# JACUSA manual
# Version 1.0

Michael Piechotta
michael.piechotta@age.mpg.de

June 18, 2016

## Contents

**1 Introduction** **1**

**2 Download** **2**
2.1 Installation and requirements . . . . . 2
2.2 Sample *in silico* data . . . . . 2

**3 Input** **3**
3.1 Alignment files . . . . . 3
3.1.1 Strand information . . . . . 3
3.2 Traverse BED-like file . . . . . 4

**4 Output** **5**

**5 Usage** **6**
5.1 SAMtools like mpileup for two samples . . . . . 7

**6 Identification of RNA editing sites** **8**

**7 Used libraries** **8**

# 1 Introduction

JAVA framework for accurate SNV assessment (JACUSA) is a one-stop solution to detect single nucleotide variants (SNVs) from comparing matched sequencing samples. Robust identification has proven to be a daunting task due to artefacts specific for NGS-data and employed mapping strategies. We implement various feature filters that reduce the number of false positives. JACUSA employs a window-based approach to traverse provided BAM files featuring highly parallel processing. JACUSA has been extensively evaluated and optimized to identify RNA editing sites in RNA-DNA and RNA-RNA sequencing samples. JACUSA requires an operating JAVA environment and uses sorted and indexed BAM files as input.

# 2 Download

Download the latest version of JACUSA from here. Check the source in the repository on GiTHub.

Just a suggestion

Is this gone be an open repository?

## 2.1 Installation and requirements

JACUSA does not need any configuration but needs a correctly configured Java environment. We developed and tested JACUSA with Java v1.7. If you encounter any Java related problems please consider to change to Java v1.7.

## 2.2 Sample *in silico* data

Download sample data that we used for the development of JACUSA. You can choose between different setups and species where the later greatly influences the data size and running time to detect variants. The gDNA VS cDNA represents the typical data setup that is encountered in detection of RNA editing sites via comparing genomic and transcriptomic sequencing reads. In this setup, variants have been only imputed to the cDNA BAM file. The cDNA VS cDNA data setup can be interpreted as representing allele specific expression of single variants or differential RNA editing. In this setup, variants with pairwise different base frequency have been imputed into both cDNA BAM files. Additionally, to make the identification of variants more challenging SNPs with pairwise similar base frequencies have been included to both BAM files. This sites should not be identified as true positive sites. gDNA data has been simulate with art[1] and cDNA reads have been simulated with flux[2]. Read simulations have been restricted to the corresponding first chromosome of the respective species. Sample data is available for *C. elegans* ce10 and *Homo sapien* hg19. Each archive consists of:

**gDNA.bam, cDNA.bam** BAM files: gDNA.bam and cDNA OR cDNA_1.bam and cDNA_2.bam

**snps.txt** Only available for cDNA VS cDNA. Coordinates of imputed SNPs. In both BAM files matching SNPs have the same target frequency but different effective or sampled frequency. The shape parameter determines how much the sampled frequency will deviate from the target frequency in each BAM file. The suffixes: _cdna_1 and _cdna_2 correspond to the respective BAM file

**variants.txt** Coordinates of imputed variants and their target and sample frequencies

Available sample data organized by data type and species:

- hg19_chr1_gDNA_VS_cDNA.tar.gz
- hg19_chr1_cDNA_VS_cDNA.tar.gz

This data has to be generated

[1] art
[2] flux simulator

- ce10_chrI_gDNA_VS_cDNA.tar.gz
- ce10_chrI_cDNA_VS_cDNA.tar.gz

# 3 Input

## 3.1 Alignment files

JACUSA needs sorted and indexed BAM files. BAM is a standardized file format for efficient storage of alignments. Check the manuals for [3] and/or [4] for how to use the respective tool to convert your alignment files to valid JACUSA input BAM.

In the following, commands for SAMtools are presented:

**SAM** $\rightarrow$ *BAM* `samtools view -Sb mapping.sam > mapping.bam`

**sort BAM** `samtools sort mapping.bam mapping.sorted`

**index BAM** `samtools index mapping.sorted.bam`

It is recommended pre-processing step to remove duplicate reads when identifying variants. Duplicated reads occur mostly due to PCR-artefacts. They are likely to harbour false variants and most statistical test require that reads are sampled independently. In the following, commands for picarf tools are presented:

```
java -jar MarkDuplicates.jar \
 I=mapping.sorted.bam O=dedup_mapping.sorted.bam \
 M=duplication.info
```

Invoke JACUSA with the additional command line option "-F 1024" to filter read that have been marked as duplicates.

### 3.1.1 Strand information

Depending on the employed sequencing library, JACUSA can use the strand orientation to build pileups. With the command line parameter "-P,–build-pileup <BUILD-PILEUP>" the user can choose from combinations of:

**S** stranded, and

**U** unstranded

to define if strand information will be utilized to build pileups. The format of <BUILD-PILEUP> is:

$$x, y : x, y \in \{S, U\}$$

where x corresponds corresponds to the first sample (BAM1_1-n files) and y to the second sample (BAM2_1-m files). The default is to ignore strand information for both samples ("-P U,U") JACUSA will process unstraded single-end and paired-end but strand orientation of base counts needs to be inferred from annotation (GFF). For stranded single-end RNA-Seq distiguish after which strand synthesis the read is sequenced:

[3] SAMtools/BCFtools
[4] picard tools

- first-strand or
- second-strand

and invert JACUSA output when first-strand library type is used. In order to utilize read orientation in paired end RNA-Seq both fragments must have same orientation f-f or r-r. Otherwise, use unstranded and invert correct orientation from annotation (see 1 for summary). In order to identify RNA editing sites by

Table 1: Summary of supported library types by JACUSA

| | **single-end** | **paired end** |
|---|---|---|
| **unstranded** | infer correct orientation from annotation | |
| **stranded** | | |
| first-strand | invert the orienation in JACUSA output | invert the orienation in JACUSA output (both fragments *must* have same orientation f-f or r-r) |
| second-strand | JACUSA output has correct orientation | JACUSA output has correct orientation (both fragments *must* have same orientation f-f or r-r) |

comparing gDNA and *stranded* RNA-Seq use "-P U,S"

## 3.2 Traverse BED-like file

Variant detection can be limited to specific regions of the genome or transcriptome. Provide a minimalistic BED-like file to restrict the search to this region(s) or site(s). Remaining region(s) of the BAM files will not be considered.

In the following traverse file, the search is confined to a 100nt region on contig 1 staring at 1,000 and a single site on contig 2 at coordinates 10,000: HINT: Many individual sites will slow down JACUSA. If possible, try to merge nearby sites into contiguous regions and extract specific sites from JACUSA output with bedtools[5] intersect:

**merge sites**

```
bedtools merge -d 500 singular_sites.bed > \
  contigous_regions.bed
```

**run JACUSA**

```
java -jar JACUSA.jar call-2 -b contigous_regions.bed -o
JACUSA.out mapping_1.sorted.bam mapping_2.sorted.bam
```

[5] bedtools

Table 2: Example of BED-like traverse file

| **contig** | **start** | **end** |
|---|---|---|
| 1 | 1000 | 1100 |
| 2 | 10000 | 10000 |

**extract sites**

```
bedtools intersect -wa -a JACUSA.out -b singular_sites.bed
```

# 4 Output

JACUSA writes its output to a user defined file or a pipe. When using multiple threads, JACUSA will create a gzipped temporary file for each allocated thread. Chosen command line parameters and current genomic position are printed to the command prompt. Furthermore, depending on the provided command line parameters, JACUSA will generate a file with sites that have been identified as potential artefacts when "-s" is provided. Currently, JACUSA supports the following output formats, controlled by "-f":

- Default (JACUSA output)
- Variant Call Format (VCF)[6]

The default output format is based on BED6[7] with additional JACUSA specific columns. The actual number of columns depends on number of provided BAM files.

Table 3: JACUSA default output format

| Column: | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| | 1 | 100 | 101 | variant | 8.07... | - | 0,0,0,6 | 0,6,0,0 | * | * |
| | | | | ... | | | | ... | | |

**(1, 2, 3) contig + start + end** 0-based, genomic coordinates of potential variant site

**(4) name** Currently, constant string: "variant". This dummy field is to ensure BED6 compatibility

**(5) score** Test-statistic $z \in \mathbb{R}$ that indicates the likelihood that this is a true variant. Higher number indicates a higher likelihood for a variant

**(6) strand** Possible values are: ".", "+", and "-" which correspond to "unstranded", "positive strand", and "negative strand" respectively. If strand is != ".", then the following base columns will be indicating base counts according to the strand - inverted base count if on the "negative strand"

**(7,8) basesIJ** The number of base columns depends on the number of BAM files. In basesIJ: $I$ corresponds to sample and $J$ to the respective replicate. Numbers indicate the base count of the following base vector: $(A, C, G, T)$

**(9) info** Additional info for this specific site. Currently, details about the parameter estimation of the Dirichlet-Multinomial can be shown. If nothing provided, the empty field is equal to "*"

[6] VCF file format
[7] BED file format

**(10) filter_info** Relevant, if feature filter(s) *X* have been provided with "-a X" on the command line. The column will contain a comma-separated list of feature filters that predict this site to be a potential artefact. Possible values are:

| Value | Description of potential artefact |
|---|---|
| D | Variant call in the vicinity of Read Start/End, Intron, and/or INDEL position |
| B | Variant call in the vicinity of Read Start/End |
| I | Variant call in the vicinity of INDEL position |
| S | Variant call in the vicinity of Splice Site |
| Y | Variant call in the vicinity of homopolymer |
| M | Max allowed alleles exceeded |
| H | "Control" sample contains non-homozygous pileup |
| d | Some pileup exceeds max depth |

# 5 Usage

Calling JACUSA without any arguments will print the available tools which currently are:

```
java -jarJACUSA.jar
call-2  Call variants - two samples
  pileup  SAMtools like mpileup for two samples

usage: jacusa.jar [OPTIONS] BAM1_1[,BAM1_2,BAM1_3,...] BAM2_1[,BAM2_2,BAM2_3,...]
```

| | | |
|---|---|---|
| -a,–pileup-filter FILTER> | <PILEUP- | chain of PILEUP-FILTER to apply to pileups:<br><br>D \| Filter distance to Read Start/End, Intron, and INDEL position. Default: 5:0.5 (D:distance:min_ratio)<br>S \| Filter distance to Splice Site. Default: 6:0.5 (S:distance:min_ratio)<br>B \| Filter distance to Read Start/End. Default: 6:0.5 (F:distance:min_ratio)<br>M \| Max allowed alleles per parallel pileup. Default: 2<br>Y \| Filter wrong variant calls in the vicinity of homopolymers. Default: 7 (Y:length)<br>H \| Filter non-homozygous pileup/BAM (1 or 2). Default: none<br>I \| Filter distance to INDEL position. Default: 6:0.5 (I:distance:min_ratio)<br>Separate multiple PILEUP-FILTER with ',' (e.g.: D,I) |
| -b,–bed <BED> | | BED file to scan for variants |
| -C,–base-config CONFIG> | <BASE- | Choose what bases should be considered for variant calling: TC or AG or ACGT or AT.... Default: ACGT |
| -c,–min-coverage COVERAGE> | <MIN- | filter positions with coverage < MIN-COVERAGE. Default: 5 |

| | |
|---|---|
| -c1,–min-coverage1 <MIN-COVERAGE1> | filter 1 positions with coverage < MIN-COVERAGE1. Default: 5 |
| -c2,–min-coverage2 <MIN-COVERAGE2> | filter 2 positions with coverage < MIN-COVERAGE2. Default: 5 |
| -d,–max-depth <MAX-DEPTH> | max per-BAM depth. Default: -1 |
| -d1,–max-depth1 <MAX-DEPTH1> | max per-sample 1 depth. Default: -1 |
| -d2,–max-depth2 <MAX-DEPTH2> | max per-sample 2 depth. Default: -1 |
| -F,–filter-flags <FILTER-FLAGS> | filter reads with flags FILTER-FLAGS. Default: 0 |
| -f,–output-format <OUTPUT-FORMAT> | Choose output format:<br><*> D: Default JACUSA<br>< > V: VCF |
| –filterNH_1 <NH-VALUE> | Max NH-VALUE for SAM tag NH |
| –filterNH_2 <NH-VALUE> | Max NH-VALUE for SAM tag NH |
| –filterNM_1 <NM-VALUE> | Max NM-VALUE for SAM tag NM |
| –filterNM_2 <NM-VALUE> | Max NM-VALUE for SAM tag NM |
| -h,–help | Print usage information |
| -m,–min-mapq <MIN-MAPQ> | filter positions with MAPQ < MIN-MAPQ. Default: 20 |
| -m1,–min-mapq1 <MIN-MAPQ1> | filter 1 positions with MAPQ < MIN-MAPQ1. Default: 20 |
| -m2,–min-mapq2 <MIN-MAPQ2> | filter 2 positions with MAPQ < MIN-MAPQ2. Default: 20 |
| -P,–build-pileup <BUILD-PILEUP> | Choose how parallel pileups are build: strand specific (S) or strand unspecific (U) default: U,U |
| -p,–threads <THREADS> | use # THREADS. Default: 1 |
| -q,–min-basq <MIN-BASQ> | filter positions with base quality < MIN-BASQ. Default: 20 |
| -q1,–min-basq1 <MIN-BASQ1> | filter 1 positions with base quality < MIN-BASQ1. Default: 20 |
| -q2,–min-basq2 <MIN-BASQ2> | filter 2 positions with base quality < MIN-BASQ2. Default: 20 |
| -r,–result-file <RESULT-FILE> | results are written to RESULT-FILE or STDOUT if empty |
| -s,–separate | Put feature-filtered results in to a separate file (= RESULT-FILE.filtered) |
| -T,–threshold <THRESHOLD> | Filter positions depending on test-statistic THRESHOLD default: DO NOT FILTER |
| -v,–version | Print version information. |
| -W,–thread-window-size <THREAD-WINDOW-SIZE> | size of the window used per thread. Default: 100000 |
| -w,–window-size <WINDOW-SIZE> | size of the window used for caching. Make sure this is greater than the read size and smaller than THREAD-WINDOW-SIZE>. Default: 10000 |

## 5.1 SAMtools like mpileup for two samples

See “Call variant - two samples” for details.

# 6 Identification of RNA editing sites

Use the following command line to identify RNA-DNA differences in BAM files that might give rise to RNA editing sites:

```
java -jar call-2 -o JACUSA.out -s -a H:1 gDNA.bam cDNA.bam
```

Option "-a H:1" ensures that potential polymorphisms in gDNA will be eliminated as artefacts. The number $x \in \{1, 2\}$ determines which sample has to be homomorph - in this case: gDNA.bam.

Use the following command line to identify RNA-DNA differences:

```
java -jar call-2 -o JACUSA.out -s cDNA1.bam cDNA2.bam
```

WARNING: If you want to identify RNA-RNA differences make sure NOT to use the filter "-a H:x"! Otherwise, potential valid variants will be filtered out.

# 7 Used libraries

picard tools — complete list