Chapter 4

Whole-Genome Shotgun Sequence CNV Detection Using Read Depth

Fatma Kahveci and Can Alkan

Abstract

With the developments in high-throughput sequencing (HTS) technologies, researchers have gained a powerful tool to identify structural variants (SVs) in genomes with substantially less cost than before. SVs can be broadly classified into two main categories: balanced rearrangements and copy number variations (CNVs). Many algorithms have been developed to characterize CNVs using HTS data, with focus on different types and size range of variants using different read signatures. Read depth (RD) based tools are more common in characterizing large (>10 kb) CNVs since RD strategy does not rely on the fragment size and read length, which are limiting factors in read pair and split read analysis. Here we provide a guideline for a user friendly tool for detecting large segmental duplications and deletions that can also predict integer copy numbers for duplicated genes.

Key words Copy number variation, Whole genome shotgun sequencing, Read depth, mrFAST, mrsFAST

Abbreviations

CNV	Copy number variation
mrCaNaVaR	Micro read Copy Number Variant Regions
mrFAST	Micro read Fast Alignment Search Tool
mrsFAST	Micro read substitution only Fast Alignment Search Tool
RD	Read depth
TRF	Tandem repeat finder
WGS	Whole genome sequencing

1 Introduction

CNVs are changes in the amount of DNA in a genome that show themselves as duplication (gain) and deletion (loss) events [1]. Traditionally, a copy number variant is defined as a gain or loss in the amount of DNA in the genome that is larger than 1 Kb, although smaller deletions or duplications (>50 bp) are considered CNV in several projects such as the 1000 Genomes Project. CNVs are known to have played major role in evolution [2–4], and several of them are correlated with complex human disease [5, 6].

There exist many algorithms to characterize CNVs in genomes that employ different strategies to characterize different classes of CNVs in different size ranges. Here we describe the mrCaNaVaR algorithm [7] that uses read depth information to predict large (>10 Kb) duplications and deletions, along with integer copy numbers for genes. mrCaNaVaR was the first tool developed solely for accurate assessment of segmental duplications on personal genomes, and it also was the only publicly available tool for integer copy number prediction. Although it was initially developed to analyze only human segmental duplications, it has been used to characterize large CNVs in many organisms including cattle [8], great apes [3, 9], Neandertal [10] and Denisovan [11, 12] ancient DNA, and even plant genomes [13].

Here we describe how to use mrCaNaVaR for detecting large CNVs. Since segmental duplication prediction necessitates tracking multiple map locations of reads, we need to use a multi-mapper [14]. Briefly, after mapping all reads to a repeat masked genome (to "clean up" common repeats) using mrFAST or mrsFAST [15, 16], mrCaNaVaR loads the mapping information (i.e., SAM files) and applies a sliding window strategy to count read depth. The read depth distribution is often affected by G + C content of genomic segments, therefore mrCaNaVaR applies a statistical smoothing method (LOESS) to correct for bias in high and low G + C regions. It then identifies those regions with higher or lower read depth than the genome-wide average (±3 standard deviations), and chains consecutive such regions. mrCaNaVaR also predicts integer copy numbers of windows of size 1 Kb as the ratio of observed read depth to average read depth, and uses this information to refine CNV predictions and the estimated breakpoints.

2 Materials

This section contains a list of prerequisite software tools that mrCaNaVaR requires. All tools described below are open source and free to use under the GNU public license.

2.1 Installing Zlib

1. Download the zlib program from https://sourceforge.net/projects/libpng/files/zlib/. An example of a Linux command to use is as follows (*see* **Notes 1**, **2**, **3**, and **4**):

   $ wget https://sourceforge.net/projects/libpng/files/zlib/X.X.X/zlib-X.X.X.tar.gz

   where X.X.X denotes version number.

2. Unpack the downloaded file using the command shown below:

   ```
   $ tar -xzvf zlib-X.X.X.tar.gz
   ```

   where X.X.X denotes version number.

   There should be a new folder named zlib-X.X.X in the current working directory.

3. Remove the zipped version of file:

   ```
   $ rm zlib-X.X.X.tar.gz
   ```

4. Change directory into zlib-X.X.X:

   ```
   $ cd zlib-X.X.X
   ```

5. Type at the shell prompt:

   ```
   $ ./configure
   $ make test
   ```

   If everything goes well, then type:

   ```
   $ sudo make install
   ```

Depending on the version of the GCC compiler and some additional library requirements, the installation should be completed without any errors (*see* **Note 5**).

2.2 Installing Blast

1. Download the Blast program from ftp://ftp.ncbi.nlm.nih.gov/blast/executables/LATEST/. An example of a Linux command to use is as follows:

   ```
   $ wget ftp://ftp.ncbi.nlm.nih.gov/blast/executables/LATEST/ncbi-blast-X.X.X+-x64-linux.tar.gz
   ```

   where X.X.X denotes version number.

2. Unpack the downloaded file using the command shown below:

   ```
   $ tar -xzvf ncbi-blast-X.X.X+-x64-linux.tar.gz
   ```

   where X.X.X denotes version number.

   There should be a new folder named ncbi-blast-X.X.X+ in the current working directory.

3. Remove the zipped version of file:

   ```
   $ rm ncbi-blast-X.X.X+-x64-linux.tar.gz
   ```

4. Change directory into ncbi-blast-X.X.X+/bin:

   ```
   $ cd ncbi-blast-X.X.X+/bin
   ```

5. Download ncbi-rmblastn from http://ftp.ncbi.nlm.nih.gov/blast/executables/rmblast/.

   ```
   $ wget ftp://ftp.ncbi.nlm.nih.gov/blast/executables/rmblast/X.X.X/ncbi-rmblastn-X.X.X-x64-linux.tar.gz
   ```

   where X.X.X denotes version number.

6. Unpack the downloaded file using the command shown below:

   ```
   $ tar -xzvf ncbi-rmblastn-X.X.X-x64-linux.tar.gz
   ```

where X.X.X denotes version number.

There should be a new folder named ncbi-rmblastn-X.X.X in the current working directory.

7. Copy rmblastn file into /path/ncbi-blast-X.X.X+/bin directory:

   $ cp /path/ncbi-rmblastn-X.X.X/rmblastn /path/ncbi-blast-X.X.X+/bin

8. Remove unused files in ncbi-blast-X.X.X+/bin directory:

   $ rm ncbi-rmblastn-X.X.X*

2.3 Installing Tandem Repeats Finder

1. Change directory into RepeatMasker:

   $ cd /path/RepeatMasker

2. Download "*trf*" file from http://tandem.bu.edu/trf/trf409.linux64.download.html.

3. Convert binary file into executable:

   $ chmod 755 trf409.linux64

4. Change file name:

   $ mv trf409.linux64 trf

2.4 Installing RepeatMasker

1. Download the RepeatMasker program from http://www.repeatmasker.org. An example of a Linux command to use is as follows:

   $ wget http://www.repeatmasker.org/RepeatMasker-open-X-X-X.tar.gz

   where X-X-X denotes version number.

2. Unpack the downloaded file using the command shown below:

   $ tar -xzvf RepeatMasker-open-X-X-X.tar.gz

   There should be a new folder named RepeatMasker in the current working directory.

3. Remove the zipped version of file:

   $ rm RepeatMasker-open-X-X-X.tar.gz

4. Change directory into RepeatMasker:

   $ cd RepeatMasker

5. RepeatMasker provides two open databases, Dfam and Dfam_consensus, and will work with these datasets, but it is advised to obtain a license for the RepBase RepeatMasker Edition to supplement these sequences. To obtain a license and download the library go to http://www.girinst.org. Copy RepBase into the RepeatMasker directory:

   $ cp RepBaseRepeatMaskerEdition-XXXXXXXX.tar.gz /path/RepeatMasker/

6. Change directory into:

   ```
   $ cd /path/RepeatMasker
   ```

7. Unpack the downloaded file using the command shown below:

   ```
   $ tar -xzvf RepBaseRepeatMaskerEdition-########.tar.gz
   ```

   There should be a new folder named Libraries in the current working directory.

8. Remove the zipped version of file:

   ```
   $ rm RepBaseRepeatMaskerEdition-########.tar.gz
   ```

9. Check for Dfam Updates (optional).
   (a) Change directory into Libraries:
   (b) Download the Dfam.hmm.gz file from http://www.dfam.org.

       ```
       $ wget http://www.dfam.org/web_download/Current_Release/Dfam.hmm.gz
       ```

   (c) Unpack the downloaded file using the command shown below:

       ```
       $ gunzip Dfam.hmm.gz
       ```

   There should be a new file named Dfam.hmm in the current working directory.

10. Change directory into RepeatMasker:

    ```
    $ cd /path/RepeatMasker/
    ```

11. Type at the shell prompt (*see* **Note 6**):

    ```
    $ perl ./configure
    ```

12. Follow the instructions.

2.5 Installing mrFAST

1. Download the mrFAST program from https://github.com/BilkentCompGen/mrfast/. An example of a Linux command to use is as follows (*see* **Notes 1**, **2**, **3**, and **4**):

   ```
   $ wget https://github.com/BilkentCompGen/mrfast/archive/vX.X.X.X.zip
   ```

   where X.X.X.X denotes version number.

2. Unpack the downloaded file using the command shown below:

   ```
   $ tar -xzvf mrfast-X.X.X.X.tar.gz
   ```

   where X.X.X.X denotes version number. There should be a new folder named mrfast-X.X.X.X in the current working directory.

3. Remove the downloaded compressed version of the file:

   ```
   $ rm mrfast-X.X.X.X.tar.gz
   ```

   where X.X.X.X denotes version number.

4. Change directory into mrfast-X.X.X.X:

```
$ cd mrfast-X.X.X.X
```

where X.X.X.X denotes version number.

5. Type at the shell prompt:

```
$ make
```

6. Make it executable from anywhere in terminal using the commands shown below (optional) or you will give the path of mrFAST tool to run:

```
$ nano ~/.bashrc
```

Add the current directory of mrFAST at the end of the file using the following command:

```
export PATH=$PATH:/mrFAST_directory/
```

Save the file and close. Then, run the following command to make it permanent:

```
$ source ~/.bashrc
```

2.6 Installing mrsFAST (Alternative to mrFAST Above)

1. Install git:

```
$ sudo apt install git
```

this assumes that the underlying Linux OS is Debian-based. For other Linux distributions, refer to official git page at https://git-scm.com/.

2. Download the mrsFAST program from https://github.com/sfu-compbio/mrsfast. An example of a Linux command to use is (*see* **Notes 1**, **2**, **3**, and **4**):

```
$ git clone https://github.com/sfu-compbio/mrsfast
```

3. Change directory into mrsfast:

```
$ cd mrsfast
```

4. Type at the shell prompt:

```
$ make
```

5. Make it executable from anywhere in terminal using the commands shown below (optional) or you will give the path of mrsFAST tool to run:

```
$ nano ~/.bashrc
```

Add the current directory of mrsFAST at the end of the file using the following command:

```
export PATH=$PATH:/mrsFAST_directory/
```

Save the file and close. Then, run the following command to make it permanent:

```
$ source ~/.bashrc
```

2.7 Installing mrCaNaVaR

1. Download the mrCaNaVaR program from https://github.com/BilkentCompGen/mrcanavar. An example command line is (*see* **Notes 1** and 7):

   ```
   $ git clone https://github.com/BilkentCompGen/mrcanavar
   ```

2. Change the working directory to mrcanavar

   ```
   $ cd mrcanavar
   ```

3. Compile by typing make:

   ```
   $ make
   ```

4. Make it executable from anywhere in terminal using the commands shown below (optional) or you will give the path of mrCaNaVaR tool to run:

   ```
   $ nano ~/.bashrc
   ```

   Add the current directory of mrCaNaVaR at the end of the file using the following command:

   ```
   export PATH=$PATH:/mrCaNaVaR_directory/
   ```

   Save the file and close. Then, run the following command to make it permanent:

   ```
   $ source ~/.bashrc
   ```

3 Methods

In the first preprocessing step, common and tandem repeats will be masked with N characters in reference genome assembly using RepeatMasker [17] and Tandem Repeats Finder [18] tools. Here lowercase letters will be treated as nonrepeats. For the human genome assemblies, we recommend to use both:

- *RepeatMasker (with sensitive "-s" parameter enabled):* The repeat masked versions of many genome assemblies are available at the UCSC Genome Browser.
- *Tandem Repeats Finder* (parameters: trf input_file 2 7 7 80 10 50 500 -m)

In the second preprocessing step, mrFAST (or mrsFAST) search index will be built using the repeat masked version of the same reference genome (*see* **Note 8**). For mrFAST, you can use the default 12 character-window size without losing the sensitivity if your calculated window size is greater than default.

```
$ ./mrfast --index $(reference_genome).fasta , OR
$ ./mrsfast --index $(reference_genome).fasta
```

Index file will be created in the same directory upon the completion of the indexing phase. The next step using samtools is optional. If skipped, merely the output SAM files will not include headers. You can create $(reference_genome).masked.fai using samtools:

$ samtools faidx $(reference_genome).masked.fa

In the third preprocessing step, reads will be mapped to the reference genome using mrFAST (or mrsFAST) in single-read mapping mode, and *not* in paired-end mapping mode. Here mrFAST (or mrsFAST) may be used for read mapping. Although other aligners might also be used if all map locations, including suboptimal alignments, are reported, no other mapper is recommended to use with mrCaNaVaR for characterization of duplications and absolute copy number prediction. It is strongly recommended to use mrFAST (or mrsFAST) that was specifically developed for SV/CNV analysis. The same repeat masked version of the reference genome described above will be used for building the mrFAST (or mrsFAST) index and read mapping. FASTA and index file of the reference genome should be found in the same folder. mrFAST reports all the locations per read by default. Here recommended edit/hamming distance threshold is 5% of the read length (i.e., 2 for 36 bp reads).

$ mrfast --search $(reference_genome).fasta --seq $(input_file).fastq -e $(distance_threshold).

The reported locations will be saved into "output" by default. If you want to save it somewhere else, you can use "-o" parameter to specify another file.

$ mrfast --search $(reference_genome).fasta --seq $(input_file).fastq -o $(output_directory).

Both mrFAST and mrsFAST are capable of generating map files in plain and gzip-compressed SAM format.

mrCaNaVaR has four running modes.

3.1 Preparing Reference--prep

Users need to run this mode only once for a new reference genome assembly unless repeat masking or the sliding window sizes are altered. In this mode, coordinates and G + C content of three sliding window classes (Long Windows (LWs), Short Windows (SWs), and Copy Windows (CWs)) over the reference genome are calculated and saved in a *genome configuration* file ($(reference_genome).cnvr) (Table 1).

Sliding windows: mrCaNaVaR operates on three classes of sliding windows over the reference genome to estimate absolute copy numbers and discover duplicated and deleted regions. The three windows are the following:

1. *Long Windows (LWs)* are used to find locations of deletions and duplications.

   The LWs are overlapping windows of size 5 kbp of unmasked characters (i.e. A, C, G, T only), sliding by 1 kbp of any characters. For example, in repeat masked GRCh37 assembly, the first three LWs are:

```
chr1        10,000 17,048
chr1        11,000 17,048
chr1        12,000 17,236
```

Table 1
mrCaNaVaR in --prep mode

Sample command line
$ mrcanavar --prep -fasta $(reference_genome)_masked.fasta -gaps $(reference_genome)_gaps.bed -conf $(reference_genome).cnvr
Inputs
--**fasta:** Repeat masked reference genome (FASTA format)
-**gaps:** Gap coordinates of the reference genome (BED format)
Outputs
-**conf:** Reference configuration file (a native binary format)
Optional parameters
-**lw_size <lw_size>:** Long window span size. Default is 5000 bp of non-masked characters which are A, C, G, and T.
-**lw_slide <lw_slide>:** Long window slide size. Default is 1000.
-**sw_size <sw_size>:** Short window span size. Default is 1000.
-**sw_slide <sw_slide>:** Short window slide size. Default is 1000.
-**cw_size <cw_size>:** Copy number window size. Default is 1000.
-**pseudoa <bed_file>:** Coordinates for pseudoautosomal regions in the reference genome (BED format).

In the example above, the first window has 7048 any characters, but 5000 non-N characters (thus 2048 N characters). We then slide the window by 1000 *any* base pairs, starting the second window at position 11,000. The end of the second window is again 17,048 since all of the first 1000 characters of chr1 are masked. The third window starts at position 12,000 and contains 5326 any characters, where 5000 are non-N.

2. *Short Windows (SWs)* are used to refine breakpoints of detected duplication and deletion intervals. Default window size is 1000 bp of nonmasked characters, and the slide size is 1000 bp of any characters, similar to LW.
3. *Copy Windows (CW)* are nonoverlapping windows and used to predict absolute copy numbers. Default window size is 1000 bp of nonmasked characters.

3.2 Reading/Loading Alignments--read

In this mode, mapping information in SAM format which is either uncompressed (*.sam) or compressed with gzip (*.sam.gz) is loaded, the read depth in LW, SW, and CW intervals are calculated, and read depth information is be saved in a *depth* output file in a native binary format (Table 2). Additionally, three human-readable text files are generated for the raw and G + C corrected read depths and G + C contents of LW, SW, and CW intervals. These three

Table 2
mrCaNaVaR in --read mode

Sample command line
$ mrcanavar --read -conf $(reference_genome).cnvr --gz -samdir $(sam_file_directory) -depth $(sample).depth
$ mrcanavar --read -conf $(reference_genome).cnvr --gz -samdir $(sam_files_directory) -depth $(sample).mult.depth --multgc
Inputs
-**conf:** Reference configuration file created using the PREP mode.
-**samdir:** Directory of plain SAM files (*.sam) or gzip-compressed SAM files (*.sam.gz)
Outputs
- **$(sample).depth:** Read depth file in a native binary format.
Optional parameters
--**gz:** Indicates the SAM files are compressed in gzip format. -**nsam <first> <second>:** Choose a sequence of SAM files between two indexes.

Table 3
mrCaNaVaR in --conc mode

Sample command line
$ mrcanavar --conc -conf $(reference_genome).cnvr -concdepth lib1.depth lib2.depth lib3.depth -depth merged.depth
Inputs
-**conf:** Reference configuration file created using the PREP mode.
-**concdepth:** Read depth file(s) to merge.
Outputs
-**depth:** Read depth file in a native binary format.

human readable text files are generated for manual inspection and debugging purposes only, if needed, and will not be loaded and needed by mrCaNaVaR when calling CNVs.

3.3 Merging Multiple Libraries--conc if they Exist

The CONC mode is optional and needed only if the SAM files are read into a number of *.depth files or if there are more than one library preparation which should be read into different *.depth files. In this mode, mrCaNaVaR will be used to merge more than 2 *.depth files using the READ mode and output the merged LW, SW, and CW intervals (Table 3).

3.4 Calling CNVs and Copy Numbers--call

In this mode, segmental duplication and deletion intervals are called and absolute copy numbers are predicted. Absolute copy number over the CW intervals, and duplicated and deleted regions in the analyzed sample using the LW and SW intervals will be predicted. Additionally, if a BED file that contains genes in the reference genome is provided, the median and average copy numbers of the intervals spanning the genes are generated. We recommend using the median copy number values for genes (Table 4).

Table 4
mrCaNaVaR in --call mode

Sample command line
$ mrcanavar --call -conf $(reference_genome).cnvr -depth dep $sample.depth -o $sample
Inputs
-**conf:** Reference configuration file created using the PREP mode.
-**depth:** Read depth file created using the READ mode.
-**o:** Prefix for the output file names.
Outputs
- **$(sample).log:** Log file that describes what mrCaNaVaR did including read depth statistics.
- **$(sample).copynumber.bed:** Estimated copy numbers over CW intervals.
- **$(sample).dels.bed:** Estimated deleted regions over CW intervals.
- **$(sample).dups.bed:** Estimated duplicated regions over CW intervals.
- **$(sample).cw_norm.bed:** GC-corrected read depth values for the inferred control regions for CW intervals.
- **$(sample).lw_norm.bed:** GC-corrected read depth values for the inferred control regions for LW intervals.
- **$(sample).sw_norm.bed:** GC-corrected read depth values for the inferred control regions for SW intervals.
Optional Parameters
--**xx:** Set gender of the sequenced sample as female. Mammalian genomes only. Default is autodetect.
--**xy:** Set gender of the sequenced sample as male. Mammalian genomes only. Default is autodetect.
--**multgc:** Perform multiplicative GC correction. Default is additive.
-**mindup <min_dup_len>:** Minimum duplication length. Default is 10000.
-**gene <genelist>:** Coordinates for genes calculating gene-based copy numbers.
--**verbose:** Verbose output.

4 Notes

1. wget is a computer program that retrieves content from web servers.
2. Perl version 5.8.0 or higher installed is required.
3. mrFAST with FastHASH (post 2.5.0.0) is drastically faster than plain mrFAST (pre 2.5.0.0). When using mrFAST, please ensure that the version is >2.5.0.0.
4. zlib is for the ability to read compressed SAM files.
5. Installing programs in Unix like systems requires admin privileges.
6. The program requires some initial configuration. This should also be rerun after future updates to the library files.
7. mrCaNaVaR is developed with gcc version 4.1.2 (C compiler)
8. mrsFAST is multithreaded and faster than mrFAST. Running mrsFAST is similar to mrFAST. Manual for mrsFAST is found at https://github.com/sfu-compbio/mrsfast//blob/master/README.md.

References

1. Alkan C, Coe BP, Eichler EE (2011) Genome structural variation discovery and genotyping. Nat Rev Genet 12:363–376
2. Ventura M, Catacchio CR, Alkan C et al (2011) Gorilla genome structural variation reveals evolutionary parallelisms with chimpanzee. Genome Res 21:1640–1649
3. Prado-Martinez J, Sudmant PH, Kidd JM et al (2013) Great ape genetic diversity and population history. Nature 499:471–475
4. Sudmant PH, Huddleston J, Catacchio CR et al (2013) Evolution and diversity of copy number variation in the great ape lineage. Genome Res 23:1373–1382
5. Sharp AJ, Cheng Z, Eichler EE (2006) Structural variation of the human genome. Annu Rev Genomics Hum Genet 7:407–442
6. Sharp AJ, Hansen S, Selzer RR et al (2006) Discovery of previously unidentified genomic disorders from the duplication architecture of the human genome. Nat Genet 38:1038–1042
7. Alkan C, Kidd JM, Marques-Bonet T et al (2009) Personalized copy number and segmental duplication maps using next-generation sequencing. Nat Genet 41:1061–1067
8. Bickhart DM, Hou Y, Schroeder SG et al (2012) Copy number variation of individual cattle genomes using next-generation sequencing. Genome Res 22:778–790
9. Prado-Martinez J, Hernando-Herraez I, Lorente-Galdos B et al (2013) The genome sequencing of an albino Western lowland gorilla reveals inbreeding in the wild. BMC Genomics 14:363
10. Green RE, Krause J, Briggs AW et al (2010) A draft sequence of the Neandertal genome. Science 328:710–722
11. Reich D, Green RE, Kircher M et al (2010) Genetic history of an archaic hominin group from Denisova Cave in Siberia. Nature 468:1053–1060
12. Meyer M, Kircher M, Gansauge M-T et al (2012) A high-coverage genome sequence from an archaic Denisovan individual. Science 338:222–226
13. Cardone MF, D'Addabbo P, Alkan C et al (2016) Inter-varietal structural variation in grapevine genomes. Plant J 88:648–661
14. Chiang DY, McCarroll SA (2009) Mapping duplicated sequences. Nat Biotechnol 27:1001–1002
15. Hach F, Hormozdiari F, Alkan C et al (2010) mrsFAST: a cache-oblivious algorithm for short-read mapping. Nat Methods 7:576–577
16. Hach F, Sarrafi I, Hormozdiari F et al (2014) mrsFAST-ultra: a compact, SNP-aware mapper for high performance sequencing applications. Nucleic Acids Res 42:W494–W500
17. Smit AFA, Hubley R, Green P (1996–2004) RepeatMasker Open-3.0
18. Benson G (1999) Tandem repeats finder: a program to analyze DNA sequences. Nucleic Acids Res 27:573–580