V-Phaser 2.0 User Manual

Xiao Yang

Genome Sequencing and Analysis Program
The Broad Institute of MIT and Harvard

February 11, 2013

Contents

1 General Description

V-Phaser 2.0 is a variant inference program for viral populations. It requires only a BAM file as the input, and it has been applied to both the 454 and Illumina paired read data. It should be applicable to other types of data as well, such as Ion Torrent. V-Phaser 2.0 has been tested on datasets with average fold coverage ranging from hundreds to up to half a million.

V-Phaser 2.0 differs from the previous version in the following aspects: 1) it reduced the runtime and memory usage substantially; 2) it recalculates consensus from the alignment; 3) it models the error probability of indels differently from substitutions; 4) it considers phasing information from Illumina paired end reads; 5) Strand bias test has been included in removing FPs. Similar to V-Phaser, V-Phaser 2.0 calibrates sequencing errors by considering read position, quality scores, di-nucleotide content, and in addition, the first or the second pair in the PE reads.

2 Quick Start – for Broad Institute Users

1. First making sure:
   - The input BAM file is sorted by coordinates. This can be achieved by using Samtools:
     $ samtools sort input.bam > sort
   - A read/read-pair is aligned to only one reference genome. This can be achieved by using Samtools:
     $ samtools view -b -f 2 input.bam > output.bam
   - In the same folder where the input bam file resides, there exists no corresponding .bti file (index file used by Bamtools).
2. Use bash environment.

   $ bash
3. Export Bamtools library path.

   $ export LD_LIBRARY_PATH=/seq/viral/analysis/xyang/programs/Library/pezmaster31-bamtools-e235c55/lib:$LD_LIBRARY_PATH
4. Run V-Phaser 2.0

   $ OMP_NUM_THREADS=8 /seq/viral/analysis/xyang/programs/VariantCaller/bin/variant_caller -h

   This will show you options to set parameters for the program. Only two parameters are mandatory, -i and -o. You can change 8 to the number of cores (CPUs) you wish to use. For example:

   $ OMP_NUM_THREADS=8 /seq/viral/analysis/xyang/programs/VariantCaller/bin/variant_caller -i myInput.sorted.bam -o myOutputFolder

```
# Ref_Pos Var Cons Strd_bias_pval Type Var_perc SNP_or_LP_Profile
# ----------------------------------------------------------
55 T C 0.8156 snp 16.1 C:65:34 T:13:6
104 G A 0.1674 snp 14.07 A:66:50 G:14:5
210 T C 0.1065 snp 10.58 C:93:93 T:15:7
... ...
# ----------------------------------------------------------
# Summary: SNPV: 132;  LPV: 0
```

Table 1: Example output variant file, detailed version.

The above command can be "bsub" -ed, for example, via the following command:

$ bsub -P ProjName -o {screen_output.txt} -q hour -W 4:00 -R "rusage[mem=4]" -n 4,8 -R 'span[hosts=1]' {variant_caller} -i {myInput.sorted.bam} -o {myOutputFolder}

Please replace the values between {}.

5. Output – the results can be found in "{myOutputFolder}".
   (a) {ReferenceName}.fdr.var.txt – the result of interest, where strand bias test + FDR (false discovery rate) correction were used. {ReferenceName} is the name of the reference in the input BAM file. An example is given in Table 1. Each variant entry consists of the following: the reference position (coordinate starts at 1), predicted variant, consensus base, strand-bias p-value, type of variant (SNP or LP), frequency of the variant and the profile, where each entry consists of three values: the base, its count in the forward strand, and its count in the reverse strand, separated by Colons.
   {ReferenceName}.var.raw.txt – the raw variants without strand bias test.
   {ReferenceName}.nofdr.var.txt – strand-bias test but no FDR correction.
   (b) {ReferenceName}.start.end.region, {ReferenceName}.eb: these are intermediate files that can be ignored.
   (c) {ReferenceName}.covplot.R: R script that provides the coverage plot.

3 Quick Start – for non-Broad Institute users

3.1 Pre-requisite

1. Installation of BamTools (version e235c55 or later) (instructions can be found here `https://github.com/pezmaster31/bamtools/wiki`).
2. Boost Library (version 1_51_0 or later).
3. Installation of Perl (recent versions are recommended)
4. g++ compiler (recent versions are recommended)

3.2 Procedure

1. Download the V-Phaser 2.0 package, decompress, and "cd" into the "VPhaser-2" folder.
2. Switch to bash environment.

   $ bash
3. Export BamTool library path. Assuming you successfully installed BamTool-e235c55 in directory [path], then you should be able to find the library in directory "[path]/lib"

   $ export `LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/[path]/lib`
4. Compile and Run V-Phaser 2.0.
   (a) Edit file "VPhaser-2/src/makefile" –
   - set MYPATH to be [path], *e.g.* MYPATH=/MyLibrary/bamtools-e235c55/
   - set BOOSTPATH to be where it is located,
     e.g. BOOSTPATH=/MyLibrary/boost_1_51_0/
   - set COMPILER to be the path of the g++ compiler you are using (you could use command "$ which g++" to find out this information).

   (b) Compile in "src" directory

   $ cd VPhaser-2
   $ make
   $ cd ../

   Note:
   - The executive file can be found in the "VPhaser-2/bin" folder.
   - By default the exe file is compiled with "-fopenmp" flag and named "variant_caller". These settings can be modified.

   (c) Run V-Phaser 2.0

   First making sure:
   - Input BAM file is sorted by coordinates. This can be achieved by using Samtools:
     $ samtools sort input.bam > sort
   - A read/read-pair is aligned to only one reference genome. This can be achieved by using Samtools:
     $ samtools view -b -f 2 input.bam > output.bam
   - In the same folder where the input bam file resides, there exists no corresponding .bti file (index file used by Bamtools).

   $ OMP_NUM_THREADS=n ./bin/variant_caller -h

Table 2: Explanation of Parameters

Mandatory Parameters	
-i	input sorted BAM file
-o	output directory
Optional Parameters	
-e	0 or 1; default 1; 1: pileup + phasing; 2: pileup
-w	default 500; alignment window size
-ig	default 0; # of bases to ignore on both end of a read
-delta	default 2; constrain PE distance by delta × fragsize_variation, which is automatically measured by the program
-ps	(0, 100]; default 30; percentage of reads to be sampled by the program in order to get basic statistics of the data
-dt	0 or 1; default 1; 1: enable dinucleotide content for calibrating error probabilities, 0: disabled.
-cy	0 or 1; default 1; 1: enable read cycle for calibrating error probabilities, 0: disabled.
-mp	0 or 1; default 1; 1: enable mate-pair for calibrating error probabilities, 0: disabled.
-qual	[0, 40] – default 20; enable quality scores for calibrating error probabilities, this value specifies how many quantile to cluster the quality scores, when set to 0, this feature is disabled.
-a	default 0.05; significance value for stat test

This will show you options to set parameters for the program. Only two parameters are mandatory, -i and -o. n is the number of cpus you would like to use. For example:

$ OMP_NUM_THREADS=8 ./bin/variant_caller -i {myInput.sorted.bam} -o {myOutputFolder}

Please replace the values between {}.

5. Output – refer to section 2 point 5.

6. Test run – if the program is compiled correctly, you can type the following command in "VPhaser-2 " folder, where you can find the results of interest in "TestData/output" folder.

   $ OMP_NUM_THREADS=8 ./bin/variant_caller -i TestData/4528.454.indelRealigned.bam -o TestData/output

4 Parameter Setting See Table 2

5 License

Please refer to license folder.

6 Citing V-Phaser 2.0

Xiao Yang, Patrick Charlebois, Alex Macalalad, Matthew R. Henn and Michael C. Zody, "V-Phaser 2.0: variant inference for viral populations", (in review)

7 Contact

If you have any question, please email Xiao Yang (xiaoyang@broadinstitute.org).