***Protocols for Perl programs***

1) ***vrGVdatabase.pl*** (original name *PopulationDistribution\_vrGV\_ids\_pop14.pl* ).

1.1. This program opens 22 input files (obtained from phase 1, 1000 GENOMES Project) with names ‘ALL.chr'.$L.'.phase1\_release\_v3.20101123.snps\_indels\_svs.genotypes.vcf.gz' , where variable $L is an integer from 1 to 22.

1.2. This program opens supplementary file 1000genomes\_ID, which is a single line (the 30th line) from the ‘ALL.chr1.phase1\_release\_v3.20101123.snps\_indels\_svs.genotypes.vcf.gz' files representing all 1092 individual identifiers (file is provided on the web).

1.3. In addition, this program opens supplementary file 20111108\_1000genomes\_samples2.txt, which was downloaded from phase 1, 1000 GENOMES Project and represents Excel file specifying populations and individuals (file is provided on the web).

1.4. Command line for execution of this program:

perl vrGVdatabase.pl

1.5. Output file name: vrGVdatabase.txt

1.6. In order to run *vrGVdatabase.pl* automatically 1092 times, the script *get\_vrGVdatabase.pl* was used.

2) ***RVC.pl*** (original names *vrGV\_1individual\_relatives2015\_v4\_2.pl* and *VRH3.pl*).

2.1. This program opens an individual-specific vrGV database (e.g. GBR\_HG00255 file) and characterizes all shared clusters of vrGVs with the rest of 1091 individuals.

2.2. Command line:

perl RVC.pl GBR\_HG00255

The argument here is the name of the individual specific vrGV database the program should process.

2.3. This program creates two output files: GBR\_HG00255\_dat3 and GBR\_HG00255\_dat4. Finally, the entire content of GBR\_HG00255\_dat4 file is appended to the All\_Pairs.txt file.

2.4. **NOTES**. Stretching window ($str variable) by default is 2 rows. If there is at least one more shared vrGV in this window, it stretches for another cycle (two more rows), and so on until there is no shared vrGVs in the current stretching window.

2.5. To run *RVC.pl* automatically 1092 times the script *TotalStat\_vrGVs\_v3.pl*  was used.

3) **TotalStat\_vrGVs\_v3.pl**

3.1. This program opens an input file Population\_individual\_table representing a complete list of population\_individual identifiers (e.g. GBR\_HG00255). File is provided on the web.

3.2. The program executes 1092 times *RVC.pl* with the script line: system("perl RVC.pl $L");

where variable $L represents each identifier (e.g. GBR\_HG00255)

3.3. The command line for the execution:

perlTotalStat\_vrGVs\_v3.pl

**4) *Step6\_Table\_Mix.pl*** (original name *Step6\_Diff\_InandBetwPop.pl*)

4.1. This script processes an input All\_Pairs.txt file and divides the content into the bins (files) of corresponding populations. Command line for the execution:

perl Step6\_Table\_Mix.pl

4.2. Output files: Table\_ASW, Table\_CEU, etc. for each of the 14 populations and also a single Table\_Mix file containing all inter-population pairs.

**5)** ***Inter\_pop\_results2.pl***(original name *Gursel\_2pop\_inputALL\_final.pl*).

5.1. This program opens 14 input files (‘Population\_individual\_table'. $pop1). For example, Population\_individual\_tableASW, … Population\_individual\_tableJPT, that contain entire sets of identifiers for a particular population. (Files are provided on the web).

5.2. The command line for the execution:

perl Inter\_pop\_results2.pl

5.3. The program creates output files Result2\_ASW\_LWK, etc. for all possible combinations of two populations.

5.4. The program prints the average number and length of IBDs on the screen: “average number of IBDs is $mean \naverage length of IBD is $meanL \n 0 counts is $c0 \n 1 counts is $c1”

**6)** ***Intra\_pop\_results2.pl***(original name *Gursel\_ASWinput.pl*)

6.1. Command line for the execution:

perl Intra\_pop\_results2.pl Population\_individual\_tableGBR Table\_GBR 50

Here two arguments are the files described in sections 5.1 and 4.2.

6.2. The program creates an output file result2\_GBR

6.3. The program also prints the average number and length of IBDs on the screen: “average number of IBDs is $mean \naverage length of IBD is $meanL \n 0 counts is $c0 \n 1 counts is $c1”

**7) *Table\_Shuhao2L.pl***

7.1. This program opens two input files: table\_list and Population\_individual\_table\_sorted2 (Files are provided on the web)

7.2. The program creates main output file Table\_RVC\_Length and supporting file Table\_Shuhao1L

7.3. Command line for the execution:

perl Table\_Shuhao2L.pl

**8) *Table\_Shuhao2\_numberRVC.pl***

This program has minor modifications from *Table\_Shuhao2L.pl* only to write numbers of RVCs instead of lengths.

**9) *median.pl***

9.1. This program opens an input file table\_list2, which contains list of all Result2\_pop1\_pop2 filenames (for example Result2\_TSI\_CHB). Then it opens and processes all these Result2\_pop1\_pop2 files.

9.2. Command line for the execution:

perl median.pl >median\_average\_Table4\_Apr22

The output of this program is saved in the file median\_average\_Table4\_Apr22:

**10) *statistics\_length\_RVC.pl***

10.1. This program opens all “dat3” files (e.g. CHB\_NA18597\_dat3).

10.2. The program creates two types of output files length\_ASW\_MXL and info\_ASW\_MXL for every possible combination of two population names. These output data were used to calculate the statistics on the lengths of RVCs.

**11) *IBDsimulator.pl***

11.1. This program opens all **‘**genetic\_map\_chr'.$x.'\_b36.txt’files (recombination map tables) downloaded from NCBI web page for all 22 autosomes (variable $x is and integer from 1 to 22). Files are provided on the web.

11.2. The program creates gametes for the progenitor and his/her offspring according to the recombination map tables for a desirable number of progenitors specified by the user (variable $person).

11.2. Command line for the execution:

perl IBDsimulator.pl $L

Where variable $L is an integer. (for example: perl IBDsimulator.pl 5)

11.3. The program creates two output files: IBDmodeling\_F\_$L (for number of IBD segments in generations) and IBDmodeling\_L\_$L (for total lengths of IBD segments in generations. For example, IBDmodeling\_F\_5. The output files are the tables, where the first column represents the consecutive number of computational simulation (number of progenitor). The second column is a control value that always has 0 value. The third column represents the number of IBD segments (or their total lengths) in the G2-offspring (grand-child of the progenitor). The fourth column represents the number of IBD segments (or their total lengths) in the G3-offspring (great-grand-child of the progenitor). And so on for next generation offspring.

11.4. To run this program simultaneously 100 times on a 64-core linux computer, the perl script *IBDmodeling\_start.pl* was used. It contains the single line:

for $L (401..500) { system("nohup perl IBDsimulator.pl $L &"); }

11.5. Tables IBDmodeling\_F\_$L and IBDmodeling\_L\_$L are provided as supplementary file.