COBASI
V5.1

USER GUIDE

DEVELOPED BY
LAURA GOMEZ-ROMERO

JANUARY 2018

CONTENT

INTRODUCTION **3**

PREREQUISITES **6**

REQUIRED FILES **6**

USE DESCRIPTION: GET CS REGIONS

- **GET CS REGIONS** .....

USE DESCRIPTION: ONE INDIVIDUAL FRAMEWORK:

- **OBTAIN VL** **8**
- **OBTAIN VSR'S** **10**
- **GET SIGNATURE READS** **13**
- **OBTAIN SNV'S LIST** **15**

USE DESCRIPTION: FAMILY-BASED FRAMEWORK (TRIO FRAMEWORK):

- **OBTAIN SIGNATURE READS** **18**
- **OBTAIN SNV'S LIST** **21**
- **OBTAIN DE NOVO SNV'S LIST** **24**

ACRONIMS **27**

REFERENCES **28**

INTRODUCTION

The COBASI approach is a unique solution to the variant calling problem. This approach is used to generate a list of SNVs from raw whole genome sequencing data from one individual (One individual framework). Besides, it can be extended to be used in a familiy-based framework.

Get CS Regions

The VGH pipeline requires a list of all unique regions in the genome. These unique regions are defined as those regions in which every kmer inside that region (of a defined size) is a COIN-String (unique string). The input and output parameters are indicated in the next figure.

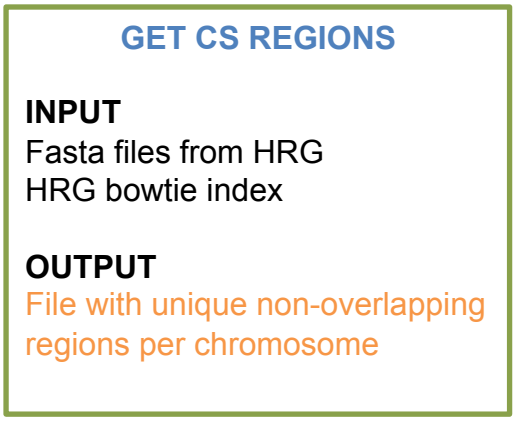

FIG 1. The input and output file for the stage: “Get CS Regions” are shown

One individual framework.

The VGH approach can be divided in four sequential stages. The four stages are illustrated in the next figure. For every stage the input and output parameters are indicated. It can be noted that the output files from one stage are used as input for the next one. Every stage is composed of several processes. In the first stage the Variation Landscape is computed (VL); in the second stage the Variation Siganture Regions are identified (VSR’s); in the third stage the Signature Reads are retrieved; and finallly, in the fourth stage a list of Single Nucleotide Variants (SNV’s) is generated.

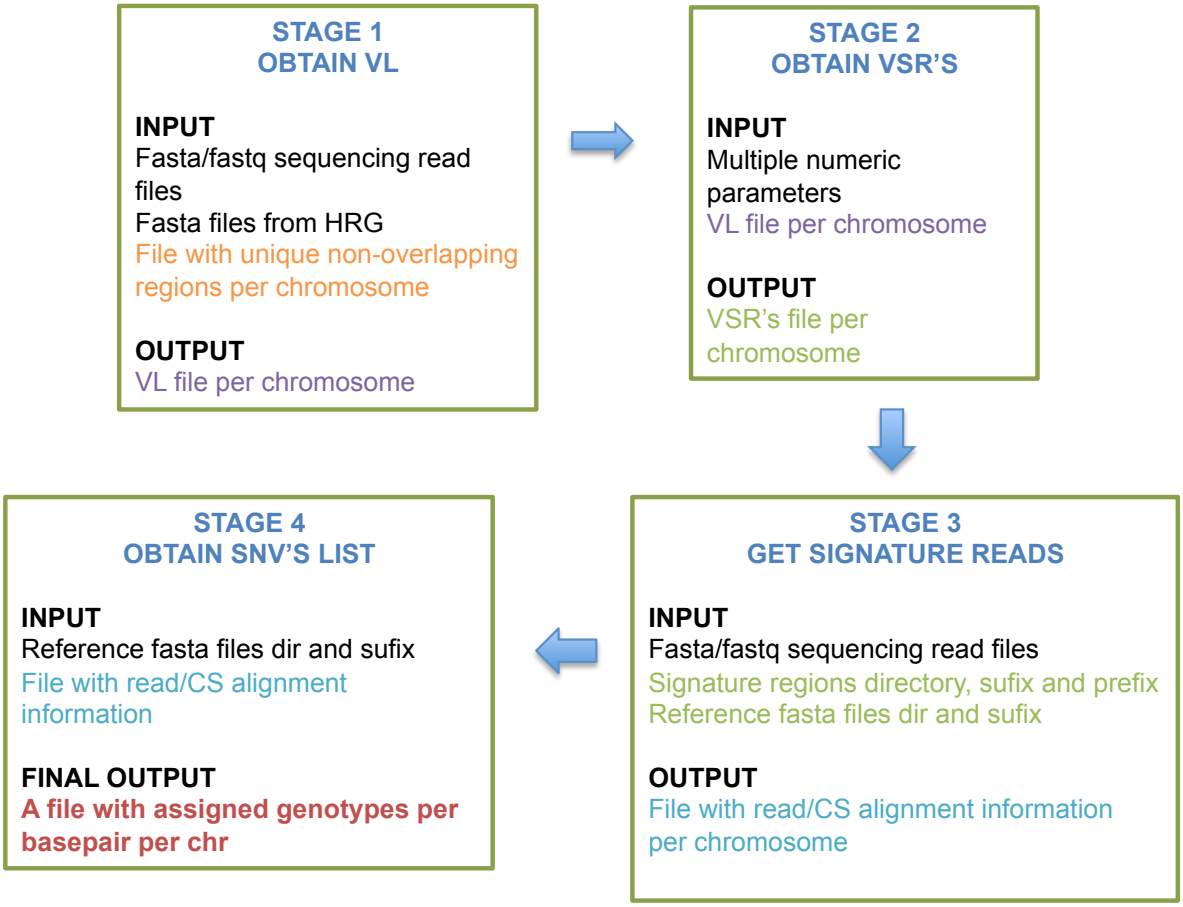

FIG 2. The whole one individual framework is shown.

Family-based framework

The family–based frameworw that is described in this manual is an extension of the one-individudal framework VGH approach. For the child all the processes from the one-individual framework must be completed. In the case of the parents only a list of VSR's must be generated (Stop at Stage 3 from One Individual Framework). It is important to note that different RCI thresholds must be set for each individual (See Stage 3- Obtain VSR's list, One-individual framework). The family-based framework uses as input such data.

In the Stage 1 the Signature Reads are obtained for each parent, in the Stage 2 a SNV is be generated for each parent. The processes described in Stage 1 and 2 must be completed for each parent independently. In the Stage 3, the data for the three individual of the family trio is analyzed and the *de novo* SNV are obtained. The whole pipeline is illustrated in Figure2.

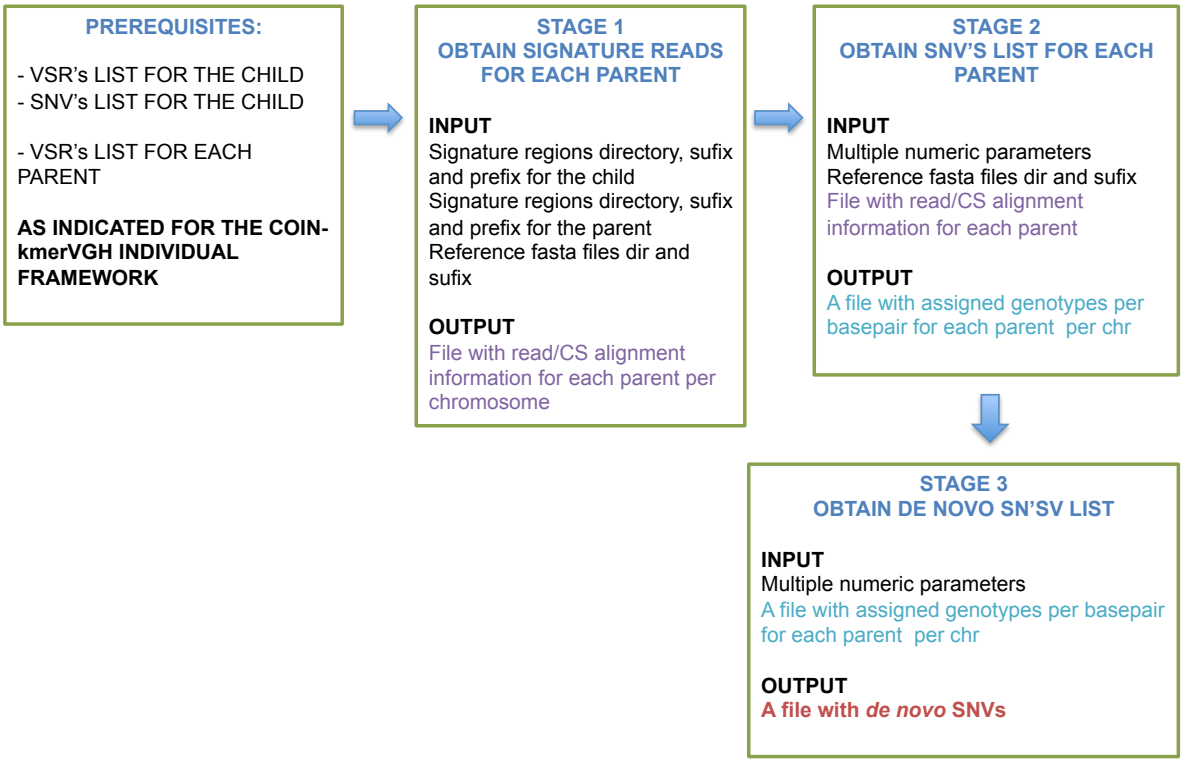

FIG 3. The family-based framework is illustrated.

PREREQUISITES

Software required. The pipeline was developed using the versions specified for each software.

- Bowtie [version supported 1.1]
- Jellyfish [version supported 1.1.6]
- AMOS [version supported 3.1.0]

  BOOST, Jellyfish and qt4 are required to install AMOS

  Execute these command lines for AMOS installaton:

  git clone git://amos.git.sourceforge.net/gitroot/amos/amos
  ./bootstrap
  ./configure --with-Boost-dir=/bin/BOOST/ --with-jellyfish=/bin/jellyfish/ --with-qmake-qt4=/bin/qt4/bin/qmake --prefix /bin/AMOS/
  make
  make install

- Python [version supported 2.7.2]

  Biopythhon is required

- Perl [version supported v5.14.2]

  Module Switch.pm is required

- c++ [version supported 4.4.6]

Hardware recommended for the analysis of human WGS (30-40X)

- 128GB RAM
- 12 cores

REQUIRED FILES

- Bowtie RG index
- Fasta files from RG
- Fasta/fastq sequencing reads files

USE DESCRIPTION: GET CS REGIONS

This process is required to be run only once for each RG. In this stage a Reference Genome (RG) COIN-String (CS) Regions database is generated. This is composed of three sequential steps.

FIG 4. The three sequential steps required in this stage are listed.

CUT REFERENCE GENOME

This script cuts one chromosome from the reference genome getting all kmers by sliding-windows of one base pair .

COMMAND LINE.

perl Cut_RG.pl -fna FASTA -k K –out OUT.STR

PARAMETERS.

- fna A fasta file with the sequence of one RG chromosome
- k kmer size (set to K=30)
- out Output file

OUTPUT.

A multi-fasta file with the sequence for every kmer along each chromosome of the RG
The ID for every multi-fasta sequence will correspond to the start position of each kmer

NOTES.

This process must be run per each chromosome to be analyzed
IMPORTANT: The ID in the fasta file should correpond to the file name:
chr1.fasta:
>chr1
NNNNNNNNNNNNNN

OBTAIN UNIQUE KMERS

This process gets all kmers from a chromosome found only once in the whole RG

COMMAND LINE.

bowtie -v V -m M –f INDEX FASTA OUTPUT &

PARAMETERS.

- v — Alignments (end-to-end) may have no more than V mismatches (SET to V=0)
- m — Suppress all alignments for a particular read or pair if more than <int> reportable alignments exist for it (SET to M=1)
- f — Query input files are (multi-)FASTA
- index — RG bowtie index (ebwt files path and prefix)
- fasta — Multi-fasta file with kmers from RG
- output — Output file [recommended extension .bowtie.out]

OUTPUT.

A bowtie output file per chromosome.
To see file explanation see Bowtie documentation
[http://bowtie-bio.sourceforge.net/index.shtml]

NOTES.

This process must be run per each chromosome.

OBTAIN NON-OVERLAPPING UNIQUE REGIONS

This script merges all positions from adjacent unique kmers (found by bowtie). It generates a list of unique non-overlapping regions.

COMMAND LINE.

perl Compute_CSDB_FromBowtie.pl –dir DIR_BOWTIE/ -sufix .BOWTIE.OUT –out DIR_OUT/

PARAMETERS.

- dir — Directory that contains the output from bowtie
- sufix — Bowtie output files sufix
- out — Output directory

OUTPUT.
One file per chr with a list of start and end positions of unique non-overlapping regions.
The ouput files will have the extension "_cs_uniq_regions.tab"

NOTES.
This script should be run only once

USE DESCRIPTION:
ONE INDIVIDUAL FRAMEWORK.

STAGE 1 – OBTAIN VL

In Stage 1 the Variation Landscape is computed (VL). This stage is composed of four sequential steps. However, process 2 (MERGE KMER COUNT) is only required if multiple databases are generated in process 1.

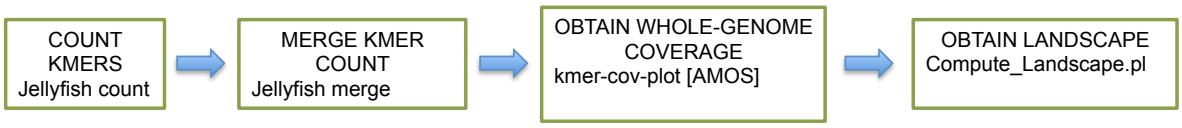

FIG 5. The processes from Stage 1 are listed.

COUNT KMERS

Jellyfish is a parallel-processing software that counts the ocurrences of all kmers (of k size) in a list of fasta or fastq files. Read the parameters used to understand how.

COMMAND LINE.

jellyfish count –o DB_OUT -m K -s S –t T --both-strands -L L [fastq1 fastq2]

PARAMETERS.

- Output — Output prefix
- m — Kmer size (SET to K=30)
- s — Hash size (RECOMMENDED S=10G)
- t — Number of threads (RECOMMENDED T=24 cores)
- both-strands — For any k-mer m, its canonical representation is m itself or its reverse-complement, whichever comes first lexicographically
- L — Don't output k-mer with count lower than L (SET to L=2)
- [files] — List of fasta or fastq files (sequencing reads files, separated by spaces)

OUTPUT.

One or more jellyfish databases storing the kmer counts for the sequencing reads.

NOTES.

Include all the fasta or fastq files from a single sequencing project in the same jellyfish command. For more documentation see:
http://www.cbcb.umd.edu/software/jellyfish/

MERGE KMER COUNT [OPTIONAL]

This command is required only if several jellyfish databases are generated in the previous step. It merges the count for every kmer in any of the listed databases.

COMMAND LINE.

jellyfish merge –o DB_MERGED [db_out_0 db_out_1]

PARAMETERS.

- o Output database
- databases List of jellyfish databases that will be merged

OUTPUT.

An unique merged jellyfish database per sequencing project

NOTES.

For more documentation see:
http://www.cbcb.umd.edu/software/jellyfish/

OBTAIN WHOLE-GENOME COVERAGE

kmer-cov-plot is a software included in the AMOS package. It returns the count (found in a kmer-count jellyfish database) for every kmer along a provided reference genome. It outputs the start position for every kmer with its count. This kmer count is taken as the read coverage for every kmer.

COMMAND LINE.

kmer-cov-plot --jellyfish –s DB < FASTA > OUTPUT.COVERAGE

PARAMETERS.

- jellyfish Use k-mer counts from a Jellyfish hash table.
- s Display only the combined count of the forward and reverse complement k-mers
- db Jellyfish database
- fasta RG fasta file
- output Output file

OUTPUT.
A file with the start position for every kmer along the fasta file sequence (RG chromosome) in column 1 and its count (retrieved from the jellyfish database) in column 2 (coverage files)

NOTES.
This process must be run per each chromosome.
For more documentation see:
[http://sourceforge.net/projects/amos/]

OBTAIN LANDSCAPE

This script filters out the positions for every non-unique kmers. It outputs a list of start positions for every unique kmer (CSs) asociated wit its count.

COMMAND LINE.
perl Compute_Landscape.pl -cov OUTPUT.COVERAGE -unique DIR_CSs/ -suf_unique SUFIX -out_dir DIR_OUT /

PARAMETERS.

○ cov	Coverage file
○ unique	The directory containing the files with the start and end positions of the non-overlapping unique region
○ suf_unqiue	Sufix of the unique regions files (SET to SUFIX =_cs_uniq_regions.tab)
○ out_dir	Output directory. The output files will be named {RG chromosome}{.land}

OUTPUT.
The Variation Landscape (VL) file per chromosome. It contains how many times each CS is found in the sequencing reads. The VL file contains two columns:

- column 1, RG position
- column2, the number of ocurrences per each CS along the reads.

NOTES.
This process must be run per each chromosome.

STAGE 2 – OBTAIN VSR's

In Stage 2 the Variation Signature Regions are identified (VSR's). Remember that every VSR can be composed of at most two drops and two rises at the start or at the end of the VSR, respectively. This stage is composed of two sequential steps.

NOTE1: The RCI threshold used for the VSR's discovery should be conservative (recommended 0.3)
NOTE2: To choose the different coverage thresholds an additional script has been developed. See Additional Scripts.

FIG 6. The processes from Stage 2 are listed.

GET ONE-DROP REGIONS (INTERNAL PART OF VSR'S)

This script obtain a list of partial VSR's. It only looks for the internal drop and rise in coverage characteristics of any VSR.

COMMAND LINE.

perl Get_Onedrop.pl -land FILE.LAND -max M -fac RCI -rmin R -fst FST -nt NT –density DEN –out FILE.ONEDROP

PARAMETERS.

- land — Input file. VL file [FILE.LAND]
- max — CS's with a coverage count higher than M are skipped in the VSR identification process
- fac — A Relative Coverage Index absolute value higher than RCI is considered as *bona fide* variation signal (SET RCI=0.3)
- rmin — The PrevCS and PostCS coverage must be at least R (SET R=10)
- fst — The median of the coverage values for InterCS's must be lower than FST (SET FST=third quartile coverage+10(IQR))
- nt — The length of the VSR must be longer than NT
- density — DEN is the mínimum density of CSs required inside the VSR
- out — Output file [FILE.ONEDROP]

OUTPUT.
The output file contains either whole VSRs or partial VSRs (only the internal drop and rise). In this step, the PrevCS and PostCS will be the ones before and after this (possibly incomplete) VSR signal. These regions will be extended, if possible, in the next process. This output file is composed of 13 columns:

1. PrevCS position
2. PrevCS count
3. Next CS after PrevCS position
4. Next CS after PrevCS count
5. Number of nucleotides in the internal region of the VSR
6. Number of interCSs (a VSR with a CS density of 1 will have the same value in columns 5 and 6)
7. Last CS before PostCS position
8. Last CS before PostCS count
9. PostCS position
10. PostCS count
11. RCI value for the VSR start
12. RCI value for the VSR end
13. Median of the coverage values for the InterCS's

NOTES.
This process must be run per each chromosome.

GET VARIATION SIGNATURE REGIONS

In the previous step possibly incomplete VSRs are detected. In this step, additional drops will be searched at the start of the previously identitfied partial VSRs and adittional rises will be looked at the end of such VSRs. Finally adjacent partial VSR's will be concatenated. Intermediate rises between the drops or intermediate drops between the rises are not allowed.

COMMAND LINE.

perl Get_SR.pl -land FILE.LAND –var FILE.ONEDROP -fac RCI -rmin R -max M -cs_size K –ratio RAT –sr_max SRM –out FILE.SR

PARAMETERS.

- land — Input file. VL file [FILA.LAND]
- var — File obtained in the Get One-Drop Regions process [FILE.ONEDROP]
- fac — A Relative Coverage Index absolute value higher than RCI is considered as *bona fide* variation signal (set to RCI=0.3)

- rmin — The PrevCS and PostCS coverage must be at least R (set to R=10)
- max — CS's with a count higher than M are skipped in the VSR process
- cs_size — An additional drop or rise are searched in the K nucleotides before and after the partial VSR that has been detected in the var file
- ratio — The ratio of the coverage between the PrevCS and PostCS should be < than RAT (recommended set RAT= 1.8)
- sr_max — The distance between PrevCS and PostCS should be < than SRM (recommenden set SRM=100000)
- out — Output file with final VSR information [FILE.SR]

OUTPUT.

The output file contains final VSRs information (VSR files). This output file is composed of 14 columns:

1. PrevCS position
2. PrevCS count
3. Next CS after PrevCS position
4. Next CS after PrevCS count
5. Number of nucleotides in the internal region of the VSR
6. Number of inter CSs (a VSR with a CS density of 1 will have the same value in columns 5 and 6)
7. Last CS before PostCS position
8. Last CS before PostCS count
9. PostCS position
10. PostCS count
11. RCI value for the VSR start
12. RCI value for the VSR end
13. Median of the coverage values for InterCS's
14. Flag indicating in which end a ladder is found (UP, DOWN, BOTH, NA)

NOTES.
This process must be run per each chromosome.

STAGE 3 - GET SIGNATURE READS

In Stage 3 the Signature Reads that contain either the PreCS or both SignatureCSs are retrieved. This stage is composed of three sequential steps.

FIG 7. The processes from Stage 3 are listed.

OBTAIN SIGNATURE CS'S SEQUENCE

This script will obtain the sequences for the PreCS and PostCS for every VSR. All the VSR file must be located on the same directory and they must have the same extension. The header of the multi-FASTA file that is generated will change for the CHILD and for the PARENTS in the family-based framework

COMMAND LINE.

python Cut_SignatureCSs.py --VSR=SR/ --REFDIR=REFDIR/ --sufixREF=sufixREF --sufixVSR=sufixVSR --prefixVSR=prefixVSR --kSIZE=kmer_size --output=SignatureCS.fa

python Cut_SignatureCSs.py -v SR/ -r REFDIR/ -x sufixREF -y sufixVSR -z prefixVSR -k kmer_size -o SignatureCS.fa

PARAMETERS.

- Script — Cut_SignatureCSs.py
- VSR, v — VSR directory path
- REFDIR , r — RG directory path
- sufixREF, x — Sufix of the fasta reference files
- sufixVSR , y — Sufix of VSR files
- prefixVSR , z — Prefix of VSR files (if the VSR files have no prefix, set to NA)
- kSIZE, k — CS size (SET to 30)
- output, o — Output file (multi-fasta file)

The name of the files must follow the next rules. Example:
RG chr1 file name: chr1.{sufixREF}
VSR chr1 file name: {prefixSR}chr1{sufixSR}

OUTPUT.
A multi-fasta file with the sequences of all Signature CSs

NOTES.
This step requires only one process

GET SIGNATURE READS

This script will obtain the sequences and some useful information from the reads that contain either the PreCS or both SignatureCSs. This script will process one FASTA or FASTQ file (with the read sequences) per run. The header of the input multi-FASTA is slightly different for the CHILD and for the PARENTS in the family-based framework

COMMAND LINE.

Retrieve_SignatureReads -c SignatureCS –t ReadType –f ReadFileX –k kmer_size –i Ind > outX

PARAMETERS.

- c [SignatureCS] A multi-fasta file with the sequences of all Signature CSs
- t [ReadType] FASTA or FASTQ formats are supported [FASTA|FASTQ]
- f [ReadFileX] A fasta or fastq file with the reads of the sequencing project
- k [kmer_size] CSs size = Jellyfish database kmer size
- i [Ind] Set to CHILD for SNV discovery in one individual
- outX Output file

OUTPUT.

The ouput file contains information about the alignment between the read and the respective Signature CS's (READALN-UNORDER files). This output file is composed of 10 columns:

1. RG chromosome
2. VSR PrevCS start position
3. VSR PostCS start position
4. SignatureCS aligned to the read (either the PrevCS or the PostCS)
5. CS sequence
6. Downstream read position of the alignment
7. Orientation of the aligment
8. Read ID
9. Read Sequence
10. Read Quality (NA, if the initial format file is FASTA)

NOTES.

This process must be run per each read file. Each process will produce one output file.

MERGE READS

If the sequencing experiments generates multiple FASTQ read files, the script that gets the Signature Reads will be run multiple times, and the hits for every SignatureCS will be distributed over multiple files. This script will concatenate such information.

COMMAND LINE.

perl Merge_Reads.pl -dir_read ALNDIR/ -sufix_read sufixREAD -dir_var SR/ -prefix_var prefixSR -sufix_var sufixSR -ind CHILD -dir_out OUTDIR/ -prefix_out GenomePrefix -sufix_out sufixREADALN &

PARAMETERS.

- dir_read — The directory where the READALN-UNORDER files were written
- sufix_read — Sufix of the READALN-UNORDER files
- dir_var — The directory where the VSR files were written.
- prefix_var — VSR file name must be: {prefixSR}chrName{sufixSR}. If there is no prefix set to NA
- sufix_var — Sufix of the VSR files. All the files from SR/ ending in sufixSR will be analyzed.
- ind — Set to CHILD for one individual SNV discovery
- dir_out — Output directory.
- prefix_out — Prefix for the output files. Set to NA if no prefix is desired.
- sufix_out — Sufix for the output files [READALN files]

OUTPUT.

One output file per chromosome.
Each output file contains ordered information about the alignment between the read and the respective Signature CS's (READALN files).
The information contained in these files is the same unordered information contained in the READALN-UNORDER files.

NOTES.

VSR file name must be: {prefixSR}chrName{sufixSR}.
This step requires only one process.
One output file is produced per each chromosome, the chromsome names are indicated in the names of the VSR files.
Output file name: {VSR file name}{ .reads}

STAGE 4 – OBTAIN SNV LIST

In Stage 4 a list of Single Nucleotide Variants (SNV's) is generated from information recorded from the Signature CSs.

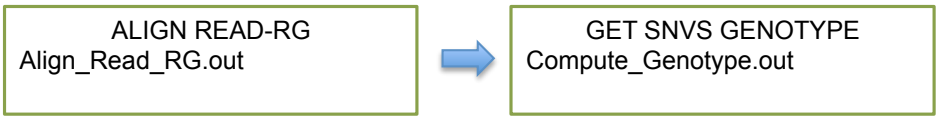

FIG 8. The processes from Stage 4 are listed.

ALIGN READ-REFERENCE GENOME

In this step every read is cut based on the positions recorded in the READALN files. This partial sequence read is aligned to the corresponding region in the RG. There are two different kind of reads, 1) the ones containing only the PrevCS from which only partial alignments of the VSR can be generated; and 2) the ones containing both Signature CSs from which global alignments (total aln) of the VSR can be generated . In this script reads with incongruency in the alignments with the Signature CS's are discarded, the PCR duplicates are eliminated, the total alignments are computed and variable regions are obtained. These variable regions must be in more than ReadRegion number of total alignments to be considered, only one allele per region is allowed, low complexity alignments from which multiple alignment positions can be obtained are discarded. . From these total alignments the minimum length of partial alignments is obtained. Partial alignments are computed. Alleles per base (reference

or alternative) are obtained, no gaps are allowed and only alleles found in ReadPerbase alignments are recorded.

COMMAND LINE.

Align_read_RG –a READALN-file –r REFDIR/ -s sufixREF –k k –t ReadRegion –p ReadPerbase –l LengthPartial –c ChildSNV –d RemoveDupFlag –i Ind –o out.total –q out.perbase

PARAMETERS.

- a [READALN-FILE] File per chromosome that contains information about the alignment between the read and the respective Signature CS's
- r [REFDIR] RG directory path
- s [sufixREF] Sufix of the fasta reference files
- k [k] CSs size = Jellyfish database kmer size
- t [ReadRegion] Each mismatch region between the reads and the RG must be supported by at least ReadRegion different reads
- p [ReadPerbase] Each mismatch nucleotide between the reads and the RG must be supported by at least ReadPerbase different reads
- l [LengthPartial] Length of extensión of partial alignment (set to 10)
- c [ChildSNV] Set to NA (Parameter used in family-framework)
- d [RemoveDupFlag] Set to TRUE to remove PCR duplicates (FALSE otherwise)
- i [Ind] For one individual SNV discovery set to CHILD
- o [out.total] Output file with regions of polymorphism data
- q [out.perbase] Output file with single nucleotide polymorphism data

OUTPUT.
A file with genotypes per region per chromosome was obtained with the following information:

1. RG chromosome
2. PrevCS start position
3. PostCS start position
4. Polymorphism region RG start
5. Polymorphism region RG end
6. RG nucleotide
7. Reads variant allele
8. Total alignments supporting variant allele/Number of total alignments

A file with genotypes per basepair per chromosome was obtained with the following information:

1. RG chromosome
2. PrevCS start position
3. PostCS start position
4. SNV RG position
5. RG nucleotide
6. Read alleles (allele1/allele2)
7. Total alignments supporting every allele (allele1/allele2/total)
8. Partial alignments supporting every allele (allele1/allele2/total)
9. Total number of alignments supporting every allele (allele1/allele2/total)

NOTES.
Nomenclature: RG chr1 file name: chr1{sufixREF}.
One process must be run per chromosome.

To detect the low complexity alignments: start and end positions for regions that contain consecutive polymorphisms were obtained, if multiple regions spanning the same nucleotide intervals were obtained, all the alignments for those low complexity regions were discarded.

To detect PCR duplicates: multiple reads for which the PrevCS was aligned to the same position were considered as PCR duplicates and one of them (randomly chosen) was assigned to be the representative read.

COMPUTE SNVs GENOTYPE

In this script, the likelihood for each genotype is computed and the most probable genotype (given the observed data) is reported as the final genotype. The probability for four possible genotypes is computed: Heterozygous reference (R/NR), Homozygous

reference (R/R), Heterozygous non-reference (NR1/NR2), Homozygous non-reference (NR/NR). For the child, this script does not print the homozygous reference-sites.

COMMAND LINE.

Compute_Genotype–p out.perbase –t FilePerbaseType –i IndividualType –u undetermined.out > genotype.out

PARAMETERS.

- p [out.perbase] File with single nucleotide polymorphism data
- t [FilePerbaseType] P if total number of alignments per allele is in column 9
- i [IndividualType] For individual SNV discovery set to CHILD
- u [undetermined.out] Output file with undetermined genotypes
- genotype.out File with genotype information per SNV

OUTPUT.

A file with assigned genotypes per basepair per chromosome with 11 columns:

- 1-9 will contain the information from the file out.perbase
- A numeric classificarion for the most probable genotype:
  - 1 - Heterozygous reference (Ref/NoRef)
  - 2 - Homozygous reference (Ref/Ref)
  - 3 - Heterozygous non-reference (NoRef1/NoRef2)
  - 5 - Homozygous non-reference (NoRef1/NoRef1)
- The different alleles for the assigned genotype

NOTES.

One process must be run per chromosome.

USE DESCRIPTION
FAMILY-BASED FRAMEWORK

STAGE 1 – OBTAIN SIGNATURE READS

In Stage 1 the Signature Reads for each parent are retrieved. This stage must be completed for each parent independently. In this stage only the regions that contain a SNV in the child are analyzed in the parents. For every SNV and its associated VSR in the child, the reads that contain the child Signature CS's for the associated VSR are retrieved from the parent fastq files.

FIG 9. The processes from Stage 1 are listed.

OBTAIN SIGNATURE CS'S SEQUENCE

This script will copy the file that contains the SignatureCSs identified in the child to the working directory for the analysis of every parent. The identifier of every Signature CSs will now reflect that is being used in the parent SNV discovery

COMMAND LINE.

python SignatureCS_Parent.py -c SignatureCSs_child.fa -o SignatureCSs_parent.fa

python SignatureCS_Parent.py --SignatureCSs SignatureCSs_child.fa
--OUTPUT SignatureCSs_parent.fa

PARAMETERS.

- Script — SignatureCS_Parent.py
- SignatureCSs, c — Multi-fasta file with all Signature CSs for the child
- OUTPUT, o — Output file (multi-fasta file), the same Child SignatureCS with IDs for the parents

OUTPUT.

A multi-fasta file with the sequences of all retrieved CSs sequence.
The identifier if every fasta sequence contains:

- CS ID for the sequence retrieved
- PrevCS for the child VSR
- Post CS for the child VSR

The last two positions will ve repeated twice

NOTES.
This step requires only one process

GET SIGNATURE READS

This script will obtain the sequences and some useful information from the reads that contain either the PreCS or both SignatureCSs. This script will process one FASTA or FASTQ file (with the read sequences) per run. The header of the input multi-FASTA is slightly different for the CHILD and for the PARENTS in the family-based framework. The software used at this stage is the same software used in One Individual – Stage3.

COMMAND LINE.

Retrieve_SignatureReads -c SignatureCS –t ReadType –f ReadFileX –k kmer_size –i Ind > outX

PARAMETERS.

- c [SignatureCS] A multi-fasta file with the sequences of all Signature CSs
- t [ReadType] FASTA or FASTQ formats are supported [FASTA|FASTQ]
- f [ReadFileX] A fasta or fastq file with the reads of the sequencing project
- k [kmer_size] CSs size = Jellyfish database kmer size
- i [Ind] Set to PARENT for SNV discovery in the parents
- outX Output file

OUTPUT.
The ouput file contains information about the alignment between the read and the respective Signature CS's (READALN-UNORDER files). This output file is composed of 12 columns:

1. RG chromosome
2. SignatureCS aligned to the read (either the PrevCS or the PostCS)
3. Parent VSR PrevCS start position
4. Parent VSR PostCS start position
5. Child VSR PrevCS start position
6. Child VSR PostCS start position
7. CS sequence
8. Downstream read position of the alignment
9. Orientation of the aligment

10. Read ID
11. Read Sequence
12. Read Quality (NA, if the initial format file is FASTA)

NOTES.
This process must be run per each read file
One output file is produced per each read file

MERGE READS

If the sequencing experiments generates multiple FASTQ read files, the script that gets the Signature Reads will be run multiple times, and the hits for every SignatureCS will be distributed over multiple files. This script will concatenate such information. The software used at this stage is the same software used in One Individual – Stage3.

COMMAND LINE.

perl Merge_Reads.pl -dir_read ALNDIR/ -sufix_read sufixREAD -dir_var SR/ -prefix_var prefixSR -sufix_var sufixSR -ind PARENT -dir_out OUTDIR/ -prefix_out GenomePrefix -sufix_out sufixREADALN &

PARAMETERS.

- dir_read — The directory where the READALN-UNORDER files were written
- sufix_read — Sufix of the READALN-UNORDER files
- dir_var — The directory where the VSR files were written.
- prefix_var — VSR file name must be: {prefixSR}chrName{sufixSR}. If there is no prefix set to NA
- sufix_var — Sufix of the VSR files. All the files from SR/ ending in sufixSR will be analyzed.
- ind — Set to PARENT for SNV discovery in the parents
- dir_out — Output directory.
- prefix_out — Prefix for the output files. Set to NA if no prefix is desired.
- sufix_out — Sufix for the output files [READALN files]

OUTPUT.
One output file per chromosome.
Each output file contains ordered information about the alignment between the read and the respective Signature CS's (READALN files).
The information contained in these files is the same unordered information contained in the READALN-UNORDER files.

NOTES.
VSR file name must be: {prefixSR}chrName{sufixSR}.
This step requires only one process.

One output file is produced per each chromosome, the chromsome names are indicated in the names of the VSR files.
Output file name: {VSR file name}{ .reads}

STAGE 2 – OBTAIN SNV LIST

In Stage 2 a list of Single Nucleotide Variants (SNV's) is generated for each parent. This Stage is almost identical to Stage 5 in the One-Indivual framework, the only different parameter is the information contained in the input files.

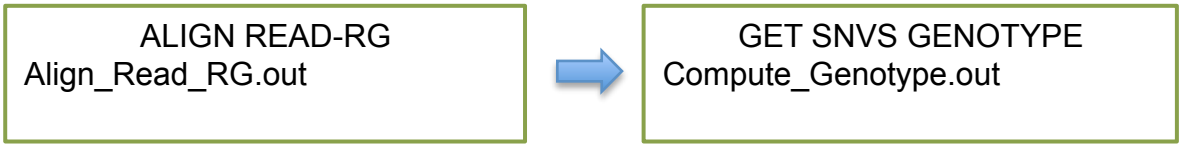

FIG 10. The processes from Stage 2 are listed.

ALIGN READ-REFERENCE GENOME

In this step every read is cut based on the positions recorded in the READALN files. This partial sequence read is aligned to the corresponding region in the RG. There are two different kind of reads, 1) the ones containing only the PrevCS from which only partial alignments of the VSR can be generated; and 2) the ones containing both Signature CSs from which global alignments (total aln) of the VSR can be generated . In this script reads with incongruency in the alignments with the Signature CS's are discarded, the PCR duplicates are eliminated**. Only the alignments which contain regions that has been categorized as variable in the child are analyzed**, no gaps are allowed and only alleles found in ReadPerbase alignments are recorded. The software used at this stage is the same software used in One Individual – Stage4.

COMMAND LINE.

Align_read_RG –a READALN-file –r REFDIR/ -s sufixREF -k k –t ReadRegion -p ReadPerbase –l LengthPartial -c ChildSNV –d RemoveDupFlag –i Ind –o out.total -q out.perbase

PARAMETERS.

- a [READALN-FILE] File per chromosome that contains information about the alignment between the read and the respective Signature CS's
- r [REFDIR] RG directory path
- a [sufixREF] Sufix of the fasta reference files
- k [k] CSs size = Jellyfish database kmer size
- t [ReadRegion] Parameter used in One Individual SNV discovery(set to 1)
- p [ReadPerbase] Each mismatch nucleotide between the reads and the RG must be supported by at least ReadPerbase different reads
- l [LengthPartial] Length of extensión of partial alignment (set to 10)
- c [ChildSNV] Path to variable regions file for child individual (out.total)
- d [RemoveDupFlag] Set to TRUE to remove PCR duplicates (FALSE otherwise)
- i [Ind] Set to PARENT for SNV discovery in the parents
- o [out.total] Set to NA for SNV discovery in the parents
- q [out.perbase] Output file with single nucleotide polymorphism data

OUTPUT.

A file with genotypes per basepair per chromosome was obtained with the following information:

1. RG chromosome

2. PrevCS start position
3. PostCS start position
4. SNV RG position
5. RG nucleotide
6. Read alleles (allele1/allele2)
7. Total alignments supporting every allele (allele1/allele2/total)
8. Partial alignments supporting every allele (allele1/allele2/total)
9. Total number of alignments supporting every allele (allele1/allele2/total)

NOTES.
Nomenclature: RG chr1 file name: chr1{sufixREF}.
One process must be run per chromosome.

To detect the low complexity alignments: start and end positions for regions that contain consecutive polymorphisms were obtained, if multiple regions spanning the same nucleotide intervals were obtained, all the alignments for those low complexity regions were discarded.

To detect PCR duplicates: multiple reads for which the PrevCS was aligned to the same position were considered as PCR duplicates and one of them (randomly chosen) was assigned to be the representative read.

GET SNVs GENOTYPE

In this script, the likelihood for each genotype is computed and the most probable genotype (given the observed data) is reported as the final genotype. The probability for four possible genotypes is computed: Heterozygous reference (R/NR), Homozygous reference (R/R), Heterozygous non-reference (NR1/NR2), Homozygous non-reference (NR/NR). For the child, this script does not print the homozygous reference-sites. The software used at this stage is the same software used in One Individual – Stage4.

COMMAND LINE.

Compute_Genotype –p out.perbase –t FilePerbaseType –i IndividualType
-u undetermined.out > genotype.out

PARAMETERS.

- p [out.perbase] File with single nucleotide polymorphism data
- t [FilePerbaseType] P if total number of aln per allele is in column 9
- i [IndividualType] Set to PARENT for SNV discovery in the parents
- u [undetermined.out] Output file with undetermined genotypes
- genotype.out File with genotype information per SNV

OUTPUT.
A file with assigned genotypes per basepair per chromosome with 11 columns:

- 1-9 will contain the information from the file out.perbase
- A numeric classificarion for the most probable genotype:
  - 1 - Heterozygous reference (Ref/NoRef)
  - 2 - Homozygous reference (Ref/Ref)
  - 3 - Heterozygous non-reference (NoRef1/NoRef2)
  - 5 - Homozygous non-reference (NoRef1/NoRef1)
- The different alleles for the assigned genotype

NOTES.
One process must be run per chromosome.

STAGE 3 – OBTAIN *DE NOVO* SNV LIST

In this Stage the SNV data for all individuals from the TRIO are analyzed jointly. In the first process the inheritance mode for each SNV in the child is obtained (either congruent with mendelian inheritance or incongruent). To interrogate any position, genotypes must be succesfuly assigned for any individual and total alignments should exist in the region. In the second process a minimum coverage is required for the variant region and for the variant allele, besides the difference in coverage between the different alleles must be lower than a required threshold

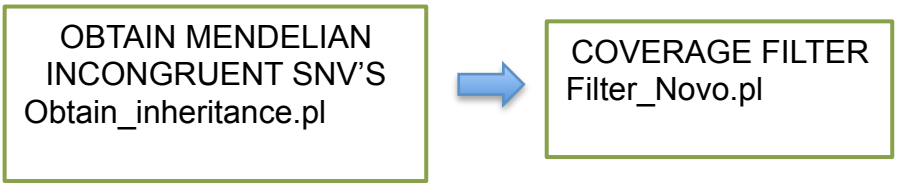

FIG11. The processes from Stage 3 are listed.

OBTAIN MENDELIAN INCONGRUENT SNV'S

In this scrip the inheritance mode for each SNV in the child is obtained (either congruent with mendelian inheritance or incongruent). To interrogate any genomic position, genotypes must be succesfuly assigned for any individual and total alignments should exist in the region.

COMMAND LINE.

Obtain_Inheritance.pl –hg3 childGenotype –hg1 fatherGenotype -hg2 motherGenotype –chr chr –out OUTDIR

PARAMETERS.

- dir_hg3 File with genotype information for the child
- dir_hg1 File with genotype information for the father
- dir_hg2 File with genotype information for the mother
- chr Chromosome
- out Output diectory

OUTPUT.

- A file with all SNVs with their inheritance mode: either mendelian congruent or incongruent. The information contained in this file is the same information of the genotype file for the child. In addition the last column contained the inheritance mode [CONGRUENT | INCONGRUENT]
- A SUMMARY file with mendelian congruent genotypes per chromosome [mend.congruent]
- A SUMMARY file with mendelian incongruent genotypes per chromosome [mend.incongruent]
- Error files with all genomic positions that:
  - Are not present in either the father or the mother SNV list
  - Have failled to succesfully assign a genotype in any individual
  - Don't have total alignments in any individual

The SUMMARY files contain:

1. RG chromosome
2. Child PrevCS start position
3. Child PostCS start position
4. SNV RG position
5. RG nucleotide
6. Read alleles for the child (allele1/allele2)
7. Total alignments supporting every allele for the child (allele1/allele2/total)
8. Partial alignments supporting every allele for the child (allele1/allele2/total)
9. Total number of alignments supporting every allele for the child (allele1/allele2/total)
10. A numeric classification for the most probable genotype for the child:
    a. 1 - Heterozygous reference (Ref/NoRef)
    b. 2 - Homozygous reference (Ref/Ref)
    c. 3 - Heterozygous non-reference (NoRef1/NoRef2)
    d. 5 - Homozygous non-reference (NoRef1/NoRef1)
11. The different alleles for the assigned genotype for the child
12. Read alleles for the father
13. Total alignments supporting every allele for the father
14. Partial alignments supporting every allele for the father
15. Total number of alignments supporting every allele for the father

16. A numeric classification for the most probable genotype for the father.
17. The different alleles for the assigned genotype for the father
18. Read alleles for the mother
19. Total alignments supporting every allele for the mother
20. Partial alignments supporting every allele for the mother
21. Total number of alignments supporting every allele for the mother
22. A numeric classification for the most probable genotype for the mother.
23. The different alleles for the assigned genotype for the mother.

These files will be written to:

- OUTDIR /{chr}.mend
- OUTDIR /{chr}.mend.congruent
- OUTDIR /{chr}.mend.incongruent

NOTES.
This step require one process per chromsome
If there is none total alignments in at least one of the individual, that event is nos reported in subsequent output files.

COVERAGE AND PURITY FILTER

Several characteristics are required for a mendelian incongruent SNV to be classified as *de novo*. 1) A minimum coverage is required for the variant region and for the variant allele for every individual; 2) The difference in coverage between the different alleles(in the child) must be lower than a required threshold; 3)No parent should habe both child alleles contained in more than one total alignment.

COMMAND LINE.

Filter_Novo.pl -dir child-GENOTYPE/ -sufix mend.incongruent –cov_child 5 –total_child 5 –cov_parent 5 –total_parent –min MIN -stat Stat.out > result.out

PARAMETERS.

- dir — Directory with mendelian incongruent genotype
- sufix — Sufix for the mendelian incongruent genotype files
- cov_child — Minimum number of reads [either partial or total] in the child
- total_child — Minimum number of total alignments in the child
- cov_parent — Minimum number of reads [either partial or total] in either parent
- total_parent — Minimum number of total alignments in either parent
- min — Events with a ratio higher than MIN between the read counts for the different alleles are filtered out
- stat — Output file with statistics of filtered SNVs

- result.out Output with the SNVs that PASSED the filtering criteria

OUTPUT.
The same information as the SUMMARY file with mendelian incongruent genotypes per chromosome [mend.incongruent]

NOTES
In the VGH pipeline, all the SNVs found in dbSNP are not considered as real de novo SNVs

ADDITIONAL SCRIPTS

COVERAGE STATISTICS

COMMAND LINE.
perl Calculate_Coverage_Statistics.pl –dir LANDDIR –pat sufixLAND -out out.stat

PARAMETERS
- dir VL directory path
- pat VL files sufix
- out Output file

OUTPUT
A file with coverage statistics:
- Total number of CS's analyzed.
- The coverage mean
- The coverage standard deviation
- The maximum coverage
- The first quartile, the median, the third quartile and the IQR of the coverage

NOTES
Some other debug values are printed

POST-PROCESSING

Once the candidate de novo SNVs have been identified, this script will analyze the regions corresponding to the child VSR for the three family individuals to identify undesired patterns. 1) regions with low CS density; 2) regions in which any CS has a

coverage higher than expected; 3) for any individual regions with low coverage for the CSs corresponding to the child Signature CSs, 4) regions with additional peaks inside the region corresponding to the child VSR : in the case of the child if there is any additional drop or rise it should correspond to a region with almost no coverage; in the case of the parents there should not exist any drop or rise that indicates a posible heterozigosity for the child SNV position or there should not exist a drop and rise that correspond to the exact same child's VSR boundaries; and 5) for the child, regions with unequal coverage in both sides of the VSR

COMMAND LINE.

perl Postprocessing.pl -novo Novo.out –land LAND-DIR/ -chr CHR –parent P -density DEN –max MAX -higher HIGH -cov COV -rci RCI –rmin RMIN -low LOW -peaks PEAK -out Novo.chr.genome.tp

PARAMETERS

- novo — File with de candidate *de novo* SNVs
- land — Directory with the VLs (Variation landscapes)
- chr — The chromosome to be analyzed [CHR]
- parent — If the landscape corresponds to the child set [P] to 0
If the landscape corresponds to the parent set [P] to 1
- density — Regions with a density of at least [DEN] will be kept
- max/higher — Regions with at most [HIGH] CSs with a coverage higher than [MAX] will be kept
- cov — Regions for which the CSs corresponding to the child's Sginature CSs have a coverage of at least COV will be kept
- rci — A change higher than RCI in the RCI index will be considered significative
- rmin — A change in coverage will be considered significative only if the coverage for any CS is no lower than RMIN
- low — For the child, a CS with a coverage lower than LOW will be considered as a region of "almost no coverage"
- peaks — Regions with less than PEAK significative changes in coverage inside the VSR will be kept
- out — Output file. This will contain all TP de novo SNVs

OUTPUT

Same columns as the input file de de novo SNVs (Novo.out)

NOTES

This script should be run for every chromosome for every individual landscape.
The TP for all individuals for every chromosome will be merged with the script in the following section.

MERGE POST-PROCCESING

The SNVs for which its VSR is classified as a TP region for all three individuals are identified.

COMMAND LINE.

python Compare_TP.py -f HG1_TP.tab -m HG2_TP.tab -c HG3_TP.tab -o Novo_TP.tab –p Novo_TP.info

python Compare_TP.py --fatherTP HG1_TP.tab --motherTP HG2_TP.tab --childTP HG3_TP.tab --output Novo_TP.tab --outputALL Novo_TP.info

PARAMETERS

- fatherTP, f — File with the whole-genome TP *de novo* SNVs for the father
- motherTP, m — File with the whole-genome TP *de novo* SNVs for the mother
- childTP, c — File with the whole-genome TP *de novo* SNVs for the child
- output, o — Output file. TP in all three individuals
- outputALL, p — More information about TP in all three individuals

OUTPUT

Same columns as the input file de de novo SNVs (Novo.out)

NOTES

This script should be run only once

ACRONIMS

CS	Coin String
RG	Reference Genome
SP	Sequencing Project
VL	Variation Landscape
VSR	Variation Signature Region
SNV	Single Nucleotide Variant
PrevCS	PreviousCS
PostCS	PosteriorCS

REFERENCES

- Bowtie version supported 1.1.0 [http://bowtie-bio.sourceforge.net/index.shtml]
- Jellyfish version supported 1.1.6 [http://www.cbcb.umd.edu/software/jellyfish/]
- Python version supported 2.7.2