
Table of Contents

Introduction

About GBC	1.1
Download	1.2
File Format	1.3
API Document	1.4

Usage Guide

Command Line Program	2.1
Compress Genotypes	2.1.1
Program Options	2.1.1.1
Example	2.1.1.2
Algorithm Details	2.1.1.3
Extract Genotypes	2.1.2
Program Options	2.1.2.1
Examples	2.1.2.2
Display Summary Information	2.1.3
Program Options	2.1.3.1
Examples	2.1.3.2
Sort Variants by Coordinates	2.1.4
Program Options	2.1.4.1
Examples	2.1.4.2
Combine Multiple GTBs	2.1.5
Concatenate multiple GTBs	2.1.5.1
Program Options	2.1.5.1.1
Examples	2.1.5.1.2
Merge Multiple GTBs	2.1.5.2
Program Options	2.1.5.2.1
Examples	2.1.5.2.2
Reset the Subject Names	2.1.6
Program Options	2.1.6.1
Examples	2.1.6.2
Prune GTB Tree	2.1.7
Program Options	2.1.7.1

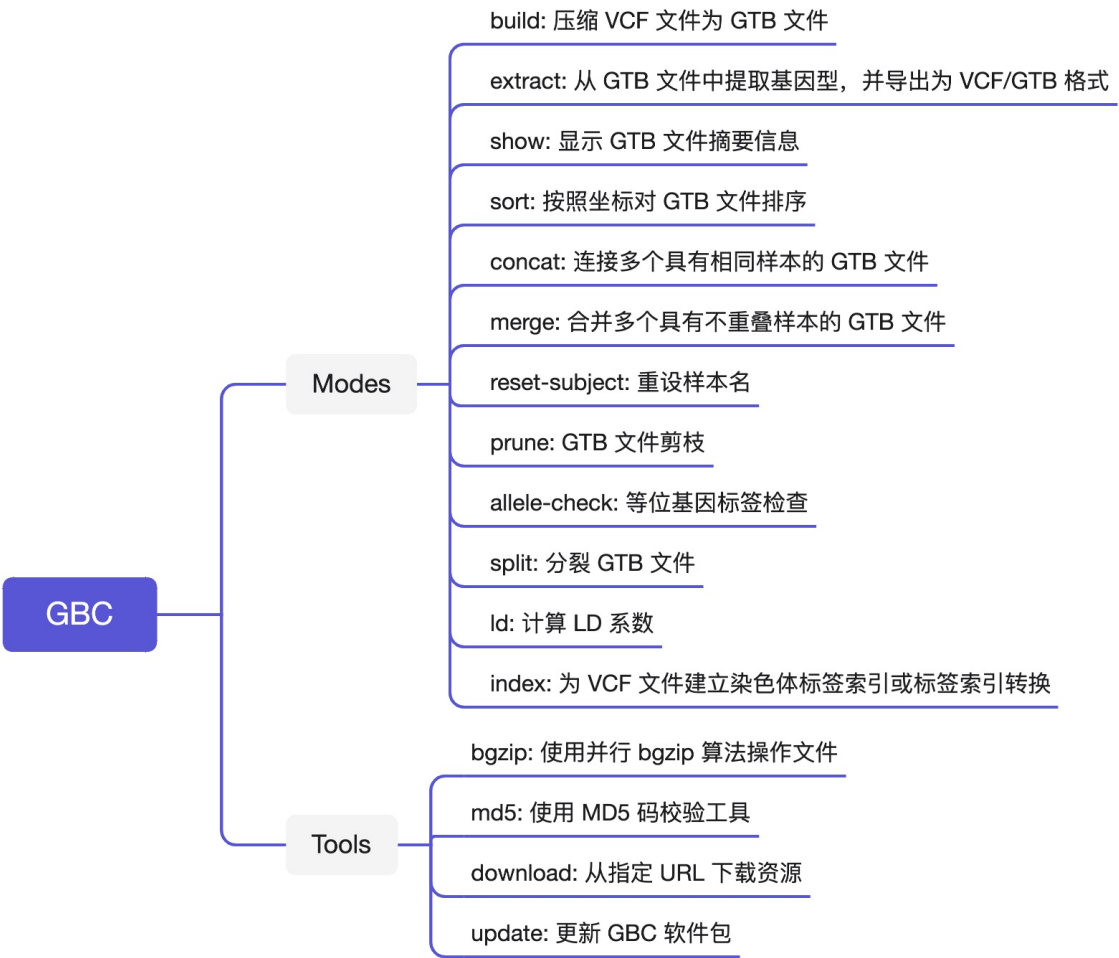
Examples	2.1.7.2
Align Coordinates and Base Labels	2.1.8
Program Options	2.1.8.1
Examples	2.1.8.2
Split GTB	2.1.9
Program Options	2.1.9.1
Examples	2.1.9.2
LD Calculation for GTB	2.1.10
Program Options	2.1.10.1
Examples	2.1.10.2
Set Chromosome Tags	2.1.11
Contig File Format	2.1.11.1
Build Contig File for VCF	2.1.11.2
Reset Chromosome Tags with New Contig File	2.1.11.3

About GBC

GBC (short for GenoType Blocking Compressor) is a blocking compressor for genotype data, which aims at creating a unified and flexible structure-GenoType Block (GTB) for genotype data in the variant call format (VCF) files. There will be a less occupation of hard disk space, a faster data access and extraction function, a more convenient management of population files and a more efficient process of data analysis with the GTB structure compared with the conventional gz format. GBC provides the following functions:

- **Efficient compression:** Linearly increasing time overhead with sample and variant size, stable memory utilization and competitive compression ratio.
 - Memory: Single-threaded compression with < 4 GB memory usage.
 - Speed: up to 78516269 genotypes/s.
- **Quality control:** Quality control at variant-level, genotype-level, population-allele frequency/count-level. The open API facilitates users to customize their own filtering methods.
- **Quick query:** query continuous/random variants, filter variants by allele frequency/count, extract subset of samples, etc;
- **File management:** merge, join, split, subset sample selection, sorting, allele label checking, etc;
- **Complex calculations:** LD calculations.
- **Genotype coding for a wide range of haploid/diploid species.**

GBC is a free standalone toolkit for improved the efficiency of storage and file management of large-scale genotypes. GBC is also a library of fundamental API development tools that can be easily integrated into existing tool flows to accelerate the analysis and computation process of genotype-based data.



[!COMMENT|label:Contact Developer]

Liubin Zhang, suranyi.sysu@gmail.com

Download

GBC is developed based on Oracle JDK 8. It is available on any computer device that supports or is compatible with Oracle JDK 8. Users are required to download and install the [Oracle JDK](#) or [Open JDK](#) firstly. Apple Silicon devices can use the [zulu JDK](#) as an alternative. In addition, we also provide Docker image for building the GBC runtime environment.

Type	URL
Software	http://pmglab.top/gbc/download/gbc.jar
Source Code	https://github.com/Zhangliubin/gbc
Online Manual	http://pmglab.top/gbc/
API docs	http://pmglab.top/gbc/api-docs/
Example Data	http://pmglab.top/gbc/download/example.zip http://pmglab.top/genotypes/#/

Download software package by wget

```
# Download GBC software
wget http://pmglab.top/gbc/download/gbc.jar -O gbc.jar

# setup GBC
java -jar gbc.jar
```

Use the GBC in Docker

```
# download dockerfile
wget http://pmglab.top/gbc/download/Dockerfile -O Dockerfile

# build image
docker build -t gbc .

# setup GBC
docker run -it --rm gbc
```

Download GBC from Github

```
# download source code from github
git clone https://github.com/Zhangliubin/gbc gbc-source

# go to the folder
cd gbc-source

# setup GBC
java -jar gbc.jar
```

Update GBC

GBC has now iterated to a stable release, and generally we only add new features to existing packages, or minor code architecture updates to enhance stability and improve performance. To check the availability of new releases, please use:

```
java -jar gbc.jar update
```

System requirements

GBC has a strict memory requirement control and can usually be run at the default memory allocation for small genomic data. For large scale genomic data, the memory usage of GBC is limited to a maximum of 4 GB for a single thread, therefore, we recommend running GBC programs in a heap memory of no less than 4 GB at all times. The user allocates the GBC runtime heap memory with the following command:

```
java -Xms4g -Xmx4g -jar gbc.jar
```

When running GBC with Docker, we recommend using the following template command:

```
# MacOS or Linux
docker run -v `pwd`:`pwd` -w `pwd` --rm -it -m 4g gbc [options]

# Windows
docker run -v %cd%:%cd% -w %cd% --rm -it -e -m 4g gbc [options]
```

In this command, `-v` means to map the specified host path to the container path (`host_path:container_path`), `-w` means to set the current working path (equivalent to running the `cd path` command), `-it` means to run in an interactive terminal, and `-m 4g` means to set the maximum heap size of the JVM to 4GB.

Updates

[!UPDATE|label:2022/07/01]

- Release the second version of GBC, version number 1.2,
 - Github repository address: <https://github.com/Zhangliubin/gbc>
 - Online Manual: <http://pmglab.top/gbc/>
- Note that GBC-1.1 and GBC-1.2 are fully compatible versions, but GBC-1.2 and later versions will focus on the development of API tools and no longer maintain GUI programs.

[!UPDATE|label:2022/04/02]

- Release the first version of GBC, version number 1.1.
 - Github repository address: <https://github.com/Zhangliubin/Genotype-Blocking-Compressor>
 - Online Manual: <http://pmglab.top/gbc/history/>

Input and Output File Format

GBC supports the conversion between VCF format, compressed VCF format (by BGZIP), and GTB format. In general, we recommend user to use GTB as input and output format, which will get full multi-threaded support.

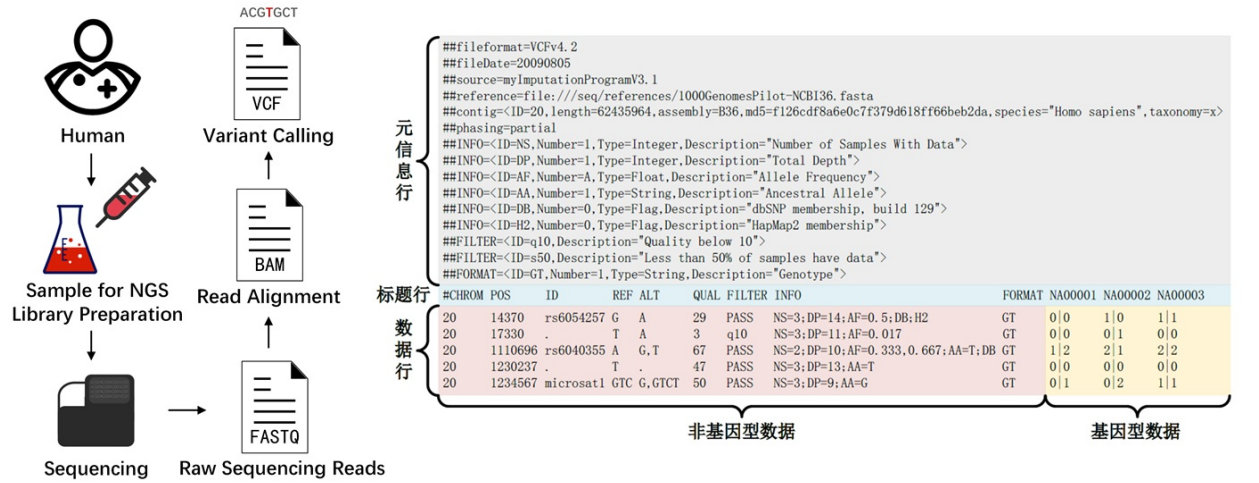
Other Formats \iff VCF / compressed VCF (by BGZIP) \iff GTB

Input Format	Output Format	Tool
Other Formats	VCF or VCF.GZ	Other tools
VCF	compressed VCF	GBC - bgzip --c
compressed VCF (by BGZIP)	VCF	GBC - bgzip --d
compressed VCF (by GZIP)	VCF / compressed VCF (by BGZIP)	GBC - bgzip --d
VCF / compressed VCF (by BGZIP)	GTB	GBC - build
GTB	VCF / compressed VCF (by BGZIP)	GBC - extract

VCF Format

VCF (Variant Call Format) is a standard format for storing variant sites, which is a text format specifically designed for recording and describing variant information such as SNP, InDel, SV, and CNV.

[!NOTE|label:For the details of the VCF file, we recommend reading: <https://samtools.github.io/hts-specs/VCFv4.2.pdf> and https://en.wikipedia.org/wiki/Variant_Call_Format]

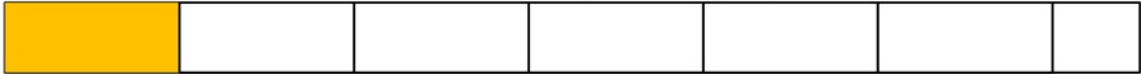


Compressed VCF with BGZIP

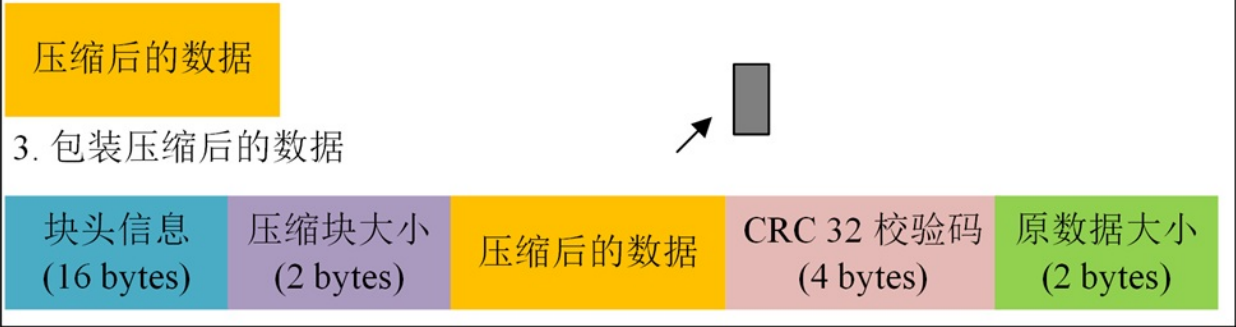
VCF files usually take up a large amount of disk space, so it is common to compress VCFs using BGZIP (the files produced using BGZIP compression are also called BGZF). BGZIP is a block compression method implemented on the standard GZIP file format, which aims to provide a good compression ratio while allowing random access to the data.

a.

1. 将原文件划分为近似 64KB 的块 (65498 bytes)



↓ 2. 每一个块使用 Deflater 算法 (zip 系列的核心算法) 压缩



4. 完成所有压缩后，在末尾追加一个空数据块 (长度为28 bytes)



Java Implementation of Parallel BGZIP Compression

Algorithm: <http://pmglab.top/commandParser/en/example/BGZToolkit.html>

[!NOTE]label>About BGZIP: <http://www.htslib.org/doc/bgzip.html>

GTB Format

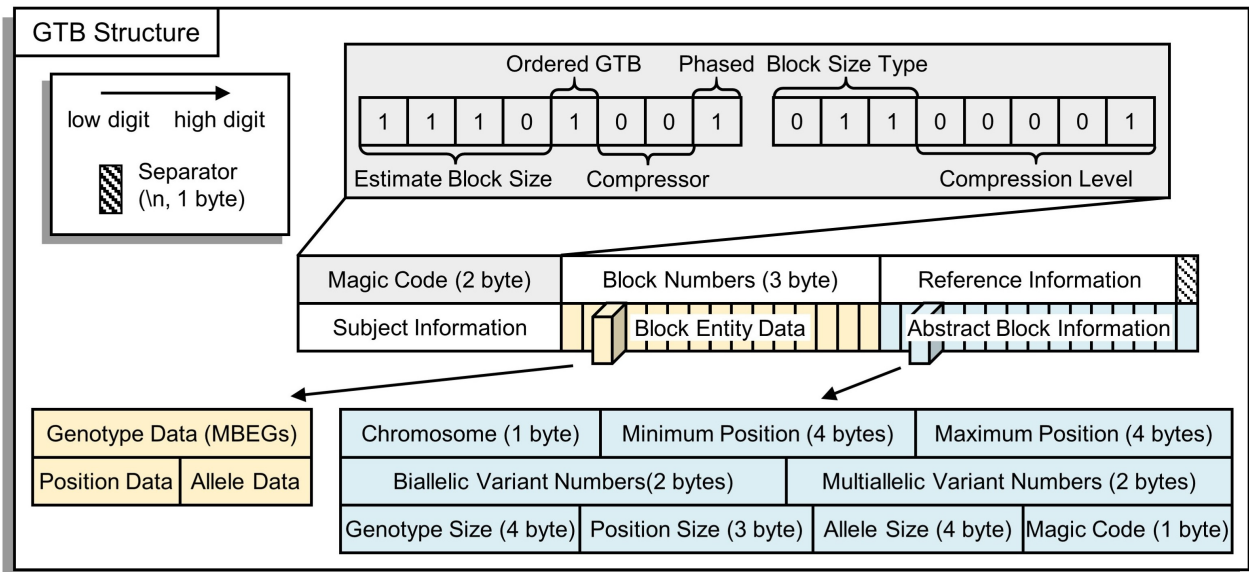
The GTB format is proposed for compressing and storing genotypes data of haploid and diploid species with various allele numbers, chromosome numbers, scales and phased or unphased statuses. It has the following excellent features:

- **Uniform:** In comparison to other file formats (e.g. GTShark, GTC, BGT, PBWT), GTB format has only separate files as output. In addition, it supports the organization of genotypes from multiple chromosomes within a single file (but not for PBWT). This facilitates file storage and transfer.
- **Decoupling:** GTB uses a decoupled design, allowing file modifications to involve as little data as possible. In addition, the independent structure feature makes parallelization computations very easy - parallelization by sites, block or chromosome can be easily implemented.
- **Extensible:** GTB is essentially a double-indexed key (chromosome, position) MDRT format, which is designed with a number of good easy-to-use API methods for fast random access to genotypes. MDRT allows variable multi-indexed keys with a similar structure to GTB, which is also used to store non-genotypic data in VCF files.

The file format of GTB is shown in the figure below. The meaning of each part is as follows:

- **Magic Code:** The first two bytes are used to store the compressed parameters;
- **Block Numbers:** The total number of blocks contained in the compressed file, which also indicates that the "Block Abstract Information" at the end of the file has (25*numbers) bytes of memory;
- **Reference Information:** The version of the reference genome and other reference data;
- **Subjects Information:** List of subjects;
- **Block Entity Data:** The compressed data is combined according to the order of the abstract block information;

- **Abstract Block Information:** Abstract information of the GTB nodes for building first-level fast index table.



Setup GBC

Launch the GBC in the command line program (terminal) with the following command:

```
# The program document is printed by default when no parameters are passed in.
java -Xms4g -Xmx4g -jar gbc.jar

# When passing -h, -help or --help, the program will also print the program document.
java -Xms4g -Xmx4g -jar gbc.jar -h
```

The results are as follows:

```
Usage: java -jar gbc.jar [mode/tool] [options]
Version: GBC-1.2 (last edited on 2022.06.20, http://pmglab.top/gbc)
Mode:
  build      Compress and build *.gtb for vcf/vcf.gz files.
              format: build <input(s)> -o <output> [options]
  extract    Retrieve variants from *.gtb file, and export them to
              (compressed) VCF format or GTB format.
              format: extract <input> -o <output> [options]
  show       Display summary of the GTB File.
              format: show <input> [options]
  sort       Sort variants in GTB by coordinates (chromosome and position).
              format: sort <input> -o <output> [options]
  concat     Concatenate multiple VCF files. All source files must have the
              same subjects columns appearing in the same order with
              entirely different sites, and all files must have to be the
              same in parameters of the status.
              format: concat <input(s)> -o <output> [options]
  merge      Merge multiple GTB files (with non-overlapping subject sets)
              into a single GTB file.
              format: merge <input(s)> -o <output> [options]
  reset-subject Reset subject names (request that same subject number and no
              duplicated names) for gtb file directly. Subject names can be
              stored in a file with ',' delimited form, and pass in via
              '--reset-subject @file'.
              format: reset-subject <input> -o <output> [options]
  prune      Prune GTB files by node-level or chromosome-level.
              format: prune <input> -o <output> [options]
  allele-check Correct for potential complementary strand errors based on
              allele labels (A and C, T and G; only biallelic variants are
              supported).
              format: allele-check <template_input> <input> -o <output>
              [options]
  split      Split a single GTB file into multiple subfiles (e.g. split by
              chromosome).
              format: split <input> -o <output> [options]
  ld         Calculate pairwise the linkage disequilibrium or genotypic
              correlation.
              format: ld <input> -o <output> [options]
  index      Index contig file for specified VCF file or reset contig file
              for specified GTB file.
```

```
format: index <input (VCF or GTB)> -o <output> [options]
```

Tool:

```
bgzip      Use parallel bgzip to compress a single file.
           format: bgzip <string> <string> ...
md5        Calculate a message-digest fingerprint (checksum) for file.
           format: md5 <string> <string> ...
download   Download resources from an URL address.
           format: download <string> <string> ...
update     Update GBC software packages.
           default: ./gbc.jar
           format: update <gbc.jar>
```

Use GBC in interactive mode

When running program commands multiple times with GBC, we recommend doing so in interactive mode (like `ipython`), which provides real-time feedback on program input and reduces the time required to start the JVM (especially when running in Docker, using interactive mode avoids frequent container creation and destruction). To start interactive mode, run:

```
java -Xmx4g -Xms4g -jar gbc.jar -i
```

Of course, the user can also choose to enter interactive mode after running the command once, with the following command:

```
java -Xmx4g -Xms4g -jar gbc.jar [mode/tool] [options] -i
```

When you type a command, you no longer need to specify `java -Xmx4g -Xms4g -jar gbc.jar`. The command line interaction mode has four additional parameters in addition to the parameters in command line mode:

Parameters	Description
exit, q, quit	Exit program, exit the command line interaction mode.
clear	Clearing the screen (actually printing out multiple blank lines).
reset	Clear the data buffer.
Lines begin with "#"	For annotation.

Compress Genotypes

Use the following command to compress the genomic VCF file(s):

```
build <input(s)> -o <output> [options]
```

- GBC help compress the file in compliance with [VCF Specification](#), and all GBC operations are based on the assumption that the file format is compliant with this specification.
- The inputFileNames can be a single .vcf file or .vcf.gz file, and it can also be the path of the folder containing all the files to be compressed. When a folder path is given, the GBC will help filter out all .vcf or .vcf.gz files in this folder (and its sub-folders) for compression.
- The GBC currently only supports compression of the human genome, for chromosome, GBC only supports 1-22, X, Y and 1-22, X, Y with the "chr" prefix (such as chr1, chrY); For other species, please use command `GBC index <input> -o <contigFile>` to build the contig file first, and users should add the command `--contig <contigFile>` when compressing.
- When using GBC to combine and compress multiple files, the files are required to have the same samples (can be disordered). If the sample of a file is a subset of other files, it can also be compressed correctly, and the missing genotypes will be replaced by `.|..`
- When input files are unordered (in coordinates), GBC also compresses them correctly and produces GTB files marked as `unordered`. In general we recommend that users further sort this file using the `sort <input> -o <output> [options]` command, otherwise it will not work for some functions (e.g. calculating LD coefficients) or affect the performance of other operations (e.g. merging VCF files).

Program Options

Usage: build <input(s)> -o <output> [options]

Options:

- `--contig` Specify the corresponding contig file.
default: /contig/human/hg38.p13
format: `--contig <file>` (Exists,File,Inner)
- `*--output, -o` Set the output file.
format: `--output <file>`
- `--threads, -t` Set the number of threads.
default: 4
format: `--threads <int>` (≥ 1)
- `--yes, -y` Overwrite output file without asking.

GTB Archive Options:

- `--phased, -p` Set the status of genotype to phased.
- `--biallelic` Split multiallelic variants into multiple biallelic variants.
- `--simply` Delete the alternative alleles (ALT) with allele counts equal to 0.
- `--blockSizeType, -bs` Set the maximum $\text{size}=2^{(7+x)}$ of each block. (-1 means auto-adjustment)
default: -1
format: `--blockSizeType <int>` (-1 ~ 7)
- `--no-reordering, -nr` Disable the Approximate Minimum Discrepancy Ordering (AMDO) algorithm.

```

--windowSize, -ws      Set the window size of the AMD0 algorithm.
                        default: 24
                        format: --windowSize <int> (1 ~ 131072)
--compressor, -c       Set the basic compressor for compressing processed data.
                        default: ZSTD
                        format: --compressor <string> ([ZSTD/LZMA/GZIP] or
                        [0/1/2] (ignoreCase))
--level, -l            Compression level to use when basic compressor works.
                        (ZSTD: 0~22, 3 as default; LZMA: 0~9, 3 as default;
                        GZIP: 0~9, 5 as default)
                        default: -1
                        format: --level <int> (-1 ~ 31)
--readyParas, -rp      Import the template parameters (-p, -bs, -c, -l) from an
                        external GTB file.
                        format: --readyParas <file> (Exists,File)
--seq-ac               Exclude variants with the alternate allele count (AC)
                        per variant out of the range [minAc, maxAc].
                        format: --seq-ac <int>-<int> (>= 0)
--seq-af               Exclude variants with the alternate allele frequency
                        (AF) per variant out of the range [minAf, maxAf].
                        format: --seq-af <double>-<double> (0.0 ~ 1.0)
--seq-an               Exclude variants with the non-missing allele number (AN)
                        per variant out of the range [minAn, maxAn].
                        format: --seq-an <int>-<int> (>= 0)
--max-allele           Exclude variants with alleles over --max-allele.
                        default: 15
                        format: --max-allele <int> (2 ~ 15)

Quality Control Options:
--no-qc               Disable all quality control methods.
--gty-gq              Exclude genotypes with the minimal genotyping quality (Phred
                        Quality Score) per genotype < gq.
                        default: 20
                        format: --gty-gq <int> (>= 0)
--gty-dp              Exclude genotypes with the minimal read depth per genotype < dp.
                        default: 4
                        format: --gty-dp <int> (>= 0)
--seq-qual            Exclude variants with the minimal overall sequencing quality
                        score (Phred Quality Score) per variant < qual.
                        default: 30
                        format: --seq-qual <int> (>= 0)
--seq-dp              Exclude variants with the minimal overall sequencing read depth
                        per variant < dp.
                        default: 0
                        format: --seq-dp <int> (>= 0)
--seq-mq              Exclude variants with the minimal overall mapping quality score
                        (Mapping Quality Score) per variant < mq.
                        default: 20
                        format: --seq-mq <int> (>= 0)

```

Example

Use the GBC to compress the example file `/example/assoc.hg19.vcf.gz`, and set the following properties:

- Store the genotype as phased.

- Set compressor compression level to 16 (ZSTD).
- Split multiallelic variants into multiple biallelic variants.
- Exclude variants with the MAF (minor allele frequency) < 0.01 .
- Overwrite the output file if it already exists.

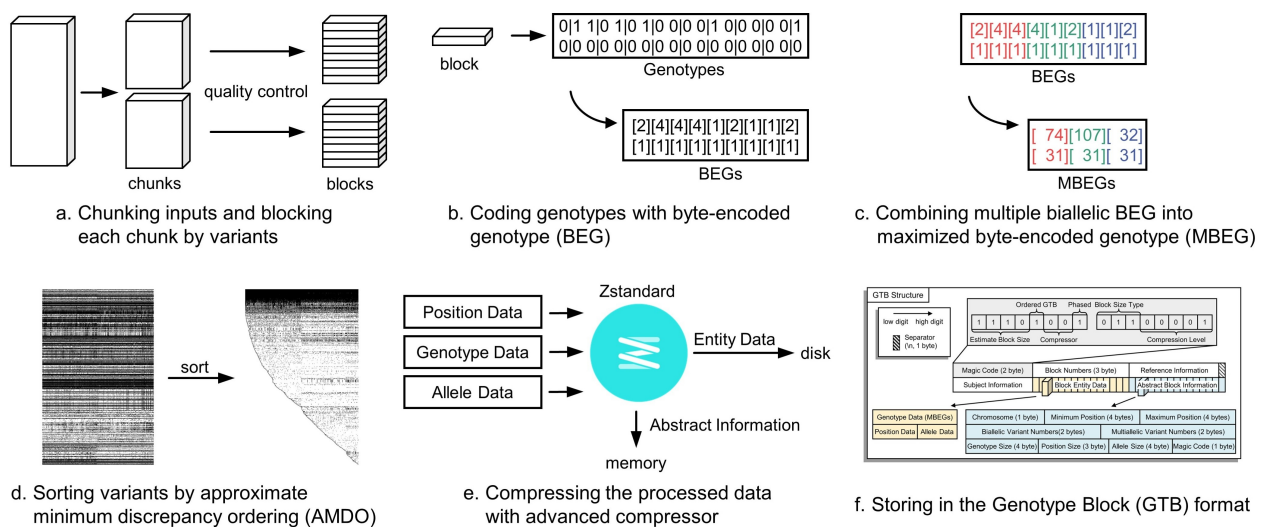
The commands to complete the task are as follows:

```
# Linux or MacOS
docker run -v `pwd`:`pwd` -w `pwd` --rm -it -m 4g gbc \
build ./example/assoc.hg19.vcf.gz -o ./example/assoc.hg19.gtb -p -l 16 --biallelic --seq-af 0.01-0.99 -y

# Windows
docker run -v %cd%:%cd% -w %cd% --rm -it -m 4g gbc build ./example/assoc.hg19.vcf.gz -o \
./example/assoc.hg19.gtb -p -l 16 --biallelic --seq-af 0.01-0.99 -y
```

Algorithm Details

At the beginning of compression, the input genotype file (in VCF format) is divided into several chunks and subsequently processed with multiple threads. In each thread, every chunk is further divided into many smaller blocks. Here, a block is the smallest unit of compression, in which the number of variants is balanced with the sample size given the maximal array length $2^{31} - 1 (\approx 2\text{GB})$ (Fig.a). Then, the genotypes of the variants are encoded into byte codes (Fig.b). For biallelic variants, their byte codes are further combined into one-byte codes by combining three phased or four unphased consecutive genotypes (Fig.c). Next, the approximate minimum discrepancy ordering (AMDO) algorithm is applied on the variant level (Fig.d) to sort the variants with similar genotype distributions for improving the compression ratio. The ZSTD algorithm is then adopted to compress the sorted data in each block (Fig.e). Finally, all the compressed blocks and metadata are written into a single GTB file (Fig.f). The procedure has a linear time complexity regarding the number of subjects and variants with small memory usage (less than 4GB), and it provides almost the fastest compression speed and the most competitive compression ratio to date.



Input VCF File(s) for compression

One or multiple VCF files can be merged into a single GTB file. For a single input file, GBC reads the file directly. For an input of multiple files, GBC first treats the file with the largest sample size as the major file and uses it to build the sample primary indexes. Other input files will be handled in turns after matching the sample indexes of the major file. In the matching process, the genotypes of missing subjects will be set as '.' subsequently to ensure that all the input files can be compressed together consistently in subsequent steps.

[!NOTE|label:Motivation]

Various human genome projects (e.g. 1000GP3, SG10K) store genotypes according to different chromosome tags (i.e. chr1.gz, chr2.gz, ...). In addition, female individuals are usually not included in the Y chromosome files, resulting in Y chromosome files that usually have smaller sample sizes than autosomes or X chromosome files.

This strategy allows for combining and compressing multiple scattered files of the genome, as well as combining and compressing sex chromosome files with autosomal files in the human genome.

Initialization

To control the maximum memory usage per thread, when the total sample size M is determined, the maximum number of variants per GTB block N is also determined. Typically, the block size parameter (or number of variants in a block) corresponding to the sample size range is related as follows:

Parameters	N	M	Parameters	N	M
-bs 7	16384	≤ 65536	-bs 3	1024	≤ 1048576
-bs 6	8192	≤ 131072	-bs 2	512	≤ 2097152
-bs 5	4096	≤ 262144	-bs 1	256	≤ 4194303
-bs 4	2048	≤ 524288	-bs 0	64	≤ 16777215

Chunking Inputs

The input files are compressed sequentially. During compressing each file, it is approximately equally divided into k blocks according to the number of parallel threads k , with each thread processing 1 block. When the input file is in BGZIP-compressed VCF format, the GBC checks the byte-data at the block boundaries and adjusts the pointers to ensure that all variants in the block are intact.

Quality Control by Variant's Non-Genotypic Fields and Genotypic Fields

When processing a single block, the thread reads variant by row and parses the non-genotypic fields. Initial variant-level quality control (e.g., QC based on Phred Quality Score or Mapping Quality Score) is preformed by the non-genotypic fields, and when variant do not meet QC requirements, the thread continues to read the next variant (skipping the current variant).

Variant that satisfy variant-level QC are parsed for their genotypes. When the value of the locus `FORMAT` is `GT` , no genotype-level QC is performed; if the value also contains other key fields, the genotype is quality-controlled for each read (e.g., QC based on DP, GQ). When a genotype does not meet the QC requirements, the genotype is `.` | `.` .

Byte-Encoded of Genotype (BEG)

For a given variant v , the non-missing phased genotype $a \mid b$ will be encoded as:

$$a \mid b \rightarrow \begin{cases} (a+1)^2 - b & , a \geq b \\ b^2 + a + 1 & , a < b \end{cases}$$

(1) For a missing genotype $. \mid .$, it is encoded to 0; (2) For an unphased genotype a/b , it is encoded as above after being transformed to $\min\{a, b\} \mid \max\{a, b\}$; (3) For a genotype a (in a male's chromosome X and Y), it is encoded as above after being converted into $a \mid a$.

Byte-Encoded Table of Genotype $a \mid b$

$a \downarrow$ $b \rightarrow$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	1	2	5	10	17	26	37	50	65	82	101	122	145	170	197
1	4	← 3	6	11	18	27	38	51	66	83	102	123	146	171	198
2	9	← 8	← 7	12	19	28	39	52	67	84	103	124	147	172	199
3	16	← 15	← 14	← 13	20	29	40	53	68	85	104	125	148	173	200
4	25	← 24	← 23	← 22	← 21	30	41	54	69	86	105	126	149	174	201
5	36	← 35	← 34	← 33	← 32	← 31	42	55	70	87	106	127	150	175	202
6	49	← 48	← 47	← 46	← 45	← 44	← 43	56	71	88	107	128	151	176	203
7	64	← 63	← 62	← 61	← 60	← 59	← 58	← 57	72	89	108	129	152	177	204
8	81	← 80	← 79	← 78	← 77	← 76	← 75	← 74	← 73	90	109	130	153	178	205
9	100	← 99	← 98	← 97	← 96	← 95	← 94	← 93	← 92	← 91	110	131	154	179	206
10	121	← 120	← 119	← 118	← 117	← 116	← 115	← 114	← 113	← 112	← 111	132	155	180	207
11	144	← 143	← 142	← 141	← 140	← 139	← 138	← 137	← 136	← 135	← 134	← 133	156	181	208
12	169	← 168	← 167	← 166	← 165	← 164	← 163	← 162	← 161	← 160	← 159	← 158	← 157	182	209
13	196	← 195	← 194	← 193	← 192	← 191	← 190	← 189	← 188	← 187	← 186	← 185	← 184	← 183	210
14	225	← 224	← 223	← 222	← 221	← 220	← 219	← 218	← 217	← 216	← 215	← 214	← 213	← 212	← 211

Write Variant to the Buffer of GTBWriter

When the genotypes of the variant are all encoded, the variant is transferred to the GTBWriter buffer to proceed to the subsequent compression step. In this step, the variant are subjected to additional filtering or conversion operations.

- If the parameter "--simply" is passed in: the ALT tag with an allele count of 0 will be removed.
- If the parameter "--biallelic" is passed in: the multiallelic variant will be split into multiple biallelic variants.
- If the variant-level QC is activated: the variant will be subjected to variant-level quality control and the variant be skipped when it does not meet QC requirements.

Once the GTBWriter buffer size does not reach the maximum number of variants in the GTB block N or the chunk file is not finished reading, repeat the above variant read, quality control, encoding, and write buffer operations. Otherwise, perform the following compression session.

Approximate minimum discrepancy ordering of variants (AMDO)

AMDO starts with extracting the genotype accumulated down-sampling features. Supposing that each block contains M variants and N subjects, a zero-count matrix is denoted as $C = [c_{mn}]_{M \times N}$, where c_{mn} is the count of reference alleles (namely 0 alleles) of the m^{th} variant for the n^{th} subject. Then, the genotype vector of a variant m $C_m = [c_{m0}, c_{m1}, \dots, c_{m(N-1)}]$ is merged into a shorter s -element vector,

$$C_m^{(l)} = [C_{m,0}^{(l)}, C_{m,1}^{(l)}, \dots, C_{m,s-1}^{(l)}]$$

where $C_{m,i}^{(l)}$ covers a maximum of $l = \lceil N/s \rceil$ consecutive genotypes and s is 24 by default. Each element $C_{m,i}^{(l)}$ in the vector is defined as an accumulated count, i.e. :

$$C_{m,i}^{(l)} = \sum_{j=i \cdot l}^{\min\{N-1, (i+1)l-1\}} \sum_{k=i \cdot l}^j c_{mk} = \begin{cases} \sum_{j=i \cdot l}^{(i+1)l-1} ((i+1)l - j) c_{mj} & , i < s - 1 \\ \sum_{j=i \cdot l}^{N-1} (N - j) c_{mj} & , i = s - 1 \end{cases}$$

The accumulation helps discriminate genotype distribution effectively. All the variants in a block are divided into two groups, i.e., the biallelic and the multiallelic groups. In the biallelic variants group, the order of the variant v_i and the biallelic variant v_j is defined as the dictionary order of $C_i^{(l)}$ and $C_j^{(l)}$, which are described as below:

- If $\exists k_0 \in [0, \lceil \frac{N}{s} \rceil - 1]$, $\forall k \in [0, k_0 - 1]$, such that $C_{i,k_0}^{(s)} < C_{j,k_0}^{(s)}$, $C_{i,k}^{(s)} = C_{j,k}^{(s)}$, then $v_i > v_j$;
- If $\exists k_0 \in [0, \lceil \frac{N}{s} \rceil - 1]$, $\forall k \in [0, k_0 - 1]$, such that $C_{i,k_0}^{(s)} > C_{j,k_0}^{(s)}$, $C_{i,k}^{(s)} = C_{j,k}^{(s)}$, then $v_i < v_j$;
- If $\forall k \in [0, \lceil \frac{N}{s} \rceil - 1]$, such that $C_{i,k}^{(s)} = C_{j,k}^{(s)}$, then $v_i = v_j$.

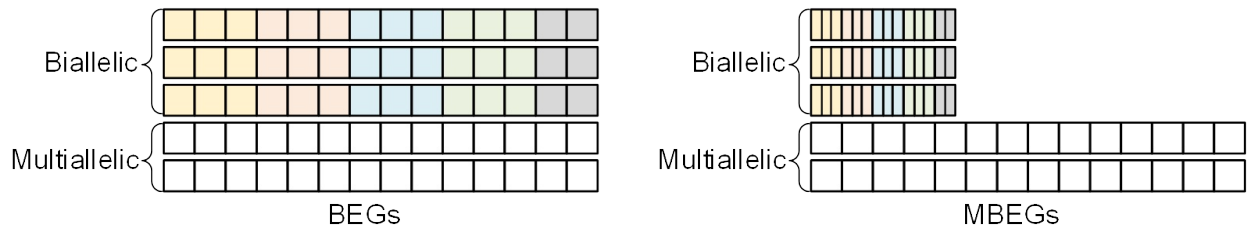
On the contrary, the order will be inverted for multiallelic variants, which helps maximize the length of similar genotype vectors. Finally, the corresponding information of positions, alleles and MBEGs for variants are sorted according to the ordered variants ($I = [m_0, m_1, \dots, m_{M-1}]$).

Maximized Byte-Encoded of Genotype (MBEG)

For biallelic variants, we further combine 3(phased)~4(unphased) multiple consecutive BEG codes into a single byte as follow:

$$\begin{aligned} \text{phased} : [\text{BEG}_0, \text{BEG}_1, \text{BEG}_2] &\rightarrow 5^2 \cdot \text{BEG}_0 + 5 \cdot \text{BEG}_1 + \text{BEG}_2 \\ \text{unphased} : [\text{BEG}_0, \text{BEG}_1, \text{BEG}_2, \text{BEG}_3] &\rightarrow 4^3 \cdot \text{BEG}_0 + 4^2 \cdot \text{BEG}_1 + 4 \cdot \text{BEG}_2 + \text{BEG}_3 \end{aligned}$$

When the number of genotypes at a variant does not constitute a multiple of 3 or 4, the "null genotype" at the end will be set to the same as the previous one, i.e., $\text{BEG}_i = \text{BEG}_{i-1}$ (if $i \geq 1$).



Merge data stream after compressing with advanced compressors

All the sorted MBEG and BEG codes in each block are further compressed by advanced compressors. Popular compression algorithms (e.g., Gzip, LZMA and zlib) can achieve a compression ratio of 100x or more on genotypes. We chose the ZSTD (short for Zstandard[19]) by default because it provides the fastest speed with a similar compression ratio among the widely used compression algorithms. In detail, the byte codes of each variant are concatenated into a byte array B_1 directly. Next, the position of each variant is converted into 4 bytes, and then all variants' positions are concatenated into a byte array B_2 . Finally, the alleles of all variants are concatenated into another byte array B_3 with a '/' delimiter. Then, these concatenated data B_1, B_2 and B_3 are compressed by the latest ZSTD to produce \hat{B}_1, \hat{B}_2 and \hat{B}_3 respectively. The data entity is a long vector composed of three sections of compressed data, including encoded genotypes, positions, alleles. Two types of information of each packed block, including abstract information and data entity, are subsequently written to the GTB file. The abstract information include the chromosome number (1 byte), minimum and maximum positions (4 bytes respectively), number of biallelic variants (2 bytes), number of multiallelic variants (2 bytes), length of \hat{B}_1 (4 bytes), length of \hat{B}_2 (3 bytes), length of \hat{B}_3 (4 bytes) and magic code (1 byte) in a block.

GBC can be integrated with different compression algorithms. ZSTD and LZMA algorithms have been embedded to compress each uncoupled genotype block. We also reserve two types of compressors for developers to extend in the future.

Finish Compression

When a thread completes compression, the block summary information is stitched into the thread's corresponding output file. Then, the file pointer is moved to the file header and the block header information is modified.

When all threads have completed compression, the GBC main program calls the `Concat` method to stitch the GTB files produced by each thread into a single GTB file.

Extract Genotypes

Use the following command to extract genotypes from the GTB:

```
extract <input> -o <output> [options]
```

- If the [options] contains --o-gtb or the output file specified by -o is end with .gtb , the program will output genotypes in GTB format.
- If the [options] contains --o-bgz or the output file specified by -o is end with .gz , the program will output genotypes in BGZIP-compressed VCF format.
- If the [options] contains --o-vcf or the output file specified by -o is not end with .gz or .gtb , the program will output genotypes in VCF format.

In general, bioinformatics tools (such as PLINK) are compatible with the BGZIP-compressed VCF file format, and we recommend that users use the --o-bgz or --o-gtb format as output to enhance the parallel output performance of the program.

Program Options

Usage: extract <input> -o <output> [options]

Output Options:

```
--contig      Specify the corresponding contig file.
               default: /contig/human/hg38.p13
               format: --contig <file> (Exists,File,Inner)

*--output,-o  Set the output file.
               format: --output <file>

--o-text      Output VCF file in text format. (this command will be executed automatically if
               '--o-bgz' or '--o-gtb' is not passed in and the output file specified by '-o' is
               not end with '.gz' or '.gtb')

--o-bgz       Output VCF file in bgz format. (this command will be executed automatically if
               '--o-text' or '--o-gtb' is not passed in and the output file specified by '-o' is
               end with '.gz')

--o-gtb       Output VCF file in gtb format. (this command will be executed automatically if
               '--o-text' or '--o-bgz' is not passed in and the output file specified by '-o' is
               end with '.gtb')

--level,-l    Compression level to use when basic compressor works. (ZSTD: 0~22, 3 as default;
               LZMA: 0~9, 3 as default; BGZIP: 0~9, 5 as default)
               default: -1
               format: --level <int> (-1 ~ 31)

--no-clm      Parallel output is not controlled using the cyclic locking mechanism (CLM). With
               this parameter, parallel output means output to multiple temporary files and
```

```

        finally concatenating them together.
--threads, -t Set the number of threads.
               default: 4
               format: --threads <int> (>= 1)
--phased, -p Force-set the status of the genotype. (same as the GTB basic informatio
n by
               default)
               format: --phased [true/false]
--hideGT, -hg Do not output the sample genotypes (only CHROM, POS, REF, ALT, AC, AN,
AF).
--yes, -y      Overwrite output file without asking.
GTB Archive Options:
--biallelic      Split multiallelic variants into multiple biallelic variants.
--simply         Delete the alternative alleles (ALT) with allele counts equal to
0.
--blockSizeType, -bs Set the maximum size=2^(7+x) of each block. (-1 means auto-adjustment)
               default: -1
               format: --blockSizeType <int> (-1 ~ 7)
--no-reordering, -nr Disable the Approximate Minimum Discrepancy Ordering (AMDO) algo
rithm.
--windowSize, -ws Set the window size of the AMD0 algorithm.
               default: 24
               format: --windowSize <int> (1 ~ 131072)
--compressor, -c Set the basic compressor for compressing processed data.
               default: ZSTD
               format: --compressor <string> ([ZSTD/LZMA/GZIP] or [0/1/2] (ignoreCase))
--readyParas, -rp Import the template parameters (-p, -bs, -c, -l) from an external GTB file.
               format: --readyParas <file> (Exists,File)
Subset Selection Options:
--subject, -s Extract the information of the specified subjects. Subject name can be
stored in a
               file with ',' delimited form, and pass in via '-s @file'.
               format: --subject <string>,<string>,...
--range, -r Extract the information by position range.
               format: --range <chrom>:<minPos>-<maxPos> <chrom>:<minPos>-<maxPos> ..
.
--random        Extract the information by position. (An inputFile is needed here, with
each line
               contains 'chrom,position' or 'chrom position'.
               format: --random <file>
--retain-node   Extract variants in the specified coordinate range of the specified chromosome.
               format: --retain-node <string>:<int>-<int> <string>:<int>-<int> ...
--seq-ac        Exclude variants with the alternate allele count (AC) per variant out
of the range
               [minAc, maxAc].
               format: --seq-ac <int>-<int> (>= 0)
--seq-af        Exclude variants with the alternate allele frequency (AF) per variant
out of the
               range [minAf, maxAf].
               format: --seq-af <double>-<double> (0.0 ~ 1.0)
--seq-an        Exclude variants with the non-missing allele number (AN) per variant out
of the

```

```

        range [minAn, maxAn].
        format: --seq-an <int>-<int> (>= 0)
--max-allele Exclude variants with alleles over --max-allele.
        default: 15
        format: --max-allele <int> (2 ~ 15)

```

Example

Use the GBC to decompress the example file `./example/assoc.hg19.gtb` and set the following properties.

- Store the genotype as unphased.
- Extract the variants with $POS \geq 1000000$.
- Extract the variants with $AF \in [0.4, 0.6]$.
- Extract the genotypes with sample names NA18963,NA18977,HG02401,HG02353,HG02064.

The commands to complete the task are as follows:

```

# Linux or MacOS
docker run -v `pwd`:`pwd` -w `pwd` --rm -it -m 4g gbc \
extract ./example/assoc.hg19.gtb -o ./example/assoc.hg19.extract.vcf \
-p true -r 1:1000000- --seq-af 0.4-0.6 -s NA18963,NA18977,HG02401,HG02353,HG02064 -y

# Windows
docker run -v %cd%:%cd% -w %cd% --rm -it -m 4g gbc extract ./example/assoc.hg19.gtb -o
./example/assoc.hg19.extract.vcf -p true -r 1:1000000- --seq-af 0.4-0.6 -s NA18963,NA18
977,HG02401,HG02353,HG02064 -y

```

Display Summary Information

Use the following command to display summary information from the GTB:

```
show <input> [options]
```

This command is used to quickly access the information of a GTB file and output it to the terminal. `show` mode contains three output functions:

- Output the summary information about the GTB file (such as shape of genotypes, GTBNodes, etc.).
- Output the sample names of the GTB file.
- Output the summary information (e.g., coordinates, allele labels, allele frequency, etc.) for the variants that satisfy the filtering criteria (if specified).

For detailed genotypes and diverse output formats, please use the `extract` mode.

Program Options

Usage: show <input> [options]

Options:

`--contig` Specify the corresponding contig file.
 default: /contig/human/hg38.p13
 format: `--contig <file>` (Exists,File,Inner)

Summary View Options:

`--add-md5` Print the message-digest fingerprint (checksum) for file
 (which may take a long time to calculating for huge files).
`--add-subject` Print subjects names of the GTB file.
`--add-tree` Print information of the GTBTrees (chromosome only by
 default).
`--add-node` Print information of the GTBNodes.
`--full,-f` Print all abstract information of the GTB file (i.e.,
`--list-baseInfo`, `--list-subject`, `--list-node`).

GTB View Options:

`--list-subject-only` Print subjects names of the GTB file only.
`--list-position-only` Print coordinates (i.e., CHROM,POSITION) of the GTB
 file only.
`--list-site-only` Print coordinates, alleles and INFOs (i.e.,
 CHROM,POSITION,REF,ALT,INFO) of the GTB file.

Subset Selection Options:

`--subject,-s` Print the information of the specified subjects. Subject name
 can be stored in a file with ',' delimited form, and pass in
 via `'-s @file'`.
 format: `--subject <string>,<string>,...`
`--range,-r` Print the information by position range.
 format: `--range <chrom>:<minPos>-<maxPos>`
`<chrom>:<minPos>-<maxPos> ...`
`--random` Print the information by position. (An inputFile is needed
 here, with each line contains 'chrom,position' or 'chrom
 position'.
 format: `--random <file>`

```

--retain-node  Print variants in the specified coordinate range of the
                specified chromosome.
                format: --retain-node <string>:<int>-<int>
                <string>:<int>-<int> ...
--seq-ac       Exclude variants with the alternate allele count (AC) per
                variant out of the range [minAc, maxAc].
                format: --seq-ac <int>-<int> (>= 0)
--seq-af       Exclude variants with the alternate allele frequency (AF) per
                variant out of the range [minAf, maxAf].
                format: --seq-af <double>-<double> (0.0 ~ 1.0)
--seq-an       Exclude variants with the non-missing allele number (AN) per
                variant out of the range [minAn, maxAn].
                format: --seq-an <int>-<int> (>= 0)
--max-allele   Exclude variants with alleles over --max-allele.
                default: 15
                format: --max-allele <int> (2 ~ 15)

```

Example

Use the GBC to list the summary information for the GTB file `./example/assoc.hg19.gtb` and set the following properties.

- Print all GTB nodes' abstract information.
- Print the MD5 checksum of the GTB.

The commands to complete the task are as follows:

```

# Linux or MacOS
docker run -v `pwd`:`pwd` -w `pwd` --rm -it -m 500m gbc \
show ./example/assoc.hg19.gtb \
--full --add-md5

# Windows
docker run -v %cd%:%cd% -w %cd% --rm -it -m 500m gbc extract ./example/assoc.hg19.gtb s
how ./example/assoc.hg19.gtb --full --add-md5

```

Here, the terminal prints the following message:

```

Summary of GTB File:
  GTB File Name: /Users/suranyi/Documents/project/GBC/GBC-1.1/example/assoc.hg19.gtb
  GTB File Size: 938.092 KB
  Genome Reference: ftp://ftp.1000genomes.ebi.ac.uk/vol1/ftp/technical/reference/phase
2_reference_assembly_sequence/hs37d5.fa.gz
  MD5 Code: dcb53e6a9844d413e05ea2a95cae7289
  Suggest To BGZF: false
  Phased: true
  Ordered GTB: true
  BlockSize: 16384 (-bs 7)
  Compression Level: 16 (ZSTD)
  Dimension of Genotypes: 1 chromosome, 18339 variants and 983 subjects
  Subject Sequence: HG01583 HG01586 HG01589 HG01593 HG02490 HG02491 HG02493 HG02494 HG0
2597 HG02600
                    HG02601 HG02603 HG02604 HG02648 HG02649 HG02651 HG02652 HG02654 HG02

```


698	HG00699	HG00701	HG00704	HG00705	HG00707	HG00708	HG00717	HG00728	HG00729	HG00
759	HG00766	HG00844	HG00851	HG00864	HG00879	HG00881	HG00956	HG00982	HG01028	HG01
029	HG01031	HG01046	HG01595	HG01596	HG01597	HG01598	HG01599	HG01600	HG01794	HG01
795	HG01796	HG01797	HG01798	HG01799	HG01800	HG01801	HG01802	HG01804	HG01805	HG01
806	HG01807	HG01808	HG01809	HG01810	HG01811	HG01812	HG01813	HG01815	HG01816	HG01
817	HG01840	HG01841	HG01842	HG01843	HG01844	HG01845	HG01846	HG01847	HG01848	HG01
849	HG01850	HG01851	HG01852	HG01853	HG01855	HG01857	HG01858	HG01859	HG01860	HG01
861	HG01862	HG01863	HG01864	HG01865	HG01866	HG01867	HG01868	HG01869	HG01870	HG01
871	HG01872	HG01873	HG01874	HG01878	HG02016	HG02017	HG02019	HG02020	HG02023	HG02
025	HG02026	HG02028	HG02029	HG02031	HG02032	HG02035	HG02040	HG02047	HG02048	HG02
049	HG02050	HG02057	HG02058	HG02060	HG02061	HG02064	HG02069	HG02070	HG02072	HG02
073	HG02075	HG02076	HG02078	HG02079	HG02081	HG02082	HG02084	HG02085	HG02086	HG02
087	HG02088	HG02113	HG02116	HG02121	HG02122	HG02127	HG02128	HG02130	HG02131	HG02
133	HG02134	HG02136	HG02137	HG02138	HG02139	HG02140	HG02141	HG02142	HG02151	HG02
152	HG02153	HG02154	HG02155	HG02156	HG02164	HG02165	HG02166	HG02178	HG02179	HG02
180	HG02181	HG02182	HG02184	HG02185	HG02186	HG02187	HG02188	HG02190	HG02250	HG02
351	HG02353	HG02355	HG02356	HG02360	HG02364	HG02367	HG02371	HG02374	HG02375	HG02
379	HG02380	HG02382	HG02383	HG02384	HG02385	HG02386	HG02389	HG02390	HG02391	HG02
392	HG02394	HG02395	HG02396	HG02397	HG02398	HG02399	HG02401	HG02402	HG02406	HG02
407	HG02408	HG02409	HG02410	HG02512	HG02513	HG02521	HG02522	NA18525	NA18526	NA18
528	NA18530	NA18531	NA18532	NA18533	NA18534	NA18535	NA18536	NA18537	NA18538	NA18
539	NA18541	NA18542	NA18543	NA18544	NA18545	NA18546	NA18547	NA18548	NA18549	NA18
550	NA18552	NA18553	NA18555	NA18557	NA18558	NA18559	NA18560	NA18561	NA18562	NA18
563	NA18564	NA18565	NA18566	NA18567	NA18570	NA18571	NA18572	NA18573	NA18574	NA18
577	NA18579	NA18582	NA18591	NA18592	NA18593	NA18595	NA18596	NA18597	NA18599	NA18
602	NA18603	NA18605	NA18606	NA18608	NA18609	NA18610	NA18611	NA18612	NA18613	NA18
614	NA18615	NA18616	NA18617	NA18618	NA18619	NA18620	NA18621	NA18622	NA18623	NA18
624	NA18625	NA18626	NA18627	NA18628	NA18629	NA18630	NA18631	NA18632	NA18633	NA18

```

634 NA18635      NA18636 NA18637 NA18638 NA18639 NA18640 NA18641 NA18642 NA18643 NA18
644 NA18645      NA18646 NA18647 NA18648 NA18740 NA18745 NA18747 NA18748 NA18749 NA18
757 NA18939      NA18940 NA18941 NA18942 NA18943 NA18944 NA18945 NA18946 NA18947 NA18
948 NA18949      NA18950 NA18951 NA18952 NA18953 NA18954 NA18956 NA18957 NA18959 NA18
960 NA18961      NA18962 NA18963 NA18964 NA18965 NA18966 NA18967 NA18968 NA18969 NA18
970 NA18971      NA18972 NA18973 NA18974 NA18975 NA18976 NA18977 NA18978 NA18979 NA18
980 NA18981      NA18982 NA18983 NA18984 NA18985 NA18986 NA18987 NA18988 NA18989 NA18
990 NA18991      NA18992 NA18993 NA18994 NA18995 NA18997 NA18998 NA18999 NA19000 NA19
001 NA19002      NA19003 NA19004 NA19005 NA19006 NA19007 NA19009 NA19010 NA19011 NA19
012 NA19054      NA19055 NA19056 NA19057 NA19058 NA19059 NA19060 NA19062 NA19063 NA19
064 NA19065      NA19066 NA19067 NA19068 NA19070 NA19072 NA19074 NA19075 NA19076 NA19
077 NA19078      NA19079 NA19080 NA19081 NA19082 NA19083 NA19084 NA19085 NA19086 NA19
087 NA19088      NA19089 NA19090 NA19091

```

Summary of GTB Nodes:

```

└─ Chromosome 1: posRange=[10177, 4999852], numOfNodes=4, numOfVariants=18339
    └─ Node 1: posRange=[10177, 1787378], seek=1157, blockSize=241089, variantNum=4430 (
4430 + 0)
        └─ Node 2: posRange=[1788685, 2914618], seek=242246, blockSize=228815, variantNum=3
998 (3998 + 0)
            └─ Node 3: posRange=[2915041, 4063039], seek=471061, blockSize=249262, variantNum=4
818 (4818 + 0)
                └─ Node 4: posRange=[4063447, 4999852], seek=720323, blockSize=240183, variantNum=5
093 (5093 + 0)

```

Use the GBC to list the variants in the GTB file `./example/assoc.hg19.gtb` that meet the following conditions:

- Print the variants with $POS \in [0, 100000]$.
- Print the variants with $AF \in [0.45, 0.55]$.
- Limit the samples to NA18963,NA18977,HG02401,HG02353,HG02064.

The commands to complete the task are as follows:

```

# Linux or MacOS
docker run -v `pwd`:`pwd` -w `pwd` --rm -it -m 500m gbc \
show ./example/assoc.hg19.gtb \
-r 1:0-100000 --seq-af 0.45-0.55 -s NA18963,NA18977,HG02401,HG02353,HG02064 --list-site
-only

# Windows

```

```
docker run -v %cd%:%cd% -w %cd% --rm -it -m 500m gbc show ./example/assoc.hg19.gtb -r 1
:0-100000 --seq-af 0.45-0.55 -s NA18963,NA18977,HG02401,HG02353,HG02064 --list-site-only
```

Here, the terminal prints the following message:

```
1    15211    T    G    AC=5;AF=0.50000000;AN=10
1    54712    T    TTTTC  AC=5;AF=0.50000000;AN=10
```

Sort Variants by Coordinates

Use the following command to sort variants by coordinate from the GTB:

```
sort <input> -o <output> [options]
```

Generally, VCF files are ordered and this feature can be ignored. When you convert the reference version (e.g., from hg19 to hg38), the coordinates of the variants may change, causing the file to become unordered.

Program Options

Usage: `sort <input> -o <output> [options]`

Options:

- `--contig` Specify the corresponding contig file.
default: /contig/human/hg38.p13
format: `--contig <file>` (Exists,File,Inner)
- `*--output, -o` Set the output file.
format: `--output <file>`
- `--threads, -t` Set the number of threads.
default: 4
format: `--threads <int> (>= 1)`
- `--subject, -s` Extract the information of the specified subjects. Subject name can be stored in a file with ',' delimited form, and pass in via '`-s @file`'.
format: `--subject <string>, <string>, ...`
- `--yes, -y` Overwrite output file without asking.

GTB Archive Options:

- `--phased, -p` Force-set the status of the genotype. (same as the GTB basic information by default)
format: `--phased [true/false]`
- `--biallelic` Split multiallelic variants into multiple biallelic variants.
- `--simply` Delete the alternative alleles (ALT) with allele counts equal to 0.
- `--blockSizeType, -bs` Set the maximum `size=2^(7+x)` of each block. (-1 means auto-adjustment)
default: -1
format: `--blockSizeType <int> (-1 ~ 7)`
- `--no-reordering, -nr` Disable the Approximate Minimum Discrepancy Ordering (AMDO) algorithm.
- `--windowSize, -ws` Set the window size of the AMDO algorithm.
default: 24
format: `--windowSize <int> (1 ~ 131072)`
- `--compressor, -c` Set the basic compressor for compressing processed data.
default: ZSTD
format: `--compressor <string>` ([ZSTD/LZMA/GZIP] or [0/1/2] (ignoreCase))
- `--level, -l` Compression level to use when basic compressor works.
(ZSTD: 0~22, 3 as default; LZMA: 0~9, 3 as default; GZIP: 0~9, 5 as default)
default: -1

```

format: --level <int> (-1 ~ 31)
--readyParas, -rp Import the template parameters (-p, -bs, -c, -l) from an
                  external GTB file.
                  format: --readyParas <file> (Exists,File)
--seq-ac          Exclude variants with the alternate allele count (AC)
                  per variant out of the range [minAc, maxAc].
                  format: --seq-ac <int>-<int> (>= 0)
--seq-af          Exclude variants with the alternate allele frequency
                  (AF) per variant out of the range [minAf, maxAf].
                  format: --seq-af <double>-<double> (0.0 ~ 1.0)
--seq-an          Exclude variants with the non-missing allele number (AN)
                  per variant out of the range [minAn, maxAn].
                  format: --seq-an <int>-<int> (>= 0)
--max-allele      Exclude variants with alleles over --max-allele.
                  default: 15
                  format: --max-allele <int> (2 ~ 15)

```

Example

Use the GBC to compress the unordered VCF file `./example/randomsimu100000V_100S.chr1.vcf.gz` :

```

# Linux or MacOS
docker run -v `pwd`:`pwd` -w `pwd` --rm -it -m 500m gbc \
build ./example/randomsimu100000V_100S.chr1.vcf.gz -o ./example/randomsimu100000V_100S.
chr1.gtb -y

# Windows
docker run -v %cd%:%cd% -w %cd% --rm -it -m 500m gbc build ./example/randomsimu100000V_
100S.chr1.vcf.gz -o ./example/randomsimu100000V_100S.chr1.gtb -y

```

Then, use `show` to print the summary information of `./example/randomsimu100000V_100S.chr1.gtb` :

```

# Linux or MacOS
docker run -v `pwd`:`pwd` -w `pwd` --rm -it -m 500m gbc \
show ./example/randomsimu100000V_100S.chr1.gtb

# Windows
docker run -v %cd%:%cd% -w %cd% --rm -it -m 500m gbc show ./example/randomsimu100000V_1
00S.chr1.gtb

```

Here, the terminal prints the following message, note that this file is an `unordered` file:

```

Summary of GTB File:
  GTB File Name: /Users/suranyi/Documents/project/GBC/GBC-1.1/example/randomsimu100000V_
_100S.chr1.gtb
  GTB File Size: 2.764 MB
  Suggest To BGZF: false
  Phased: false
  Ordered GTB: false
  BlockSize: 16384 (-bs 7)
  Compression Level: 3 (ZSTD)
  Dimension of Genotypes: 1 chromosome, 100000 variants and 100 subjects

```

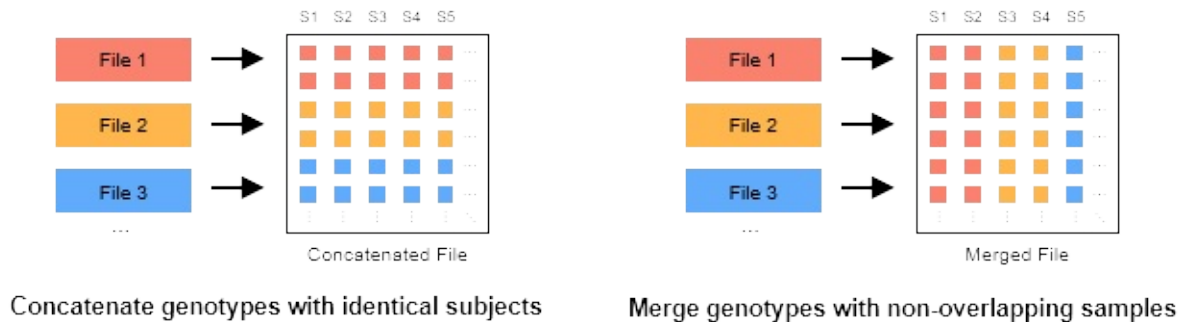
Sort the current GTB file using the `sort` mode:

```
# Linux or MacOS
docker run -v `pwd`:`pwd` -w `pwd` --rm -it -m 500m gbc \
sort ./example/randomsimu100000V_100S.chr1.gtb -o ./example/randomsimu100000V_100S.chr1
.order.gtb

# Windows
docker run -v %cd%:%cd% -w %cd% --rm -it -m 500m gbc sort ./example/randomsimu100000V_1
00S.chr1.gtb -o ./example/randomsimu100000V_100S.chr1.order.gtb
```


Combine Multiple GTBs

There are two main types of combining multiple GTB files: concatenation (concat, left panel) and merge (merge, right panel). The former is usually used to concatenate multiple single chromosome GTB files (e.g. chr1.gtb, chr2.gtb, ...), and the coordinates of these GTB files do not overlap with each other. The latter is usually used to combine genotypes from several different sequencing projects to increase the sample size of the study, and the samples of these GTB files do not overlap with each other.



Concatenate multiple GTBs

Use the following command to concatenate multiple GTBs (left panel):

```
concat <input(s)> -o <output> [options]
```

- When the input multiple files with the same status of genotype, compressor, and the samples, the operation of concatenation will be completed in seconds. Otherwise, GBC needs more time for file pre-conversion.
- The input files can be single or multiple .gtb files, or they can be the path to the folder containing them. When the path is a folder path, GBC will filter all .gtb files in that folder (and its subfolders) for concatenation. Note that GBC only determines the file type based on the file extension, so the correct file extension is the only one that can be concatenated.

Program Options

```
Usage: concat <input(s)> -o <output> [options]
Options:
  --contig          Specify the corresponding contig file.
                    default: /contig/human/hg38.p13
                    format: --contig <file> (Exists,File,Inner)
  *--output,-o      Set the output file.
                    format: --output <file>
  --yes,-y          Overwrite output file without asking.
```

Example

Use the GBC to compress an example file that containing multiple chromosome

```
./example/simu100.coding.vcf.gz :
```

```
# Linux or MacOS
docker run -v `pwd`:`pwd` -w `pwd` --rm -it -m 500m gbc \
build ./example/simu100.coding.vcf.gz -o ./example/simu100.coding.gtb -y

# Windows
docker run -v %cd%:%cd% -w %cd% --rm -it -m 500m gbc build ./example/simu100.coding.vcf
.gz -o ./example/simu100.coding.gtb -y
```

Then, we split the GTB file into multiple single chromosome subfiles according to chromosome tags:

```
# Linux or MacOS
docker run -v `pwd`:`pwd` -w `pwd` --rm -it -m 500m gbc \
split ./example/simu100.coding.gtb -o ./example/simu100.coding --by chromosome

# Windows
docker run -v %cd%:%cd% -w %cd% --rm -it -m 500m gbc split ./example/simu100.coding.gtb
-o ./example/simu100.coding --by chromosome
```

Finally, we use `concat` to concatenate all the GTB files in the `./example/simu100.coding` folder.

```
# Linux or MacOS
docker run -v `pwd`:`pwd` -w `pwd` --rm -it -m 500m gbc \
concat ./example/simu100.coding -o ./example/simu100.coding.concat.gtb

# Windows
docker run -v %cd%:%cd% -w %cd% --rm -it -m 500m gbc concat ./example/simu100.coding -o
./example/simu100.coding.concat.gtb
```

Merge Multiple GTBs

Use the following command to merge multiple GTBs (right panel) from non-overlapping sample sets:

```
merge <input(s)> -o <output> [options]
```

- If the sample names of the files overlap, the sample names need to be reset using the `reset-subject` mode (see: [Reset the Subject Names](#)).
- When no GTB archive format is set, the output GTB archive format defaults to same as the first incoming file.
- The input files can be single or multiple `.gtb` files, or they can be the path to the folder containing them. When the path is a folder path, GBC will filter all `.gtb` files in that folder (and its subfolders) for concatenation. Note that GBC only determines the file type based on the file extension, so the correct file extension is the only one that can be concatenated.

Program Options

```
Usage: merge <input(s)> -o <output> [options]
Options:
  --contig          Specify the corresponding contig file.
                    default: /contig/human/hg38.p13
```

```

format: --contig <file> (Exists,File,Inner)
*--output,-o Set the output file.
format: --output <file>
--threads,-t Set the number of threads.
default: 4
format: --threads <int> (>= 1)
--union Method for handling coordinates in different files (union or
intersection, and intersection is the default), the missing
genotype is replaced by '.'.
--yes,-y Overwrite output file without asking.
GTB Archive Options:
--phased,-p Force-set the status of the genotype. (same as the GTB
basic information by default)
format: --phased [true/false]
--biallelic Split multiallelic variants into multiple biallelic
variants.
--simply Delete the alternative alleles (ALT) with allele counts
equal to 0.
--blockSizeType,-bs Set the maximum size=2^(7+x) of each block. (-1 means
auto-adjustment)
default: -1
format: --blockSizeType <int> (-1 ~ 7)
--no-reordering,-nr Disable the Approximate Minimum Discrepancy Ordering
(AMDO) algorithm.
--windowSize,-ws Set the window size of the AMDO algorithm.
default: 24
format: --windowSize <int> (1 ~ 131072)
--compressor,-c Set the basic compressor for compressing processed data.
default: ZSTD
format: --compressor <string> ([ZSTD/LZMA/GZIP] or
[0/1/2] (ignoreCase))
--level,-l Compression level to use when basic compressor works.
(ZSTD: 0~22, 3 as default; LZMA: 0~9, 3 as default;
GZIP: 0~9, 5 as default)
default: -1
format: --level <int> (-1 ~ 31)
--readyParas,-rp Import the template parameters (-p, -bs, -c, -l) from an
external GTB file.
format: --readyParas <file> (Exists,File)
--seq-ac Exclude variants with the alternate allele count (AC)
per variant out of the range [minAc, maxAc].
format: --seq-ac <int>-<int> (>= 0)
--seq-af Exclude variants with the alternate allele frequency
(AF) per variant out of the range [minAf, maxAf].
format: --seq-af <double>-<double> (0.0 ~ 1.0)
--seq-an Exclude variants with the non-missing allele number (AN)
per variant out of the range [minAn, maxAn].
format: --seq-an <int>-<int> (>= 0)
--max-allele Exclude variants with alleles over --max-allele.
default: 15
format: --max-allele <int> (2 ~ 15)

```

Example

Download the 1000GP3 dataset from <http://pmglab.top/genotypes> and store it in `./example/1000GP3` folder with the following path structure:

```
- 1000GP3
  - AFR
  - AMR
  - EAS
  - EUR
  - SAS
- randomsimu100000V_100S.chr1.vcf.gz
- rare.disease.hg19.vcf.gz
- query.txt
- query_1000GP3.txt
- simu100.coding.vcf.gz
- assoc.hg19.vcf.gz
```

Use the GBC to compress genotype data for each population.

```
# Linux or MacOS
docker run -v `pwd`:`pwd` -w `pwd` --rm -it -m 4g gbc \
build ./example/1000GP3/AFR -o ./example/1000GP3/AFR.gtb -l 16 -p -y
docker run -v `pwd`:`pwd` -w `pwd` --rm -it -m 4g gbc \
build ./example/1000GP3/AMR -o ./example/1000GP3/AMR.gtb -l 16 -p -y
docker run -v `pwd`:`pwd` -w `pwd` --rm -it -m 4g gbc \
build ./example/1000GP3/EAS -o ./example/1000GP3/EAS.gtb -l 16 -p -y
docker run -v `pwd`:`pwd` -w `pwd` --rm -it -m 4g gbc \
build ./example/1000GP3/EUR -o ./example/1000GP3/EUR.gtb -l 16 -p -y
docker run -v `pwd`:`pwd` -w `pwd` --rm -it -m 4g gbc \
build ./example/1000GP3/SAS -o ./example/1000GP3/SAS.gtb -l 16 -p -y

# Windows
docker run -v %cd%:%cd% -w %cd% --rm -it -m 4g gbc build ./example/1000GP3/AFR -o ./example/1000GP3/AFR.gtb -l 16 -p -y
docker run -v %cd%:%cd% -w %cd% --rm -it -m 4g gbc build ./example/1000GP3/AMR -o ./example/1000GP3/AMR.gtb -l 16 -p -y
docker run -v %cd%:%cd% -w %cd% --rm -it -m 4g gbc build ./example/1000GP3/EAS -o ./example/1000GP3/EAS.gtb -l 16 -p -y
docker run -v %cd%:%cd% -w %cd% --rm -it -m 4g gbc build ./example/1000GP3/EUR -o ./example/1000GP3/EUR.gtb -l 16 -p -y
docker run -v %cd%:%cd% -w %cd% --rm -it -m 4g gbc build ./example/1000GP3/SAS -o ./example/1000GP3/SAS.gtb -l 16 -p -y
```

Merge the GTB files of the 5 populations in the `1000GP3` folder:

```
# Linux or MacOS
docker run -v `pwd`:`pwd` -w `pwd` --rm -it -m 4g gbc \
merge ./example/1000GP3/AFR.gtb ./example/1000GP3/AMR.gtb ./example/1000GP3/EAS.gtb ./example/1000GP3/EUR.gtb ./example/1000GP3/SAS.gtb -o ./example/1000GP3.gtb -l 16 -p true -y

# Windows
docker run -v %cd%:%cd% -w %cd% --rm -it -m 4g gbc merge ./example/1000GP3/AFR.gtb ./example/1000GP3/AMR.gtb ./example/1000GP3/EAS.gtb ./example/1000GP3/EUR.gtb ./example/1000GP3/SAS.gtb -o ./example/1000GP3.gtb -l 16 -p true -y
```

```
0GP3/SAS.gtb -o ./example/1000GP3.gtb -l 16 -p true -y
```

Reset the Subject Names

Use the following command to reset subject (or sample) names of GTB:

```
reset-subject <input> -o <output> [options]
```

Program Options

```
Usage: reset-subject <input> -o <output> [options]
Options:
  --contig          Specify the corresponding contig file.
                    default: /contig/human/hg38.p13
                    format: --contig <file> (Exists,File,Inner)
  *--output,-o      Set the output file.
                    format: --output <file>
  --yes,-y          Overwrite output file without asking.
  --subject         Reset subject names (request that same subject number and no
                    duplicated names) for gtb file directly. Subject names can be
                    stored in a file with ',' delimited form, and pass in via
                    '--subject @file'.
                    format: --subject <string>,<string>,...
  --prefix          Use the format `[prefix][number][suffix]` to reset the subject
                    names.
                    default: S_
                    format: --prefix <string>
  --suffix          Use the format `[prefix][number][suffix]` to reset the subject
                    names.
                    format: --suffix <string>
  --begin           Use the format `[prefix][number][suffix]` to reset the subject
                    names.
                    default: 1
                    format: --begin <int>
```

Example

Use the GBC to compress an example file `./example/rare.disease.hg19.vcf.gz` :

```
# Linux or MacOS
docker run -v `pwd`:`pwd` -w `pwd` --rm -it -m 4g gbc \
build ./example/rare.disease.hg19.vcf.gz -o ./example/rare.disease.hg19.gtb -y

# Windows
docker run -v %cd%:%cd% -w %cd% --rm -it -m 4g gbc build ./example/rare.disease.hg19.vcf.gz -o ./example/rare.disease.hg19.gtb -y
```

Reset the subject names to `CASE_6_1` , `CASE_7_1` and `CASE_8_1` :

```
# Linux or MacOS
```

```
docker run -v `pwd`:`pwd` -w `pwd` --rm -it -m 4g gbc \
reset-subject ./example/rare.disease.hg19.gtb -o ./example/out.gtb --prefix CASE_ --begin 6 --suffix _1 -y
```

Windows

```
docker run -v %cd%:%cd% -w %cd% --rm -it -m 4g gbc reset-subject ./example/rare.disease
.hg19.gtb -o ./example/out.gtb --prefix CASE_ --begin 6 --suffix _1 -y
```

or:

Linux or MacOS

```
docker run -v `pwd`:`pwd` -w `pwd` --rm -it -m 4g gbc \
reset-subject ./example/rare.disease.hg19.gtb -o ./example/out.gtb --subject CASE_6_1,CASE_7_1,CASE_8_1 -y
```

Windows

```
docker run -v %cd%:%cd% -w %cd% --rm -it -m 4g gbc reset-subject ./example/rare.disease
.hg19.gtb -o ./example/out.gtb --subject CASE_6_1,CASE_7_1,CASE_8_1 -y
```

Prune GTB Tree

Use the following command to prune the GTB Tree:

```
prune <input> -o <output> [options]
```

Compared with `extract`, `prune` does not need to decompress any data for node extraction or deletion, which is faster and less memory-intensive, and all operations can be completed in seconds.

Program Options

```
Usage: prune <input> -o <output> [options]
Options:
  --contig          Specify the corresponding contig file.
                    default: /contig/human/hg38.p13
                    format: --contig <file> (Exists,File,Inner)
  *--output, -o     Set the output file.
                    format: --output <file>
  --yes, -y         Overwrite output file without asking.
  --delete-node     Delete the specified GTBNodes.
                    format: --delete-node <string>:<int>,<int>,... <string>:<int>,<int>,..
  . ...
  --retain-node     Retain the specified GTBNodes.
                    format: --retain-node <string>:<int>,<int>,... <string>:<int>,<int>,..
  . ...
  --delete-chrom    Delete the specified Chromosomes.
                    format: --delete-chrom <string>,<string>,...
  --retain-chrom    Retain the specified Chromosomes.
                    format: --retain-chrom <string>,<string>,...
```

Example

Use the GBC to extract the sex chromosomes (chrX and chrY) of `1000GP3.gtb`.

```
# Linux or MacOS
docker run -v `pwd`:`pwd` -w `pwd` --rm -it -m 4g gbc \
prune ./example/1000GP3.gtb -o ./example/1000GP3.chrXY.gtb \
--retain-chrom X,Y \
-y

# Windows
docker run -v %cd%:%cd% -w %cd% --rm -it -m 4g gbc prune ./example/1000GP3.gtb -o ./example/1000GP3.chrXY.gtb --retain-chrom X,Y -y
```

View the summary information of extracted GTB file:

```
# Linux or MacOS
```



```
docker run -v `pwd`:`pwd` -w `pwd` --rm -it -m 4g gbc \
show ./example/1000GP3.chrXY.gtb --add-tree

# Windows
docker run -v %cd%:%cd% -w %cd% --rm -it -m 4g gbc show ./example/1000GP3.chrXY.gtb --a
dd-tree
```

Here, the terminal prints the following message:

```
Summary of GTB File:
  GTB File Name: /Users/suranyi/Documents/project/GBC/GBC-1.1/example/1000GP3.chrXY.gtb
  GTB File Size: 66.759 MB
  Genome Reference: ftp://ftp.1000genomes.ebi.ac.uk/vol1/ftp/technical/reference/phase
2_reference_assembly_sequence/hs37d5.fa.gz
  Suggest To BGZF: false
  Phased: true
  Ordered GTB: true
  BlockSize: 16384 (-bs 7)
  Compression Level: 16 (ZSTD)
  Dimension of Genotypes: 2 chromosomes, 3530137 variants and 2504 subjects

Summary of GTB Nodes:
├─ Chromosome X: posRange=[60020, 155260478], numOfNodes=212, numOfVariants=3468095
└─ Chromosome Y: posRange=[2655180, 28770931], numOfNodes=4, numOfVariants=62042
```

Align Coordinates and Base Labels

When merging genotypes from different batches, the variants that with the same coordinate may introduce confusion (e.g., batch 1 uses forward-stranded DNA and batch 2 uses reverse-stranded DNA). With the same coordinates and base complementarity (e.g., [A, T] and [C, G]), GBC designs three functions to identify inconsistent allele labels:

- Check for allele frequency: the difference between the allele frequency of variant 1 and the allele frequency of variant 2 is less than the threshold (i.e., $|AF_1 - AF_2| < \text{gap}$). This will work for variants with minor allele frequencies much less than 0.5 (say 0.3).
- Check for allele count: 2×2 column tables were constructed using the number of reference alleles at a variant of two batches to be merged. The chi-square tests were performed. If the hypothesis test rejects the H_0 hypothesis (i.e., the allele frequencies of the two variants are identical), then the variants in different batches cannot be considered potentially identical. Note that this will not be suitable for the scenario that the two batches are used for cases and controls, respectively.
- Check for LD pattern: identifies other variants in which the absolute value of the genotypic correlation is over a threshold (say, 0.8) in two batches separately. We then count the positive signs of the correlation coefficients in the two batches. If the numbers of signs are very different between the two batches, the allele labels should be flipped; otherwise, the allele labels are not flipped. This function can be used for variants with minor allele frequencies close to 0.5.

Use the following command to correct for potential complementary strand errors:

```
allele-check <template_input> <input> -o <output> [options]
```

Program Options

```
Usage: allele-check <template_input> <input> -o <output> [options]
```

Options:

```
--contig          Specify the corresponding contig file.
                   default: /contig/human/hg38.p13
                   format: --contig <file> (Exists,File,Inner)
*--output,-o      Set the output file.
                   format: --output <file>
--threads,-t      Set the number of threads.
                   default: 4
                   format: --threads <int> (>= 1)
--union           Method for handling coordinates in different files (union or
                   intersection, and intersection is the default), the missing
                   genotype is replaced by '.'.
--yes,-y          Overwrite output file without asking.
```

Alignment Coordinate Options:

```
--p-value         Correct allele labels of rare variants (minor allele
                   frequency < --maf) with the p-value of chi^2 test >=
                   --p-value.
                   default: 0.05
                   format: --p-value <double> (1.0E-6 ~ 0.5)
--freq-gap        Correct allele labels of rare variants (minor allele
                   frequency < --maf) with the allele frequency gap <=
```

```

--freq-gap.
format: --freq-gap <double> (1.0E-6 ~ 0.5)
--no-ld      By default, correct allele labels of common variants
              (minor allele frequency >= --maf) using the ld pattern
              in different files. Disable this function with option
              '--no-ld'.
--min-r      Exclude pairs with genotypic LD correlation |R| values
              less than --min-r.
              default: 0.8
              format: --min-r <double> (0.5 ~ 1.0)
--flip-scan-threshold Variants with flipped ld patterns (strong correlation
              coefficients of opposite signs) that >= threshold
              ratio will be corrected.
              default: 0.8
              format: --flip-scan-threshold <double> (0.5 ~ 1.0)
--maf        For common variants (minor allele frequency >= --maf)
              use LD to identify inconsistent allele labels.
              default: 0.05
              format: --maf <double> (0.0 ~ 0.5)
--window-bp, -bp The maximum number of physical bases between the
              variants being calculated for LD.
              default: 10000
              format: --window-bp <int> (>= 1)

GTB Archive Options:
--phased, -p   Force-set the status of the genotype. (same as the GTB
              basic information by default)
              format: --phased [true/false]
--biallelic    Split multiallelic variants into multiple biallelic
              variants.
--simply       Delete the alternative alleles (ALT) with allele counts
              equal to 0.
--blockSizeType, -bs Set the maximum size=2^(7+x) of each block. (-1 means
              auto-adjustment)
              default: -1
              format: --blockSizeType <int> (-1 ~ 7)
--no-reordering, -nr Disable the Approximate Minimum Discrepancy Ordering
              (AMDO) algorithm.
--windowSize, -ws Set the window size of the AMDO algorithm.
              default: 24
              format: --windowSize <int> (1 ~ 131072)
--compressor, -c Set the basic compressor for compressing processed data.
              default: ZSTD
              format: --compressor <string> ([ZSTD/LZMA/GZIP] or
              [0/1/2] (ignoreCase))
--level, -l    Compression level to use when basic compressor works.
              (ZSTD: 0~22, 3 as default; LZMA: 0~9, 3 as default;
              GZIP: 0~9, 5 as default)
              default: -1
              format: --level <int> (-1 ~ 31)
--readyParas, -rp Import the template parameters (-p, -bs, -c, -l) from an
              external GTB file.
              format: --readyParas <file> (Exists,File)
--seq-ac       Exclude variants with the alternate allele count (AC)
              per variant out of the range [minAc, maxAc].
              format: --seq-ac <int>-<int> (>= 0)
--seq-af       Exclude variants with the alternate allele frequency

```

```

(AF) per variant out of the range [minAf, maxAf].
format: --seq-af <double>-<double> (0.0 ~ 1.0)
--seq-an      Exclude variants with the non-missing allele number (AN)
               per variant out of the range [minAn, maxAn].
               format: --seq-an <int>-<int> (>= 0)
--max-allele  Exclude variants with alleles over --max-allele.
               default: 15
               format: --max-allele <int> (2 ~ 15)

```

Example

Download the `EAS hg38` dataset from `http://pmglab.top/genotypes` and use the dataset as a template file to examine the allele tags of the local exome sequencing variants `SNP.gtb` :

```

# Linux or MacOS
docker run -v `pwd`:`pwd` -w `pwd` --rm -it -m 4g gbc \
allele-check ./example/EAS.gtb ./example/SNP.gtb -o ./example/SNP.checked.gtb --seq-af
0.000001-0.999999 -y

# Windows
docker run -v %cd%:%cd% -w %cd% --rm -it -m 4g gbc allele-check ./example/EAS.gtb ./exa
mple/SNP.gtb -o ./example/SNP.checked.gtb --seq-af 0.000001-0.999999 -y

```

Here, the terminal prints the following message:

```

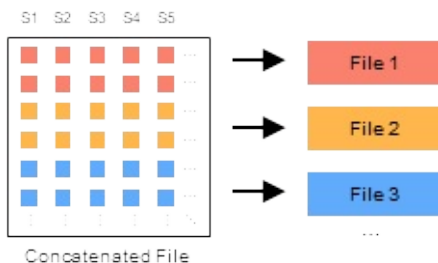
2022-06-27 01:16:02 INFO [ThreadPool-thread-1] AlleleCheck chr11:244961 TempREF=G Temp
ALT=C TempAF=0.8035714285714286 REF=G ALT=C AF=0.6944444444444444 -> REF=C ALT=G
2022-06-27 01:16:02 INFO [ThreadPool-thread-1] AlleleCheck chr11:251057 TempREF=C Temp
ALT=G TempAF=0.08630952380952381 REF=C ALT=G AF=0.22093023255813954 -> REF=G ALT=C
2022-06-27 01:16:04 INFO [ThreadPool-thread-1] AlleleCheck chr17:42969194 TempREF=C Te
mpALT=G TempAF=0.9742063492063492 REF=C ALT=G AF=0.02608695652173913 -> REF=G ALT=C
2022-06-27 01:16:06 INFO [ThreadPool-thread-2] AlleleCheck chr1:1041823 TempREF=G Temp
ALT=C TempAF=0.9990079365079365 REF=G ALT=C AF=0.0045045045045045045 -> REF=C ALT=G

```

Split GTB

Use the following command to split the GTB file into multiple independent subfiles:

```
split <input> -o <output> [options]
```



Program Options

Usage: split <input> -o <output> [options]

Options:

- contig Specify the corresponding contig file.
default: /contig/human/hg38.p13
format: --contig <file> (Exists,File,Inner)
- *--output, -o Set the output folder.
format: --output <file>
- by Split input files by node-level/chromosome-level into multiple subfiles, which can be rejoined by the concat mode.
default: chromosome
format: --by <string> ([chromosome/node] or [0/1])
- yes, -y Overwrite output file without asking.

Example

Split `./example/1000GP3.gtb` by chromosome tags:

```
# Linux or MacOS
docker run -v `pwd`:`pwd` -w `pwd` --rm -it -m 500m gbc \
split ./example/1000GP3.gtb -o ./example/1000GP3-chr -y

# Windows
docker run -v %cd%:%cd% -w %cd% --rm -it -m 500m gbc split ./example/1000GP3.gtb -o ./example/1000GP3-chr -y
```

After running this command, 24 GTB subfiles are generated in the folder `./example/1000GP30-chr`.

LD Calculation for GTB

GBC integrates the calculation of LD coefficients. Use the following command to calculate LD coefficients for GTB file:

```
ld <input> -o <output> [options]
```

Program Options

Usage: ld <input> -o <output> [options]

Output Options:

- contig Specify the corresponding contig file.
 default: /contig/human/hg38.p13
 format: --contig <file> (Exists,File,Inner)
- *--output,-o Set the output file.
 format: --output <file>
- o-text Output LD file in text format. (this command will be executed
 automatically if '--o-bgz' is not passed in and the output file
 specified by '-o' is not end with '.gz')
- o-bgz Output LD file in bgz format. (this command will be executed
 automatically if '--o-text' is not passed in and the output
 file specified by '-o' is end with '.gz')
- level,-l Set the compression level. (Execute only if --o-bgz is passed
 in)
 default: 5
 format: --level <int> (0 ~ 9)
- threads,-t Set the number of threads.
 default: 4
 format: --threads <int> (>= 1)
- yes,-y Overwrite output file without asking.

LD Calculation Options:

- hap-ld,--hap-r2 Calculate pairwise the linkage disequilibrium.
- geno-ld,--gene-r2 Calculate pairwise the genotypic correlation.
- window-bp,-bp The maximum number of physical bases between the
 variants being calculated for LD.
 default: 10000
 format: --window-bp <int> (>= 1)
- min-r2 Exclude pairs with R2 values less than --min-r2.
 default: 0.2
 format: --min-r2 <double> (0.0 ~ 1.0)
- maf Exclude variants with the minor allele frequency (MAF)
 per variant < maf.
 default: 0.05
 format: --maf <double> (0.0 ~ 0.5)
- subject,-s Calculate the LD for the specified subjects. Subject
 name can be stored in a file with ',' delimited form,
 and pass in via '-s @file'.
 format: --subject <string>,<string>,...
- range,-r Calculate the LD by specified position range.
 format: --range <chrom>:<minPos>-<maxPos>
 <chrom>:<minPos>-<maxPos> ...

Example

Calculate the LD coefficients for the `EAS hg38` dataset.

```
# Linux or MacOS
```

```
docker run -v `pwd`:`pwd` -w `pwd` --rm -it -m 4g gbc \
ld ./example/EAS.gtb -o ./example/EAS.ld.gz -t 8 -y
```

```
# Windows
```

```
docker run -v %cd%:%cd% -w %cd% --rm -it -m 4g gbc ld ./example/EAS.gtb -o ./example/EA
S.ld.gz -t 8 -y
```

Contig File Format

The GBC application does not explicitly define the index of chromosome tags internally, but declares it through the contig file, where the index of the chromosome located in the first line of the contig file is 0, and the index of the chromosome in the second line is 1... Therefore, you can easily extend or modify the contig file for chromosome ploidy and chromosome tags modification.

By default, GBC supports genotype compression of human beings. It can also encode and compress genotypes of other haplotypic and diploid species. For non-human genomes, GBC only requires a different contig file to declare the label of the assigned chromosome (e.g., chrX, chrY, chrMT) and its ploidy. A contig file has "`#chromosome,ploidy,length`" as the header line, and then each line represents one chromosome. Since only 1 byte is reserved for storing chromosome numbers in GTB format, we require that the number of chromosomes in the input contig file does not exceed 256.

```
##reference=https://www.ncbi.nlm.nih.gov/grc/human/data?asm=GRCh38.p13
#chromosome,ploidy,length
1,2,248956422
2,2,242193529
3,2,198295559
4,2,190214555
5,2,181538259
6,2,170805979
7,2,159345973
8,2,145138636
9,2,138394717
10,2,133797422
11,2,135086622
12,2,133275309
13,2,114364328
14,2,107043718
15,2,101991189
16,2,90338345
17,2,83257441
18,2,80373285
19,2,58617616
20,2,64444167
21,2,46709983
22,2,50818468
X,2,156040895
Y,2,57227415
MT,2,4485509
```

Build Contig File For VCF

For non-human genotype files, use the following command to build a contig file for the original VCF file.

```
index <input> -o <output> [options]
```