SPRINT manual

Version 0.1.7

Feng Zhang

15110700005@fudan.edu.cn

Nov 21, 2017

Contents

1 Introduction

Sn**P**-free **R**na editing **I**de**N**tification **T**oolkit (SPRINT) was designed for identifying RNA editing sits (RESs) without the need to filter out SNPs. SPRINT also integrates the detection of hyper RESs from remapped reads, and has been fully automated to any RNA-seq data with reference genome sequence available. In general, **SPRINT** requires FASTQ files as input, and uses BWA as the aligner. Users can also apply **sprint_from_bam** to aligned reads (in BAM format).

2 Download and install

Download: For binary version, users can download **SPRINT** and **sprint_from_bam** from http://sprint.tianlab.cn/SPRINT/. For package version, users can turn to https://pypi.python.org/pypi/SPRINT or use the command options of "pip install sprint".

Operation environment: Linux, Python2.7.

Attention: We built the binary version of SPRINT in Centos 7 (x64), and tested it in Centos 7 (x64) and Ubuntu 16.04 (x64). If you encounter any binary version related problems please change to proper operating system or use package version of SPRINT. We developed and tested the package version of SPRINT with Python2.7. If you encounter any Python related problems please change to Python2.7. Please install SPRINT (python package) before using the script version of sprint_from_bam.

3 Input

Reference genome FASTA format.

See https://en.wikipedia.org/wiki/FASTA_format for details about FASTA.

Raw reads FASTQ format.

See https://en.wikipedia.org/wiki/FASTQ_format for details about FASTQ.

Before using SPRINT, users should trim the adapters and remove reads with low quality by using TrimGalore or other softwares.

TrimGalore: http://www.bioinformatics.babraham.ac.uk/projects/trim_galore/

Aligned reads BAM format.

See https://en.wikipedia.org/wiki/Binary_Alignment_Map for details about BAM.

Repeat annotation files see http://sprint.tianlab.cn/SPRINT/dbrep/

Gene annotation files GTF format.

See http://www.ensembl.org/info/website/upload/gff.html for details.

4 Dependencies

BWA http://bio-bwa.sourceforge.net

Samtools http://www.htslib.org

5 Output

PARAMETER.txt: The parameters used by users.

SPRINT_identified_regular.res: Hyper-RESs (A-to-I) identified by SPRINT.

SPRINT_identified_hyper.res: Regular-RESs (all kinds) identified by SPRINT.

SPRINT_identified_all.res: Regular- and hyper- RESs identified by SPRINT.

The format of **SPRINT_identified_regular.res, SPRINT_identified_hyper.res,** and **SPRINT_identified_all.res:**

Column	1	2	3	4	5	6	7
	Chromosome	Location (0base)	Location (1base)	Type	Supporting information	Strand	AD:DP

Each raw shows the information of each RES.

(1) **Chromosome** The name of chromosome.

(2, 3) **Location** The location of the RES in the chromosome.

(4) **Type** Type of the identified RES (e.g. AG means "A-to-G", etc.).

(5) **Supporting information** Number of supporting reads used by SPRINT.

(6) **Strand** Possible values are: ".", "+", and "-" which correspond to "uncertain", "positive strand", and "negative strand" respectively.

(7) **AD:DP** The number of reads that has this RES and the number of reads that cover this location.

6 Usage

6.1 Start from raw reads:

6.1.1 Prepare: Mask reference genome and build mapping index

Usage:

sprint prepare [options] reference_genome(.fa) bwa_path

options:

-t transcript_annotation(.gtf) #Optional

6.1.2 Main: Identify regular- and hyper- RESs

Attention:

Before using 'sprint main', please use 'sprint prepare' to build mapping index.

Usage:

sprint main [options] reference_genome(.fa) output_path bwa_path samtools_path

Options:

-1 read1(.fq) # Required !!!
-2 read2(.fq) # Optional
-rp repeat_file # Optional, you can download it from http://sprint.software/SPRINT/dbrep/
-ss INT # when input is strand-specific sequencing data, please clarify the direction of read1. [0 for antisense; 1 for sense] (default is 0)
-c INT # Remove the fist INT bp of each read (default is 0)
-p INT # Mapping CPU (default is 1)
-cd INT # The distance cutoff of SNV duplets (default is 200)
-csad1 INT # Regular - [-rp is required] cluster size - Alu - AD >=1 (default is 3)
-csad2 INT # Regular - [-rp is required] cluster size - Alu - AD >=2 (default is 2)
-csnar INT # Regular - [-rp is required] cluster size - nonAlu Repeat - AD >=1 (default is 5)
-csnr INT # Regular - [-rp is required] cluster size - nonRepeat - AD >=1 (default is 7)
-csrg INT # Regular - [without -rp] cluster size - AD >=1 (default is 5)
-csahp INT # Hyper - [-rp is required] cluster size - Alu - AD >=1 (default is 5)
-csnarhp INT # Hyper - [-rp is required] cluster size - nonAlu Repeat - AD >=1 (default is 5)
-csnrhp INT # Hyper - [-rp is required] cluster size - nonRepeat - AD >=1 (default is 5)

-cshp INT # Hyper - [without -rp] cluster size - AD >=1 (default is 5)

6.2 Start from aligned reads:

Attention:

1. Before using sprint_from_bam, BAM file should be sorted by using samtools, "samtools sort". Currently, in order to get the optimized results, we suggest that users should apply SPRINT to raw sequencing data (FASTQ format). See **6.1** for details. Please install SPRINT (python package) before using the script version of sprint_from_bam.

2. "sprint_from_bam" was designed for aligned reads without using any aligner. In exchange, "sprint_from_bam" doesn't include the function of detecting hyper RESs, because detecting hyper RESs needs to use aligner. However, in order to detect hyper RESs from BAM format, users can use SAMTOOLS to extract unaligned reads (BAM format) with command options of "samtools view -f4 -b", and then convert it into FASTQ format with command options of "samtools bam2fq". Finally, users can apply SPRINT (see **6.1** for details) to the unaligned reads (FASTQ format) to obtain hyper RESs and hyper-edited reads.

Usage:

sprint_from_bam [options] alinged_reads(.bam) reference_genome(.fa) output_path samtools_path

options:

-rp repeat_file # Optional, you can download it from http://sprint.software/SPRINT/dbrep/
-cd INT # The distance cutoff of SNV duplets (default is 200)
-csad1 INT # Regular - [-rp is required] cluster size - Alu - AD >=1 (default is 3)
-csad2 INT # Regular - [-rp is required] cluster size - Alu - AD >=2 (default is 2)
-csnar INT # Regular - [-rp is required] cluster size - nonAlu Repeat - AD >=1 (default is 5)
-csnr INT # Regular - [-rp is required] cluster size - nonRepeat - AD >=1 (default is 7)
-csrg INT # Regular - [without -rp] cluster size - AD >=1 (default is 5)

7 Tips

Download Scripts: http://sprint.tianlab.cn/SPRINT/tips

7.1 Edited reads can be extracted from "tmp/all_combined.zz"

all_combined.zz :

| Chr | SAM_Flag | MapQ | Loc | SNV | BaseQ | Read-loc | Seq | Read-name | Fragment-loc |

Users can use zz2sam.py to convert 'tmp/all_combined.zz' into BAM format (Download: zz2sam.zip, python2.7):

Step 1: Please move to the output-directory of SPRINT, and download zz2sam.zip;

Step 2: "unzip zz2sam.zip";

Step 3: "python zz2sam.py tmp/all_combined.zz";

Step 4: "samtools view -H tmp/genome/all.bam > SAMheader.txt";

Step 5: "cat SAMheader.txt tmp/all_combined.zz.sam > all_combined.zz.sam.header";

Step 6: "samtools view -bS all_combined.zz.sam.header > all_combined.zz.bam";

Step 7: "samtools sort all_combined.zz.bam -f all_combined.zz.sorted.bam".

7.2 Get A-to-I RESs from the ouput directory of SPRINT

Users can use getA2I.py to extract A-to-I RESs from the output of SPRINT (version>=0.1.7)

python | getA2I.py | 0 (1 for strand-specific data) | SPRINT_OUT | A_to_I_OUT

7.3 About "Supporting reads" and "AD"

For a given RES,

Supporting reads (regular-RES): mapped high-quality reads (MQ>=20 AND BASEQ >=25 AND fragment-loc >5);

Supporting reads (hyper-RES): remapped high-quality reads (BASEQ >=25 AND fragment-loc >5 AND Poly(N) <10 AND n(C)+n(T)<20);

AD (all RES): mapped reads (without the restriction of quality) + remapped reads (without the restriction of quality).

7.4 Change RepeatMasker File (rmsk) into BED file (used by SPRINT)

Step 1: Users can get RepeatMasker file from UCSC Table Browser (http://genome.ucsc.edu/cgi-bin/hgTables).

Step 2: python | rp2bed.py | hg38.rmsk | hg38_repeat.bed