**VARIANT-CALLING**

This document describes the implementation of a variant calling pipeline for the analysis of next generation sequencing data, primarily for cfDNA data obtained from liquid biopsies. The implementation of this tool was created by Francesc Muyas Remolar while doing his PhD in the group of computational genomics of Prof. Stephan Ossowski.

**Note:** for running this variant calling pipeline, the input bam file needs to be deduplicated and needs to contain an extra column (DP:i:x). Otherwise, this pipeline will return empty tsv and vcf files. For receiving such bam file, Francesc Muyas script for deduplicating bam files (Deduplication.sh, path: =/mnt/SRV018/users/ahwernj1/scripts/Deduplication.sh) needs to be run with the respective bam file.

|  |  |
| --- | --- |
| **Required packages R** | **Required packages Python** |
| argparse  VGAM  Bbmle  data.table  ggplot2  survcomp (not available for R version 3.4.4 via CRAN, use Bioconductor instead) | Pybedtools  Argparse  Time  scipy.stats  numpy  pysam  sys |

**Version:** R 3.4.4, Python 2.7.15+

**Example Commands:**

Main script (Step1\_MRD.sh):

VCALLING=/mnt/SRV018/users/ahmuyaf1/scripts/Somatic\_caller\_new.v2/Pipelines/Step1\_MRD.sh

bash $VCALLING -bam $BAM -b $BED\_ALL -r $REF -o $OUT\_folter/$OUT\_file -p $PARAMS\_MID -bq 20 -d 0 -ac 2 -ns 5 -str 0

Information about bash commands:<https://www.tldp.org/LDP/abs/html/special-chars.html>

**Reference genome:** GRCh37

$REF=/mnt/share/data/genomes/GRCh37.fa

**Input parameter**

Input parameter are parsed via argparse (file path to argparse needs to be adjusted in the script Step1\_MRD.sh).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | **Default** | **Required** | | **Information** |
| -bam  -nbam  -b  -r  -o  -p  -mq  -bq  -d  -ac  -ns  -str | --  ‘ ‘  ‘ ‘  --  $PWD/Sample.vcf  ‘ ‘  30  20  5  3  1  0 | True  False  True  True  False  True  False  False  False  False  False  False | | Tumor bam file  Normal bam file  Bed file of the targeted regions, 0-based  Reference genome – fasta  Out vcf file  Beta binomial parameters table  Minimum mapping quality  Minimum base quality  Minimum distance allowed between variants  Minimum number of reads supporting a variant  Number of sites to be analyzed  Strand filter activation. 0 for deactivating, 1 for activating |
|  |  | |  |  |

**Procedure:**

The script Step1\_MRD.sh includes split\_bam\_step1.py, pileup2tsv\_08\_01\_19.py, Parameters\_step1.r, Combine\_functions\_step1.r and TSV2VCF.py scripts. This pipeline consists of the following 5 steps.

**STEP 1**

First, bam files are splitted by barcodes and the amount of duplicates. Bam files are indexed in order to identify alignments overlapping same regions in the genome. For IGV, it is helpful in order to display alignments in each genomic region

Use of script: split\_bam\_step1.py

|  |  |
| --- | --- |
| **Input** | **Output** |
| bam file | Indexed bam files  .txt  \_DP1.bam  \_DP2.bam  \_DP3.bam  \_DP4.bam |

**STEP 2**

Alignment BAM files are converted into pileup text files by mpileup from samtools (for every file in temp directory with DP and .bam in their file name).

Use of script: pileup2tsv\_08\_01\_19.py

|  |  |
| --- | --- |
| **Input** | **Output** |
| bed file  (of the targeted regions, 0-based) | pileup text files (Deduplicated.tsv files) |

|  |  |
| --- | --- |
| **Parameters as stated in the script, with default values** | |
| -d (maximum read depth)  -f (reference)  -l (positions, list of regions where pileup should be generated)  - q (minimum mapping quality to be used)  -Q (minimum base quality for base to be considered)  -x (ignore overlaps, disable read-pair overlap detection) | 9999999  $REF  $BED  $MQ  1 |

Each line consists of a pileup of reads at one single genomic position = base-pair information at each chromosomal position (<http://samtools.sourceforge.net/pileup.shtml>) 🡪 facilitates SNP calling and alignment viewing by eyes.

Mpileup commands: <http://www.htslib.org/doc/samtools-mpileup.1.html>

**STEP 3**

Parameters.txt file is generated from the deduplicated tsv files.

Use of script: Parameters\_step1.r

|  |  |
| --- | --- |
| **Input** | **Output** |
| Deduplicated\_DP\*.tsv files | Parameters.txt |

|  |  |
| --- | --- |
| **Parameters as stated in the script, with default values** | |
| -t1  -t2  -t3  -t4 | $TEMP/DEDUPLICATED\_DP1.tsv  $TEMP/DEDUPLICATED\_DP2.tsv  $TEMP/DEDUPLICATED\_DP3.tsv  $TEMP/DEDUPLICATED\_DP4.tsv |

Attention: option settings are options(warn=-1), meaning that all warnings generated by this file are ignored. Note: show warnings to test program.

**STEP 4**

Use of script: Combine\_functions\_step1.r

|  |  |  |
| --- | --- | --- |
| **Input** | | **Output** |
| Deduplicated.tsv files  Parameters.txt file | | Stats.tsv  (in Temp folder) |
| **Parameters as stated in the script, with default values** | | | |
| -t1  -t2  -t3  -t4  -params  -num (total number of sites analyzed, important for p-value correction | $TEMP/DEDUPLICATED\_DP1.tsv  $TEMP/DEDUPLICATED\_DP2.tsv  $TEMP/DEDUPLICATED\_DP3.tsv  $TEMP/DEDUPLICATED\_DP4.tsv  $TEMP/Parameters.txt  $NUM\_SITES | | |

**STEP 5**

Use of script: TSV2VCF.py

|  |  |
| --- | --- |
| **Input** | **Output** |
| Stats.tsv | final output file, vcf (containing variant calls), file with MRD values |

|  |  |
| --- | --- |
| **Parameters as stated in the script, with default values** | |
| -tID  -ref (reference genome)  -o (output file)  -cov (minimum coverage)  -ac (minimum reads supporting alternative allele, default 3)  -strand (strand bias test (Fisher test), 0 for turning it off, default 1)  -variant\_dist (minimum distance allowed between variants, default 20)  -tmpdir (folder for temporary files) | $IDt  $REF  $OUT\_FILE  10  $AC  $STRAND  $DIST  $TEMP |

**Error messages**

(mpileup) Max depth is above 1M. Potential memory hog!