT Minimal number of split reads for a fusion transcript to be identified [default: 1]

-r --min\_read\_pairs INT Minimal number of read pairs for a fusion transcript to be identified [default: 1]

-t --min\_total\_reads INT Minimal number of total reads for a fusion transcript to be identified [default: 3]

Note: To speed up TopHat2 process, tophat.bam are generated by aligning split reads and read pairs in preliminary\_candidates.tsv. The script select\_fastq.pl is used to extract fastq of reads. The commands are (user must provide $thread\_number and $tophat\_index):

1. cut -f11,12 preliminary\_candidates.tsv | sed "s#\t#\n#;s#,#\n#g" | grep -vw "NA" | grep -vP "Split\_reads|Read\_pairs" | sort | uniq >selected\_discordant\_reads.tsv
2. perl select\_fastq.pl -s selected\_discordant\_reads.tsv 1.fastq\* 2.fastq\* >selected\_discordant\_reads\_1.fastq 2>selected\_discordant\_reads\_2.fastq

-s –-have\_space Read name have space. It can be set even there is no space in read name.

1. tophat --no-coverage-search \

--fusion-search \

--fusion-anchor-length 12 \

--fusion-min-dist 100000 \

--read-mismatches 4 \

--read-gap-length 4 \

--read-edit-dist 4 \

--splice-mismatches 2 \

--max-insertion-length 4 \

--max-deletion-length 4 \

--segment-mismatches 3 \

--fusion-read-mismatches 4 \

-o tophat\_output \

-p $thread\_number \

$tophat\_index \

selected\_discordant\_reads\_1.fastq selected\_discordant\_reads\_2.fastq

1. Step 6. Confirm split reads by BLAT processe\_by\_blat.pl. The fusion transcripts are filtered by canonical splice site motif, sequence identity, and the distance between split read and breakpoint.

perl processe\_by\_blat.pl [options] -f $genome\_fasta processed\_with\_tophat.tsv >processed\_with\_blat.tsv

-m --motif\_searching\_length INT Splice site motifs (GT in the donor, AAG/CAG/TAG in the acceptor) are searched within this window size of breakpoints [default: 5]

-f --fasta STR Reference genome fasta file (must be given)

-a --align\_percentage FLOAT Minimal percentage of bases of the whole read that is alignable by BLAT for split reads [default: 0.9]

-o --outside\_length\_fusion INT Specify the window of breakpoint flanking sequence for artificial reference [default: 1000000]

-i --inside\_length\_fusion INT Specify the window of breakpoint flanking sequence for artificial reference [default: 100]

-l --length\_for\_identity INT Flanking sequences size of both fusion breakpoints to calculate sequence identity [default: 10]

-p --min\_split\_reads INT Minimal number of split reads for a fusion transcript to be identified [default: 1]

-t --min\_total\_reads INT Minimal number of total reads for a fusion transcript to be identified [default: 3]

-d --max\_split\_read\_blat\_distance INT Maximal distance between BLAT alignment and breakpoints [default: NA, filter not applied]

-e --max\_sequence\_identity FLOAT Maximal sequence identity between flanking sequences of two fusion breakpoints [default: 1]

-c --filter\_by\_canonical\_splice\_motif STR Filter by canonical splice site motif [default: “Y”]

Note: selected\_discordant\_reads\_1.fastq and selected\_discordant\_reads\_2.fastq generated in step 5 must exists and user must provide $genome\_fasta. To speed up the process and reduce memory requirement, processed\_with\_tophat.tsv is split into 50 small pieces in run\_SFyNCS.sh.

1. Step 7. Calculate the standard deviation of the number of reads in clusters and the minimal distance between read pair and breakpoints. For each breakpoint’s upstream and downstream 100bp region, the number of clusters that are assigned in step 3 and the number of reads supporting each cluster is calculated, the standard deviation is then calculated as the formula below.

Standard deviation = , where and

N is the number of clusters, mi is the number of reads in clusteri, ni is the proportion of reads in clusteri.

The script of step 7 is cluster\_statistics.pl

perl cluster\_statistics.pl [options] processed\_with\_blat.tsv >fusion\_statistics.tsv

-f --flanking\_length INT Breakpoint flanking size to calculate standard deviation [default: 100]

-s --sd\_cutoff FLOAT Standard deviation cutoff to filter fusions [default: 0.1]

Note: A file named temp\_cluster.bed including cluster information must exist, and it is generated by the following command. To speed up the process and reduce memory requirement, processed\_with\_blat.tsv is split into 50 small pieces in run\_SFyNCS.sh.

The command used to generate temp\_cluster.bed:

awk 'BEGIN{FS=OFS="\t"} NR>1{print $1,$2-1,$2,$3,$7,$8,$9; print $4,$5-1,$5,$6,$7,$8,$9;}' cluster.tsv | sort -k1,1 -k2,2n | uniq >temp\_cluster.bed

1. Step 8. Filter fusions by gene annotation annotate\_fusions.pl.

perl annotate\_fusions.pl [options] $annotation\_file fusion\_statistics.tsv > fusions\_unfiltered\_by\_normal.tsv

-a --fasta STR Reference genome fasta file, must be given.

-f --filter\_in\_the\_same\_gene STR Whether filter fusions in the same gene [default: Y]

Note: user must provide $annotation\_file in section 2.5 in II. To speed up the process and reduce memory requirement, fusion\_statistics.tsv is split into 50 small pieces in run\_SFyNCS.sh.

1. Step 9. Filter by breakpoints from normal samples filter\_fusion\_by\_normal\_junctions.pl. Fusion transcript will be filtered if there are at least two fusion supporting reads (either discordant read pairs or split reads) within 10kb of both breakpoints in any normal samples.

perl filter\_fusion\_by\_normal\_junctions.pl [options] normal\_junctions.tsv fusions\_unfiltered\_by\_normal.tsv > fusions\_filtered\_by\_normal.tsv

-a --adjacent\_distance INT Window size to search for breakpoints in normal samples [default: 10000]

-t --threshould\_normal\_read\_count INT Filter fusions if numbers of reads (discordant pairs or split reads) in normal samples are equal to or more than the specified value [default: 2]

**V. Reference**

TBD

**VI. Contact**

If users have questions, please contact Xiaoming Zhong ([xiaomingzhong@uchicago.edu](mailto:xiaomingzhong@uchicago.edu)) and Lixing Yang ([lixingyang@uchicago.edu](mailto:lixingyang@uchicago.edu)).