Documentation for Creating Exon Data

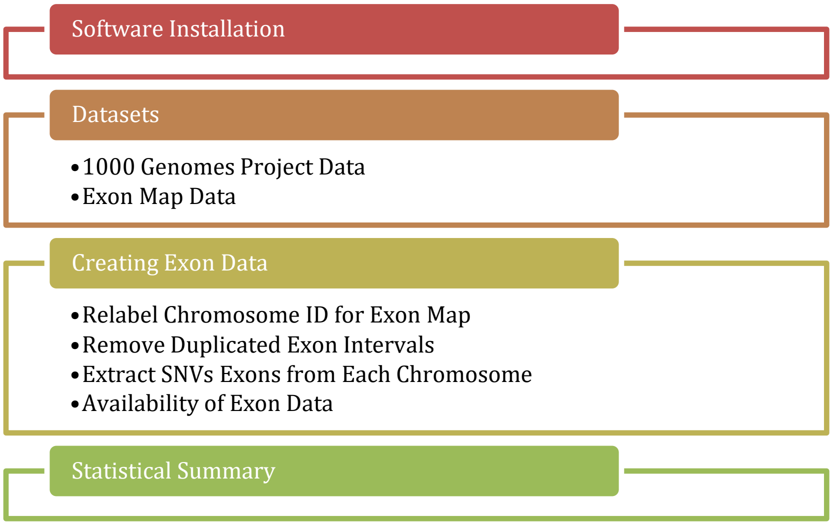
Wen Tian (Wendy) Wang, Christina Nieuwoudt, Jinko Graham, Brad McNeney
Department of Statistics, Simon Fraser University

Table of Contents

Introduction

Exons are the protein-coding regions in human genes and are fragmented into several disconnected pieces. Single nucleotide variants (SNVs) are a type of genetic variations that can be detected in the exon regions [1]. We have the whole-genome data from the 1000 Genome Project [2], and the ultimate goal is to extract SNVs in exon regions from the chromosomes. This documentation demonstrates how users may use bcftools and other shell scripting commands to read and manipulate genetic data, especially in an Ubuntu terminal on the Windows operating system. This documentation has not been tested in any other environment. The work flow in figure 1 provides users with an overview of what we did throughout the process of data extraction.

Figure 1. work flow


Software Installation
Datasets
•1000 Genomes Project Data
•Exon Map Data
Creating Exon Data
•Relabel Chromosome ID for Exon Map
•Remove Duplicated Exon Intervals
•Extract SNVs Exons from Each Chromosome
•Availability of Exon Data
Statistical Summary

We begin with step-by-step instructions to guide users through the installation of bcftools. Next, we describe the 1000 Genome Project data followed by a description of exon maps, which are required to subset the data. Next, we demonstrate the necessary Unix commands required to manipulate the genetic data. Fianlly, we conclude with a statistical summary of the resultant genome-wide whole-exome data.

Software Installation

The source code for bcftools is available at: https://samtools.github.io/bcftools/. To install bcftools on a WindowsOS, users can issue the commands in the terminal:

```
git clone git://github.com/samtools/htslib.git
git clone git://github.com/samtools/bcftools.git
cd bcftools
make
```

In the first and second lines of the code above, the command "`git clone`" clones the remote repository and downloads each package to the user's computer. The command "`cd`" changes the working directory to the location of the bcftools executable. The command "`make`", at the last line, complies the packages.

Before we can use bcftools, we must install several libraries that are required for Ubuntu. Both bcftools and HTSlib depend on the following libraries: `zlib` (http://zlib.net), `bzip2`(http://bzip.org/) and `liblzma` (http://tukaani.org/xz/). To install these libraries, the user must execute the following commands:

- zlib1g-dev

```
sudo apt-get install zlib1g-dev
```

- libbz2-dev

```
sudo apt-get install libbz2-dev
```

- liblzma-dev

```
sudo apt-get install liblzma-dev
```

The command "`sudo apt-get install`" installs and compiles the specified libraries. After installing these libraries, we can access the manual for bcftools by executing the command:

```
bcftools
```

We note that MacOS or other Linux distributions (e.g. RedHat / Fedora / CentOS) require different libraries. The other libraries are available at https://samtools.github.io/bcftools/howtos/install.html.

Datasets

1000 Genomes Project Data

The chromosome data from the 1000 Genomes Project, which is released on March 12, 2019, can be retrieved from http://ftp.1000genomes.ebi.ac.uk/vol1/ftp/data_collections/1000_genomes_project/release/20190312_biallelic_SNV_and_INDEL/. The data file contains both biallelic SNVs and INDEL variants for the 2548 samples separated by chromosome. The data for each chromosome is in Variant Call Format (VCF) file. Users can manually download the chromosome data from the website or use the "`wget`" Unix command or equivalent. The "`wget`" Unix command can download files directly from a website. For example, to download the VCF file for chromosome 22, the command is:

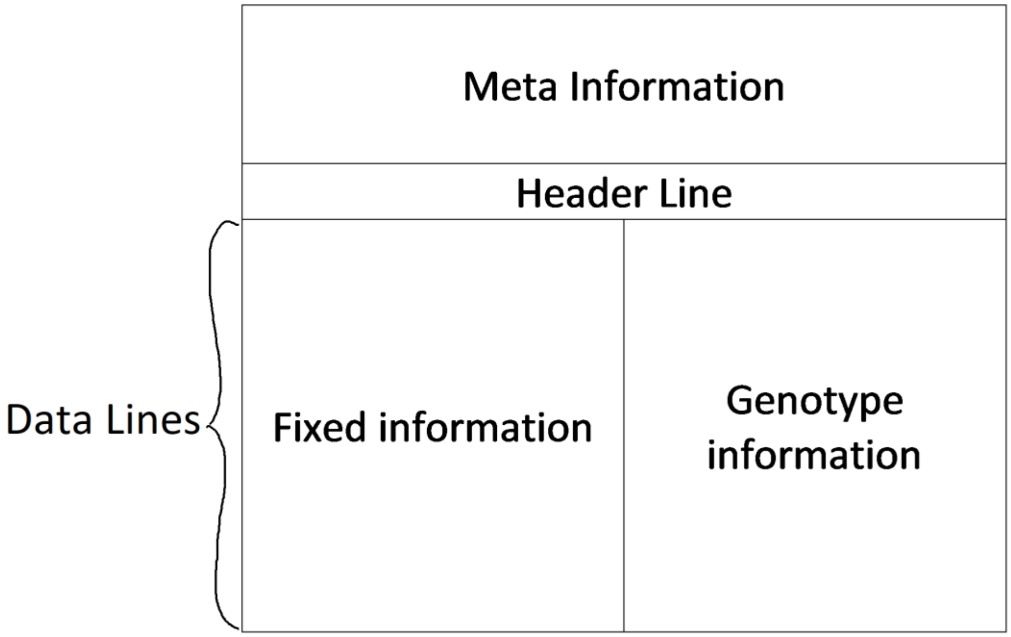
```
wget http://ftp.1000genomes.ebi.ac.uk/vol1/ftp/data_collections/1000_genomes_
project/release/20190312_biallelic_SNV_and_INDEL/ALL.chr22.shapeit2_integrate
d_snvindels_v2a_27022019.GRCh38.phased.vcf.gz
```

We also download the index file (ends with "vcf.gz.tbi") for chromosome 22. The index file is required for extraction, and allows users to quickly retrieve specified regions in the corresponding VCF file. The command for downloading the index file for chromosome 22 is:

```
wget http://ftp.1000genomes.ebi.ac.uk/vol1/ftp/data_collections/1000_genomes_
project/release/20190312_biallelic_SNV_and_INDEL/ALL.chr22.shapeit2_integrate
d_snvindels_v2a_27022019.GRCh38.phased.vcf.gz.tbi
```

Now, let's get back to VCF files. VCF files contain meta-information, header and data lines [3]. Each row in the data line consists of fixed and genotype fields for a single variant in the genome. An overview structure of VCF file is shown in figure 2.

Figure 2. Structure of a VCF file



Meta-information lines begin with double pound signs ("##") and contain information such as file creation (e.g. the date of the file is created and/or retrieved), filter conditions when applicable, as well as definitions of any abbreviations elsewhere used in the file. The demonstration of meta-information lines will be explained later. Now let's move on to header lines of a VCF file.

The header line is a row of tab-delimited column names, and it contains 8 required (fixed-field) columns:

1. `#CHROM`: Chromosome ID
2. `POS`: The reference position in increasing order
3. `ID`: Variant ID. If there is a dbSNP variant, a corresponding rs number(s) will be used. Otherwise it is replaced by a dot sign (".").
4. `REF`: The reference alleles, which must be one of A,C,G,T,N
5. `ALT`: The alternative alleles, which must be one of A,C,G,T,N
6. `QUAL`: Phred-scaled quality score out of 100. High QUAL scores indicate high possibility of variation
7. `FILTER`: indicates if this position has passed all filters, and is shown as "PASS" or "FAIL"
8. `INFO`: additional information about a variant, which are encoded as a semicolon-separated series of key with values in the format, "key=value". The abbreviation for these keys can be found in meta-information lines.

When genotype data are included, the 8 required fixed-field columns are followed by a `FORMAT` column as well as sample IDs for each individual.

E.g.

```
#CHROM  POS     ID      REF     ALT     QUAL    FILTER  INFO    FORMAT  HG00096
```

Lastly, we introduce the data lines; these are underneath the header line. Each data line contains 2 types of information for a single variant, the fixed information and the genotype information (see Figure 2). Any missing values in the data line are specified by a dot (".").

The genotype information starts from the beginning in the ninth column, which is the `FORMAT` column, and this column defines the data type. For example, an entry of "GT" in the `FORMAT` column indicates that any columns following the `FORMAT` column are the genotypes for each sample. That is, beginning in the 10th column, each column will contain the genotype data for an individual. We expect to see genotypes of the form such as "0|1", where 0 indicates the reference allele, and 1 indicates the alternate allele. The pipe symbol, "|", indicates that the genotype is phased. The entry to the left of the pipe symbol indicates the inherited allele from one parent, and the item to the right of the pipe symbol indicates the inherited allele from another parent. There are four distinct genotypes cases:

- 1|1 indicates that the individual carries 2 copies of the alternate alleles. Individuals with this genotype are referred to as "homozygous alternate".
- 1|0 and 0|1 indicate that the individual carries 1 copy of the alternate allele and one copy of the reference allele. Individuals with this genotype are referred to as "heterozygous"
- 0|0 indicates that the individual carries 2 copies of the reference alleles, and therefore does not carry any mutation. Individuals with this genotype are referred to as "homozygous reference".

An example of part of the VCF file for chromosome 22 is given below:

```
##fileformat=VCFv4.3
##FILTER=<ID=PASS,Description="All filters passed">
##fileDate=27022019_15h52m43s
##source=IGSRpipeline
##reference=ftp://ftp.1000genomes.ebi.ac.uk/vol1/ftp/technical/reference/GRCh38_reference_genome/GRCh38_full_analysis_set_plus_decoy_hla.fa
##FORMAT=<ID=GT,Number=1,Type=String,Description="Phased Genotype">
##contig=<ID=22>
##INFO=<ID=AF,Number=A,Type=Float,Description="Estimated allele frequency in the range (0,1)">
##INFO=<ID=AC,Number=A,Type=Integer,Description="Total number of alternate alleles in called genotypes">
##INFO=<ID=NS,Number=1,Type=Integer,Description="Number of samples with data">
##INFO=<ID=AN,Number=1,Type=Integer,Description="Total number of alleles in called genotypes">
##INFO=<ID=EAS_AF,Number=A,Type=Float,Description="Allele frequency in the EAS populations calculated from AC and AN, in the range (0,1)">
##INFO=<ID=EUR_AF,Number=A,Type=Float,Description="Allele frequency in the EUR populations calculated from AC and AN, in the range (0,1)">
##INFO=<ID=AFR_AF,Number=A,Type=Float,Description="Allele frequency in the AFR populations calculated from AC and AN, in the range (0,1)">
##INFO=<ID=AMR_AF,Number=A,Type=Float,Description="Allele frequency in the AMR populations calculated from AC and AN, in the range (0,1)">
##INFO=<ID=SAS_AF,Number=A,Type=Float,Description="Allele frequency in the SAS populations calculated from AC and AN, in the range (0,1)">
##INFO=<ID=VT,Number=.,Type=String,Description="indicates what type of variant the line represents">
##INFO=<ID=EX_TARGET,Number=0,Type=Flag,Description="indicates whether a variant is within the exon pull down target boundaries">
##INFO=<ID=DP,Number=1,Type=Integer,Description="Approximate read depth; some reads may have been filtered">
##bcftools_viewVersion=1.9-162-g33ecfe8+htslib-1.9-150-gc76b3b2
##bcftools_viewCommand=view ALL.chr22.shapeit2_integrated_snvindels_v2a_27022019.GRCh38.phased.vcf.gz; Date=Tue Aug  6 12:01:54 2019
#CHROM  POS     ID      REF     ALT     QUAL    FILTER  INFO    FORMAT  HG00096
22  10516173  .  A  G   .  PASS  AC=121;AN=5096;DP=8203;AF=0.02;EAS_AF=0;EUR_AF=0.02;AFR_AF=0.06;AMR_AF=0.02;SAS_AF=0;VT=SNP;NS=2548  GT  0|0
22  10522217  .  G  A  .  PASS   AC=89;AN=5096;DP=9085;AF=0.02;EAS_AF=0;EUR_AF=0;AFR_AF=0.07;AMR_AF=0;SAS_AF=0;VT=SNP;NS=2548  GT  0|0
```

Note that in the third line of the meta-information lines, we are given the information for file creation date, which is on February 27, 2019. As well, at the last line of the meta-information section, we are provided the information for file retrieve date, which is on August 6, 2019. Next, let's discuss the first data line beginning after the header, and firstly focus on CHROM, POS, REF, ALT, FORMAT and HG00096 columns. From this line we see that, in chromosome 22, at position 10516173, there is a record of a mutation, A|G. The reference allele is A, and the alternate allele is G. We note that this is a new mutation because the ID for this mutation is missing and therefore has not yet been identified by an rs number in the dbSNP database. Recall that we expect to see genotype of the form such as "0|0". Beginning in the tenth column, we see the first genotype record for the first sample subject named "HG0096". Since this sample has genotype 0|0, this subject does not carry any alternate allele at this variant. Next, we explain some of the features in the INFO column. Looking at the INFO column for the first mutation we see the following information:

```
AC=121;AN=5096;DP=8203;AF=0.02;EAS_AF=0;EUR_AF=0.02;AFR_AF=0.06;AMR_AF=0.02;SAS_AF=0;VT=SNP;NS
=2548
```

The abbreviations in this field are defined in the meta information contained in the VCF file and begin with "##INFO=". For example, the 8th meta-information line:

```
##INFO=<ID=AF,Number=A,Type=Float,Description="Estimated allele frequency in the range (0,1)">
```

indicates that the INFO variable "AF" is the estimated allele frequency, and it is rounded with two decimal places. From the information included in the INFO field for the first mutation we see that: the total number of alternative alleles in called genotypes is 121 (AC=121), the total number of alleles in called genotypes is 5096 (AN=5096), and the estimated alternative allele frequency is 0.02 (AF=0.02). Since the estimated alternate allele frequency is rounded to two decimal places, users may wish to calculate the alternate allele frequency independently by dividing AC from AN.

Exon Map Data

The exon map data is in Browser Extensible Data (BED) file format and can be retrieved from http://ftp.1000genomes.ebi.ac.uk/vol1/ftp/data_collections/1000_genomes_project/working/20190125_coords_exon_target/output_1000G_Exome.v1.bed. This dataset was released on January 25, 2019, and we retrieved it on May 21, 2019. Note that the exact locations of exons can differ according to the whole-exome sequencing technology that is being used; thus, the exon positions that we use are based on the 1000 Genomes Project sequencing. Besides, BED file format provides a flexiable way for data creators to define their own data fields. Our data only includes 3 required fields and 2 optional fields, which will be descripted in detail in the overview of BED file format.

The BED files contain 3 to 12 fields of tab-delimited data. Three required fields are:

chrom - The name of the chromosome (e.g. chr3, chr2_random) or scaffold (e.g. scaffold10671).

chromStart - The start position of the feature in the chromosome or scaffold.

chromEnd - The end position of the feature in the chromosome or scaffold.

There are other 9 optional fields can be included to the data. However, our exon map data, `output_1000G_Exome.v1.bed`, only contains two optional fields. Thus, we restrict our attention to the following two optional fields:

`strand` - Defines the strand orientation. Either "." (=no strand) or "+" or "-"

`name` - Defines name of each feature. In our dataset, the name indicates the targeted regions captured in the Exome capture platforms

For the full description of the nine optional fields, it is available in the appendix A. Furthermore, when any optional fields are included, the number of fields per line are consistent throughout the data, so any missing values are replaced with a dot symbol ("."). Moreover, each row of the data describes a targeted exon segment with the start and end positions and additional information about the target region if any other optional fields are added. The following demonstrates the first ten rows of the BED file with the five fields previously defined.

```
chr1    14642   14882   +       target.0
chr1    14943   15063   +       target.1
chr1    15751   15990   +       target.2
chr1    16599   16719   +       target.3
chr1    16834   17074   +       target.4
chr1    17211   17331   +       target.5
chr1    30275   30431   +       target.6
chr1    69069   70029   +       target.7
chr1    129133  129253  +       target.8
chr1    258482  258603  +       target.9
```

From the output above, the first column is the chromosome ID, second and third columns indicate start and end positions of an exon segment respectively. The fourth column describes the strand orientation, and the last column is the user defined name, which is named after the target region in sequencing. Now, let's look at the first row as an example. We see that an exon segment is detected in chromosome 1, and it starts from position 14642 and ends at position 14882. The "+" sign in the `strand` column indicates a positive strand orientation, and it is labelled as target.0. Later when we do the extraction from the chromosome data, our goal is to extract the variants that are falling within these exon intervals.

Creating Exon Data

Relabel Chromosome ID for Exon Map

First, we download the exon map file, `output_1000G_Exome.v1.bed`, using the following command:

```
wget http://ftp.1000genomes.ebi.ac.uk/vol1/ftp/data_collections/1000_genomes_
project/working/20190125_coords_exon_target/output_1000G_Exome.v1.bed
```

The Unix command “wget” allows users to download files from the internet. Also note that, right after the last slash symbol, “/”, that is the name of the exon positions data.

The first ten rows of data are displayed below.

```
chr1    14642   14882   +       target.0
chr1    14943   15063   +       target.1
chr1    15751   15990   +       target.2
chr1    16599   16719   +       target.3
chr1    16834   17074   +       target.4
chr1    17211   17331   +       target.5
chr1    30275   30431   +       target.6
chr1    69069   70029   +       target.7
chr1    129133  129253  +       target.8
chr1    258482  258603  +       target.9
```

The chromosome IDs in the first field are of the form “chrN”, where N=1,…,22,X,Y. Thus, in order to match the chromosome IDs in the VCF files for the genome data, which are of the form 1,…,22, we remove the first three characters, “chr”, from chromosome ID in the first column with the “cut” command, as suggested by user “finswimmer” on the biostars forum page https://www.biostars.org/p/309753/. This is accomplished by executing the following command:

```
cut -c4- output_1000G_Exome.v1.bed  > 1KG.exons.bed
```

The options in above command are:

“cut”: removes the regions we specified from each row in the data

“-c4-”: selects characters from the fourth index to the last index in each row, so the first three characters, “chr”, are removed.

“output_1000G_Exome.v1.bed”: is the input file that we just download.

“>” symbol creates a new output file and redirects the result to this output file

“1KG.exons.bed”: is the output file

After relabelling the chromosome ID, the first ten rows of the output of “1KG.exons.bed” now appear as follows:

```
1       14642   14882   +       target.0
1       14943   15063   +       target.1
1       15751   15990   +       target.2
1       16599   16719   +       target.3
1       16834   17074   +       target.4
1       17211   17331   +       target.5
1       30275   30431   +       target.6
1       69069   70029   +       target.7
1       129133  129253  +       target.8
1       258482  258603  +       target.9
```

Remove Duplicated Exon Intervals

We note that some exon segments in this BED file may overlap. This can occur in one of three ways:

1) the end position of one interval may equal to the start position of the next exon;

2) exon intervals may be duplicated;

3) exon intervals may overlap other exon intervals.

To avoid duplicated exon segments, we use the "`merge`" command provided by "bedtools" package to combine overlapping exon intervals into unique intervals. We then save the result to a new file called `1KG.exons.bash_merged.bed`. Note that data is required to be sorted before merging, so we sort by chromosome IDs and then by start positions of the exon intervals. Then, we can use "`bedtools merge`" to remove duplicated exon intervals. These two commands can be written in one line using a pipe symbol, "|", and this action is similar as the pipe operation, "%>%", in R-studio, which takes the output from the "`sort`" command and passes to the "`bedtools merge`" command. The command line for doing these two tasks is shown as the follows:

```
sort -V -k1,1, -k2,2 1KG.exons.bed | bedtools merge > 1KG.exons.bash_merged.b
ed
```

The options are:

"`sort`": sort lines of input

"`-V`" : sort by natural version

"`-k1,1`": sort by the first column, which is the Chromosome ID column ("`-k`" stands for sorting "key")

"`-k2,2`": sort by the second columns (start position of the exon interval)

"`bedtools merge`": merge overlapped exons intervals

"`1KG.exons.bed`": input file

"`1KG.exons.bash_merged.bed`": output file

As an example, we show the first ten rows of the output after running both of the "`sort`" and "`merge`" commands. Note that the "`merge`" command automatically keeps the first three columns only, and that is the information we really need for extracting exon data in the chromosomes.

```
1       14642   14882
1       14943   15063
1       15751   15990
1       16599   16719
1       16834   17074
1       17211   17331
```

```
1       30275   30431
1       69069   70029
1       129133  129253
1       210818  210938
```

Extract SNVs in Exons from Each Chromosome

After preparing the exon map, we are ready to extract SNVs in the exons from each chromosome. Fortunately, `bcftools` provides users a convenient way to select SNVs in the exon regions for each chromosome by using the "`view`" command, and we can use this command for filtering [4]. This powerful tool also allows users to save output directly into a compressed VCF file. As a demonstration, we show users how to extract SNVs from chromosome 22. The code is displayed as follows:

```
bcftools view
  --regions-file 1KG.exons.bash_merged.bed
  --types snps --min-alleles 2 --max-alleles 2
  --include 'FILTER="PASS"'
  --output-type z
  --output-file exons_chr22.vcf.gz
  ALL.chr22.shapeit2_integrated_snvindels_v2a_27022019.GRCh38.phased.vcf.gz
```

In the first line, we use "`view`" command in the "bcftools" packages for filtering. Then the following lines indicates the filter conditions and these options are:

`--regions-file 1KG.exons.bash_merged.bed`: filters to exons specified in bed file

`--types snps --min-alleles 2 --max-alleles 2`: includes diallelic SNVs only

`--include 'FILTER="PASS"'`: includes variants that passed all quality filters. However, this option may not be necessary because the 1000 Genome Project data has already done this step by keeping only the variants that pass all filters. Nevertheless, it could be a good practice if other genome data is used.

`--output-type z`: compresses the output

`--output-file exons_chr22.vcf.gz`: output file for the extracted exon data

`ALL.chr22.shapeit2_integrated_snvindels_v2a_27022019.GRCh38.phased.vcf.gz`: input file for the 22nd chromosome

An overview of all the above commands, such as the cleaning step for exon map data and extraction step in the chromosome data, can be found in the appendix B. Note that the first two commands for cleaning the exon map data are one-time commands, which means that we only need to do it once. However, for the command for extraction, it is applied to each of the chromosome, so using a "`for`" loop for this iteration is more convenient. In the "`for`" loop, we download both of VCF file (e.g files that end with vcf.gz) and its index file (e.g files that end with vcf.gz.tbi) and do the extraction based on our filter conditions. The general syntax of a "`for`" loop command in the shell scripting is:

```
for (( expr1; expr2; expr3 ));
do
 command1;
 command2;
 ..;
done
```

The commands to extract exons are given below.

```
for ((i=1;i<=22;i++));

do printf "\n start downloading vcf file for Chr $i \n";

wget http://ftp.1000genomes.ebi.ac.uk/vol1/ftp/data_collections/1000_genomes_
project/release/20190312_biallelic_SNV_and_INDEL/ALL.chr$i.shapeit2_integrate
d_snvindels_v2a_27022019.GRCh38.phased.vcf.gz;

wget http://ftp.1000genomes.ebi.ac.uk/vol1/ftp/data_collections/1000_genomes_
project/release/20190312_biallelic_SNV_and_INDEL/ALL.chr$i.shapeit2_integrate
d_snvindels_v2a_27022019.GRCh38.phased.vcf.gz.tbi;

bcftools view
  --regions-file 1KG.exons.bash_merged.bed
  --types snps --min-alleles 2 --max-alleles 2
  --include 'FILTER="PASS"'
  --output-type z
  --output-file exons_chr$i.vcf.gz
  ALL.chr$i.shapeit2_integrated_snvindels_v2a_27022019.GRCh38.phased.vcf.gz;

done;
```

The first command line indicates that this "for" loop is applied to the first chromosome until the 22nd chromosome. The rest of options after the first command line are:

printf: prints a string during each loop

$i: is the command substitution, and it takes values from 1,…,22

wget: downloads files from websites, and the first file is for VCF file then follows by the index file

The remaining options of the "view" command in the "bcftools" package have been described previously.

As an example, the first five rows (without header) and the first ten columns of the output of exons in chromosome 22 are shown as the following:

```
22      16390584        .       C       T       .       PASS    AC=23;AN=5096;DP=82786;AF=0;EAS_AF=0;EUR_A
F=0;AFR_AF=0.01;AMR_AF=0;SAS_AF=0;EX_TARGET;VT=SNP;NS=2548    GT      0|0
22      16390592        .       A       G       .       PASS    AC=1386;AN=5096;DP=86545;AF=0.27;EAS_AF=0.
37;EUR_AF=0.34;AFR_AF=0.08;AMR_AF=0.33;SAS_AF=0.32;EX_TARGET;VT=SNP;NS=2548    GT      0|1
22      16390594        .       C       G       .       PASS    AC=1;AN=5096;DP=87990;AF=0;EAS_AF=0;EUR_AF
```

```
=0;AFR_AF=0;AMR_AF=0;SAS_AF=0;EX_TARGET;VT=SNP;NS=2548        GT      0|0
22      16390595        .       A       G       .       PASS    AC=1;AN=5096;DP=88574;AF=0;EAS_AF=0;EUR_AF
=0;AFR_AF=0;AMR_AF=0;SAS_AF=0;EX_TARGET;VT=SNP;NS=2548        GT      0|0
22      16390599        .       G       T       .       PASS    AC=1;AN=5096;DP=90114;AF=0;EAS_AF=0;EUR_AF
=0;AFR_AF=0;AMR_AF=0;SAS_AF=0;EX_TARGET;VT=SNP;NS=2548        GT      0|0
```

Please refer to page 5, where we describe the 1000 Genome Project Data, for a detailed description of these output. From the tenth column onwards, there are the genotype data for the 2548 individuals. However, the output above only shows the genotype data for the first individual. Users can view the first data line with the command below.

```
bcftools view -H exon_chr1.vcf.gz | head -1
```

where the options are:

`-H`: show only data lines as result

`head -1`: output the first row

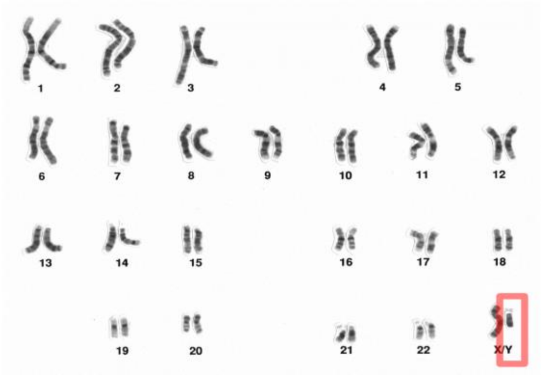
Availability of Extracted Exon Data

We have extracted SNVs in exon regions for all chromosomes except for the non-sex chromosomes (e.g chr X and chr Y). The extracted data for each chromosome is stored in a separate VCF file and can be retrieved from https://github.com/simrvprojects.

Statistical Summary

After extracting SNVs in the exons, we summarize the data. Before showing the summaries, we provide a figure of all chromosomes for a male in figure 3 [5].

Figure 3. All chromosomes in a male human



From the figure, we see that the length of each non-sex chromosome decreases as the number increase. We expect our results to display a similar trend.

We first summarize the chromosome and exon data in table 1 with information on:

1. The length of each chromosome, in base-pairs
2. The total length of exons in each chromosome, in base-pairs
3. The proportion of the total length of the chromosome comprised of exons
4. Th number of SNVs in each chromosome

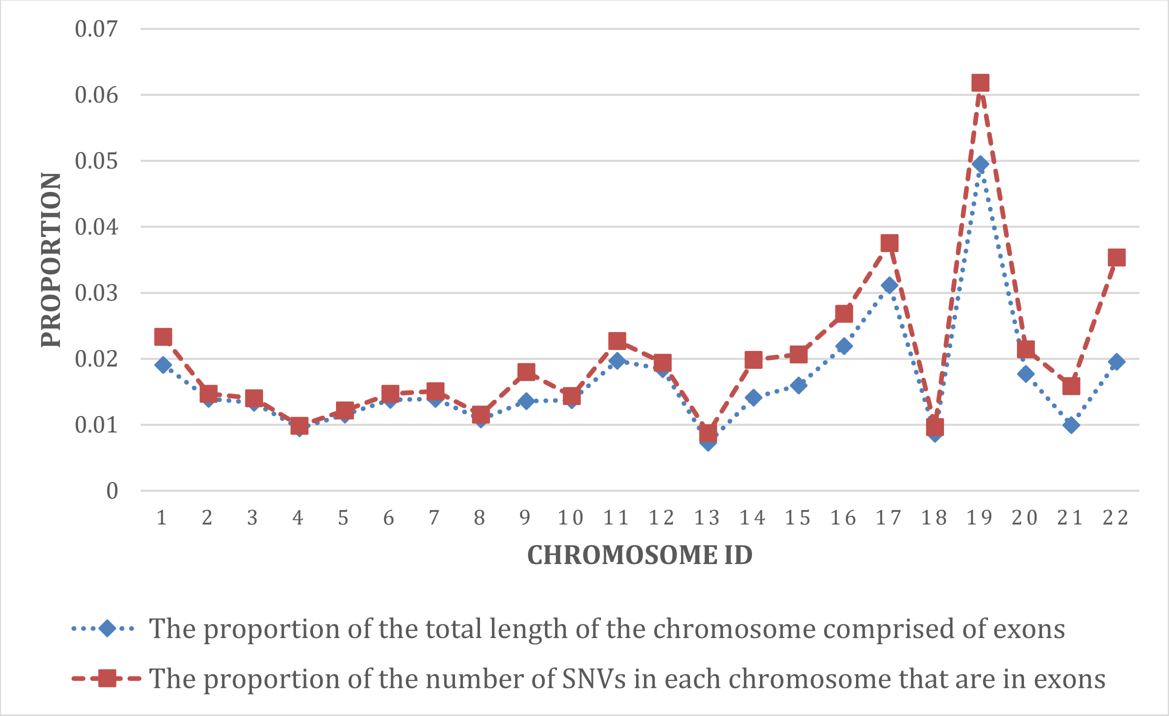
5. The number of SNVs in exons in each chromosome.
6. The proportion of the number of SNVs in each chromosome that are in exons.

Table 1. Chromosome and exon summary statistics.

	SEQUENCE LENGTH			NUMBER OF SNVs		
CHR ID	**Chromosome**	**Exons**	**Proportion (exon/chrom)**	**Chromosome**	**Exons**	**Proportion (exon/chrom)**
1	248956422	4744046	0.019055729	5795045	135131	0.023318369
2	242193529	3380247	0.013956801	6356815	93369	0.014688016
3	198295559	2649196	0.013359835	5280536	73981	0.014010131
4	190214555	1804113	0.009484621	5120664	50394	0.009841302
5	181538259	2094357	0.011536725	4788374	58450	0.012206649
6	170805979	2355905	0.013792872	4539754	66706	0.014693748
7	159345973	2217707	0.013917559	4222931	63714	0.015087625
8	145138636	1573127	0.010838789	4162377	48157	0.011569591
9	138394717	1880161	0.013585497	3178999	57197	0.017992142
10	133797422	1836619	0.013726864	3632297	52226	0.01437823
11	135086622	2664265	0.019722641	3642067	82633	0.02268849
12	133275309	2456270	0.018430045	3500318	67909	0.019400809
13	114364328	831833	0.007273536	2575087	22499	0.008737181
14	107043718	1514628	0.014149621	2383125	47345	0.019866772
15	101991189	1627578	0.015958026	2153932	44498	0.020658962
16	90338345	1983729	0.021958881	2410531	64662	0.026824795
17	83257441	2592240	0.031135235	2066684	77558	0.03752775
18	80373285	694372	0.008639338	2047353	19751	0.009647091
19	58617616	2902587	0.049517316	1625698	100551	0.061850971
20	64444167	1140997	0.017705202	1706442	36585	0.021439346
21	46709983	464800	0.009950764	976599	15524	0.015895982
22	50818468	994178	0.019563321	993880	35146	0.035362418
TOTAL	2875001522	44402955	0.015444498	73159508	1313986	0.017960564

We expect the proportion of the total length of the chromosome comprised of exons to be similar to the proportion of the number of SNVs in each chromosome that are in exons. To verify this, we plot both proportions in figure 4. The shell scripting and excel commands for making this figure are available in appendix C. From figure 4, we observe that the curve for the total length of the chromosome comprised of exons is very close to the curve for the proportion of the number of SNVs in each chromosome that are in exons, which it is reassuring.

Figure 4. Comparison of the proportion of the total length of the chromosome comprised of exons to the proportion of the number of SNVs in each chromosome that are in exons



References

1. Sastre, Leandro. (2014). Exome sequencing: what clinicians need to know. Advances in Genomics and Genetics. 2014. 15. 10.2147/AGG.S39108.
2. 1000 Genomes Project Consortium (2010) A map of human genome variation from population-scale sequencing. Nature, 467, 1061–1073
3. The variant call format and VCFtools. Danecek P, Auton A, Abecasis G, Albers CA, Banks E et al. Bioinformatics (Oxford, England) 2011;27;15;2156-8
4. Variant calling and filtering, https://samtools.github.io/bcftools/howtos/variant-calling.html
5. National Human Genome Research Institute, "NHGRI human male karyotype." National Human Genome Research Institute, 1 Mar. 2012, http://www.genome.gov/glossary/resources/karyotype.pdf.

Appendix

A. The Nine Optional Fields for the BED File

name - Defines the name of the BED line. This is display to the left of the BED line in the Genome Browser window.

score - A score between 0 and 1000.

strand - Defines the strand. Either "." (=no strand) or "+" or "-".

thickStart - The starting position at which the feature is drawn thickly (for example, the left boundary of coding sequence). ThickStart and thickEnd are set to the chromStart if no coding sequence is displayed.

thickEnd - The ending position at which the feature is drawn thickly (for example, the right boundary of coding sequence).

itemRgb - Defines an RGB color value of the form R,G,B (e.g. 255,0,0).

blockCount - The number of blocks (exons) in the BED line.

blockSizes - A comma-separated list of the block sizes. The number of items in this list should correspond to blockCount.

blockStarts - A comma-separated list of block starts. All of the blockStart positions should be calculated relative to chromStart. The number of items in this list should correspond to blockCount.

B. Overview of All Commands for Creating Exon Data for Chromosome 22

The command lines are given below, and comments are enclosed in `# `.

```
`# relabel chromosome ID`
cut -c4- output_1000G_Exome.v1.bed  > 1KG.exons.bed

`# remove duplicated exon intervals`
sort -V -k1,1 -k2,2 1KG.exons.bed | bedtools merge > 1KG.exons.bash_merged.be
d
`# download vcf file for chromosome 22`
wget http://ftp.1000genomes.ebi.ac.uk/vol1/ftp/data_collections/1000_genomes_
project/release/20190312_biallelic_SNV_and_INDEL/ALL.chr22.shapeit2_integrate
d_snvindels_v2a_27022019.GRCh38.phased.vcf.gz;

`# download index file for chromosome 22`
wget http://ftp.1000genomes.ebi.ac.uk/vol1/ftp/data_collections/1000_genomes_
project/release/20190312_biallelic_SNV_and_INDEL/ALL.chr22.shapeit2_integrate
d_snvindels_v2a_27022019.GRCh38.phased.vcf.gz.tbi;
```

```
`# extract SNVs in exon region for chromosome 22`
bcftools view  --regions-file 1KG.exons.bash_merged.bed --types snps --min-al
leles 2 --max-alleles 2 --include 'FILTER="PASS"'  --output-type z  --output-
file exons_chr22.vcf.gz ALL.chr22.shapeit2_integrated_snvindels_v2a_27022019.
GRCh38.phased.vcf.gz
```

C. Shell Scripting and Software Used for Figure 4

There are two steps to create figure 4. At the first step, we use shell scripting to count the lengths and number of SNVs for each chromosome and for exons in each chromosome; then we save these results to excel files. At the second step, we draw figures in the Excel.

The total length of each chromosome is retrieved from https://www.ncbi.nlm.nih.gov/grc/human/data. For the total sequence length of exons in each chromosome, we use the “awk” command, which is useful in manipulating data and generating report. The command is as follows:

```
awk -v OFS=',' '{arr[$1]+= $3-$2+1}  END {for (i in arr) {print i, arr[i]}}'
1KG.exons.bash_merged.bed |sort -k1,1g > length_exons.csv
```

The first option, “-v OFS=','”, forces the output to be delimited by a comma (“,”). To access values in a column, we prefix the column number with a dollar sign (“$”). As in the command line above, “$1” refers to the first column (chromosome ID). Similarly, “$2” and “$3” refers to the second and the third column, which are the start and end positions of the exon intervals respectively.

In the first curly bracket, “arr[$1]” creates an array of 24 elements for each of the chromosome (e.g chr1 to chr22, chr X and chr Y). Each element stores the total length of exon intervals for each chromosome. “$3-$2+1” calculates the length of each exon interval. This calculation is processed on each row, and the number of each calculation will be added to its corresponding element according to the chromosome ID. Since the array will not be shown automatically after the iteration, “END {for (i in arr) {print i, arr[i]}}” allows users to look at the array output of the form: chromosome ID, sequence length.

Note that the input exon data, “1KG.exons.bash_merged.bed”, is the one that contains unique exon intervals. Then we sort the lengths in the ascending order of the chromosome ID using the “sort” command. The option, “k1,1g” sorts the first column (chromosome ID) in general-numerical order, and then we save the result to “length_exons.csv” in the csv file format.

For a demonstration, the first ten row of “length_exons.csv” is now as the follows:

```
X, 1754987
Y, 140738
1, 4744046
2, 3380247
3, 2649196
4, 1804113
5, 2094357
```

```
6, 2355905
7, 2217707
8, 1573127
```

After this file is created, we manually copy a column for the sequence lengths for each chromosome from the website and paste to the file, and we also delete the first two sex chromosomes.

Next, for calculating the number of SNVs, we need to first download the data file that contains all variants for all chromosomes. We retrieve this at the 1KG website by using the "wget"command as the follows:

```
wget http://ftp.1000genomes.ebi.ac.uk/vol1/ftp/data_collections/1000_genomes_
project/release/20190312_biallelic_SNV_and_INDEL/ALL.wgs.shapeit2_integrated_
snvindels_v2a.GRCh38.27022019.sites.vcf.gz
```

In the data lines of this data file, it only contains the fixed information, which are the first eight columns; in another word, this dataset does not contain the genotype data. However, it has no effect on counting the number of SNVs, and we do not need the genotype data for counting. Now, for counting the total number of SNVs for all chromosomes, we firstly filter our data to contain only the diallelic SNVs using the "view" command in bcftools. Then, we count the number of SNVs for each chromosome by using the "awk" command.

```
`# filter data to SNVs`
bcftools view  --types snps --min-alleles 2 --max-alleles 2 --output-type z
--output-file snv_chr_all.vcf.gz ALL.wgs.shapeit2_integrated_snvindels_v2a.GR
Ch38.27022019.sites.vcf.gz

`# count number of snvs in each chromosome`
bcftools view -H snv_chr_all.vcf.gz | awk -v OFS=',' '{arr[$1]++}  END {for (
i in arr) {print i, arr[i]}}'|sort -k1,1g > num_snv_chr.csv
```

In the code above, most of the options have been explained previously. The only option that we have not explained is in the "awk" command, "arr[$1]++". This is the same as "arr[$1] = arr[$1]+1", and if there is a variant for a chromosome, 1 is added to such element in the array according to its chromosome ID.

Similarly, we filter the SNVs in the exon regions of chromosome, and then apply the "awk" command to count the total number of SNVs in exon for each chromosome.

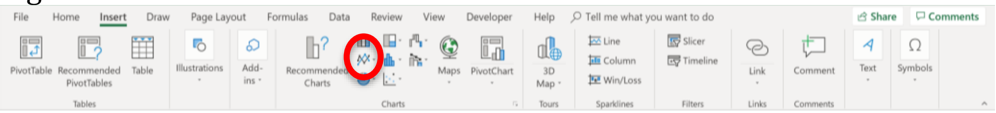
```
`# filter data to SNVs in exon regions of chromosomes`
bcftools view  --regions-file 1KG.exons.bash_merged.bed --types snps --min-al
leles 2 --max-alleles 2 --include 'FILTER="PASS"'  --output-type z  --output-
file snv_exon_all.vcf.gz ALL.wgs.shapeit2_integrated_snvindels_v2a.GRCh38.270
22019.sites.vcf.gz

`# count number of snvs in each chromosome`
bcftools view -H snv_exon_all.vcf.gz | awk -v OFS=',' '{arr[$1]++}  END {for
(i in arr) {print i, arr[i]}}' |sort -k1,1g > num_snv_exon.csv
```

After these two files are created, we manually copy the columns for the number of SNVs in exon region and the number of SNVs in chromosome, and paste to the file we create, in "`length_exons.csv`".

At the last step, we calculate the proportions in excel and follow the steps below to plot these numbers to figure, which is as shown in figure 4. The steps for creating bar graph in excel are:

1. Select the desired dataset. (e.g. The proportion of the total length of the chromosome comprised of exons and the proportion of the number of SNVs in each chromosome that are in exons).
2. Click the Insert tab.
3. Click on the line chart icon at the menu bar which as shown in figure 5.

Figure 5. line chart icon



4. Then select any type of line graph for your own preference.