# Standardised and efficient querying of GWAS summary statistics by adapting the VCF format

Matthew Lyon1,2\*, Shea J Andrews3\*, Ben Elsworth2, Tom Gaunt1,2, Gibran Hemani2¥, Edoardo Marcora3¥

\* Joint first author

¥ Joint last author

Correspondence to Matthew Lyon (matt.lyon@bristol.ac.uk)

1. National Institute for Health Research Bristol Biomedical Research Centre
2. Medical Research Council (MRC) Integrative Epidemiology Unit (IEU), Bristol Medical School (Population Health Sciences), University of Bristol, Bristol, UK
3. Ronald M. Loeb Center for Alzheimer’s disease, Department of Neuroscience, Icahn School of Medicine at Mount Sinai, New York, NY, USA

## Abstract

TO DO

## Introduction

The genome-wide association study (GWAS) is a powerful tool for identifying genetic loci associated with traits, diseases and molecular phenotypes such as gene expression and biomarker concentration [1]. Sharing of non-identifiable test summary statistics (i.e. variant, effect size, standard error, p-value etc) has enabled a range of important secondary research applications including gene prioritization [2], causal inference (Mendelian randomization; MR) [3], risk prediction [1], genetic correlation [4] and heritability estimation [5]. However, the utility of summary statistics is hampered by the absence of a universal storage format and associated tools for scalable querying and unambiguous distribution.

Historic lack of a common standard has led to GWAS analysis tools outputting results in different formats (e.g. plink [6], GCTA [7], BOLT-LMM [8], GEMMA [9] and meta-analysis tools e.g. METAL [10]). As a consequence, various processing issues are typically encountered during secondary analysis. First, inconsistency and ambiguity in which allele relates to the effect size estimate. Confusion of the effect allele can have disastrous consequences on interpretation of findings, for example MR studies may provide causal estimates with incorrect effect directionality [11]. Second, summary statistics are provided with different content and field names. Missing data limits potential analyses. Although approaches exist to estimate some values (i.e. standard error from P value) these methods introduce imprecision, thus reducing power. Varying field names are easily addressed in principle but can be cumbersome and error prone. Third, data are frequently distributed with no or insufficient metadata describing the trait, measurement units, publication source etc which can lead to errors and impedes reproducibility. Fourth, plain text is slow and memory inefficient to query making potential applications computationally infeasible i.e. hypothesis-free analyses.

Some proposals for a standard format have been made. The EBI-NHGRI GWAS catalog developed a text format with uniformed column names [12]. The SMR tool [13] introduced a binary format for rapid querying of quantitative trait loci. These approaches are adequate for storing variant level summary statistics but do not enforce allele consistency or have sufficient capacity for essential metadata. Learning from these examples, our own experiences processing GWAS data across two research centers and considering future needs i.e. GWAS of rare variants using genome sequencing, we identified a set of requirements for a suitable universal format.

We determined that adapting the widely used variant call format (VCF) [14], [15] was a convenient and constructive approach to meet these requirements. Here we outline the implementation, explain how it meets these requirements, describe existing and new software that creates and connects to the data format, and show results of query performance. Finally, we provide access to over 10,000 complete GWAS summary datasets that have been converted to this format as part of the IEU GWAS database, and are freely available for download: <https://gwas.mrcieu.ac.uk>.

## Method

### Specification

We have identified a set of useful features that a universal format for storing GWAS summary data should have (Table 1). This specification arises from experience of collecting and harmonising GWAS summary data and performing a range of representative high throughput analysis on these data (for example LD score regression, Mendelian randomisation, genetic colocalization analysis and polygenic risk scores).

### Implementation

The VCF format is organized into three units: flexible file header containing meta-data (lines beginning with ‘#’), variant information (one locus per row) and sample information (one sample per column). We adapt the format such that each sample data column represents GWAS of a single trait (Figure 1).

Meta-data define important characteristics of the GWAS: trait description and units, genome build, number of variants, type of trait (continuous or case/control), sample size and study identifier. The VCF header is also mandatory for defining fields used in the file body including variable description, value requirements (i.e. number of values permitted and null values) and data type (i.e. string, number and boolean).

Each row of the file body contains a single variant position including contig (chromosome) name, base-pair position, variant identifier (i.e. dbSNP identifier), reference (major/non-effect allele) and alternative (minor/effect allele) alleles. The sample column is used to store allele-trait association metrics: marker identifier, allele frequency, regression coefficient, standard error and association P-value.

The full specification provides detailed information including reserved keys: <https://github.com/MRCIEU/gwas_vcf_spec>.

### Gwas2VCF

To automate mapping tabular GWAS summary statistics to VCF, we developed open-source Python3 software that may be run natively on a UNIX machine or using Docker (gwas2vcf; <https://github.com/MRCIEU/gwas2vcf>). First, the application reads in GWAS data using a user-defined file schema. As a minimum chromosome names and position, alleles, effect size, standard error and association P value are required. Second, each variant is aligned to a supplied reference genome file (FASTA) and harmonized to ensure the reference allele matches and the effect allele is non-reference. Reference files are easily obtained from Ensembl or UCSC. Third, records are sorted karyotypically and written along with metadata to a single compressed VCF file. Finally, the VCF is indexed using tabix [16] and rsidx [17] which enable rapid queries by genomic position and dbSNP identifier, respectively.

Dockerisable implantation of gwas2vcf is available from: <https://github.com/MRCIEU/gwas2vcf>. We have also developed a web application that serves as a wrapper for this implementation, and automatically converts flat files to GWAS VCF files while annotating against dbSNP build XXX. Tools for reading, querying and manipulating GWAS VCF files have been developed for R and python also (Table 2).

### Query performance simulation

Densely imputed summary statistics (13,791,467 variants) for a large GWAS of body mass index data were obtained from MR Base [18]. The data were mapped to VCF using Gwas2VCF v1.1.1 and processed using bcftools v1.10 [16] to remove multiallelic variants or records with missing dbSNP identifiers. A tabular (unindexed) file was prepared from the VCF to replicate a typical storage medium currently used by the community. Runtime performance of a range of queries using tabix or UNIX commands was evaluated: single variant selection using dbSNP identifier or chromosome position, multi-variant selection by association P value or 1 Mb genomic interval. Testing was undertaken with five repetitions using VCF or unindexed text formats with and without GZIP compression on an Ubuntu v18.04 server with Intel Xeon(R) 2.0 Ghz processor.

Dockerised implementation code are available from: <https://github.com/MRCIEU/gwas-vcf-performance>

## Results

### How the GWAS VCF format meets the specification

*Human readable and easy to parse*

The compressed VCF format can be easily read with any text viewer. Open-source parsing libraries are available in C (HTSLIB [19]) and Java (HTSJDK [19]) which can be implemented in most modern programming languages. For example the VariantAnnotation package is available in the R/Bioconductor project, and the pysam package in python are mature options for handling VCF files. Bcftools [16] provides user-friendly functionality accessible from the command line.

*Unambiguous interpretation of the data*

Data field descriptions and value types are required and defined in the file header. File validity is enforced during each read/write.

*Unambiguous representation of bi-allelic, multi-allelic and insertion-deletion variants*

Each locus (row) has capacity to store multiple alternative alleles as required. GWAS effect sizes are stored one per alternative allele allowing for bi/multi-allelic and insertion-deletion variants. HTSLIB [19] and HTSJDK [19] parsing libraries have routines for handling complex variants. Using this approach alternative allele(s) are always the effect allele allowing consistency between studies for ease of comparison.

*Genomic information can be validated*

The file header contains genome build, contig identifiers and sequence length. Reference alleles must match the specified reference FASTA. GATK [20] ValidateVariants can be used to verify file validity in which the reference information is compared against the corresponding human genome reference sequence.

*Flexibility on which GWAS fields are recorded and enforcement of essential fields*

All fields are defined in the file header and can be set optional or required as desired. Our specification implements essential fields and reserved keys.

*Capacity to store meta-data about the study or studies*

Each GWAS trait has a row in the file header to store trait description and units, number of variants, study type (case/control or continuous) and unique identifier.

*Allows multiple studies to be stored together*

The sample column was chosen to store GWAS association metrics to allow for multiple traits in a single file enabling distribution of related phenotypes or individually as desired.

*Rapid querying by dbSNP identifier, genomic position range or GWAS summary data values*

The file is sorted karyotypically and indexed to allow rapid queries by genomic position. Refer to query performance section for comparison with standard UNIX tools.

*File compression*

VCF files may be compressed with block GZIP [16] or converted to a binary call file which is a binary VCF companion format [16].

*Readable by existing open-source tools*

A large number of tools support VCF files including: GATK [20], Picard [21], bcftools [16], bedtools [22], vcftools [15] and plink [6]. Bcftools [16] can also provide a tabular extract for use with non-compatible tools.

*Amenable to cloud-based streaming and database storage*

Genomic intervals may be extracted over a network using range-requests which allows for file segments to be read without transferring the whole file. This enables rapid streaming of queries over the internet.

For high-throughput and distributed storage and querying, VCF files can be easily imported into GenomicsDB [23].

## Query performance of GWAS VCF format

We evaluated runtime performance of querying GWAS summary statistics in VCF and unindexed text through simulation studies with five repetitions using a range of common operations.

The fastest mean runtime to extract a single variant by dbSNP identifier (Figure 2) was 0.09 seconds (95% CI 0.09, 0.10) for compressed VCF indexed with rsidx [17]. The second-best performer was 8.6 times slower at 0.78 seconds (95% 0.68, 0.89) using UNIX grep on uncompressed plain text.

Compressed VCF was also quickest to retrieve a single record by chromosome position (Figure 3; mean runtime 0.08 seconds [95% CI 0.07, 0.08]) compared with the second-best performer using UNIX grep on uncompressed plain text which was 21 times slower (mean runtime 1.71 seconds [95% CI 1.58, 1.84]).

The lowest mean runtime to extract a 1 Mb interval (Figure 4) was for compressed VCF (mean query time 0.10 seconds [95% CI 0.09, 0.10]). Using unindexed text, the best performer was for UNIX AWK using an uncompressed file, which was 2730 times slower (mean query time 4.55 seconds [95% CI 4.46, 4.63]).

Finally, we evaluated methods to select variants by association P value (Figure 5). The fastest mean query took 6.19 seconds (95% CI 5.62, 6.76) using AWK on uncompressed text. Meanwhile, bcftools took an average of 39.29 seconds (95% CI 34.25, 44.33) using compressed VCF.

## Discussion

We introduced a specification for storing GWAS summary statistics using the VCF format. This approach has many advantages over existing solutions: provides established methods for handling complex variation (multiallelic and insertion-deletion variants), format validation to ensure data integrity, compatibility with existing open-source tools, human readable, compressed and rapid to query.

Simulations of query performance suggest VCF is inordinately quicker than unindexed and uncompressed flat files for querying by position. Although extracting variants by association P value threshold was quicker using UNIX tools, this could be improved by flagged variants below prespecified thresholds if the exact value is unimportant. For example, all variants at genome-wide significance (P=5e-8) or suggestive association (P=5e-5). Alternatively, VCF files could be loaded into a dedicated database such as GenomicsDB [23] which might offer better query performance.

To encourage adoption, we developed open-source tools: for mapping tabular data to VCF (Gwas2VCF) including web interface, libraries for reading GWAS-VCF files using R or Python3 and processing data for secondary analysis using R (gwasglue). Additionally, we made available over 10,000 complete GWAS summary statistics in VCF format (links below).

## Specification

Available from: <https://github.com/MRCIEU/gwas_vcf_spec>

## Data availability

Full summary statistics for over 10,000 GWAS in VCF format available from the IEU GWAS Database (<https://gwas.mrcieu.ac.uk>)

## Acknowledgments

This study was funded by the NIHR Biomedical Research Centre at University Hospitals Bristol National Health Service Foundation Trust and the University of Bristol. The views expressed are those of the author(s) and not necessarily those of the NIHR or the Department of Health and Social Care.

M.L., B.E., T.R.G. work in the Medical Research Council Integrative Epidemiology Unit at the University of Bristol, which is supported by the Medical Research Council and the University of Bristol (MC\_UU\_00011/4). G.H. is supported by the Wellcome Trust and Royal Society [208806/Z/17/Z].

## References

[1] P. M. Visscher *et al.*, “10 Years of GWAS Discovery: Biology, Function, and Translation,” *American Journal of Human Genetics*, vol. 101, no. 1. Cell Press, pp. 5–22, 06-Jul-2017.

[2] L. Hou and H. Zhao, “A review of post-GWAS prioritization approaches,” *Front. Genet.*, vol. 4, no. DEC, p. 280, Dec. 2013.

[3] G. D. Smith and S. Ebrahim, “‘Mendelian randomization’: Can genetic epidemiology contribute to understanding environmental determinants of disease?,” *International Journal of Epidemiology*. 2003.

[4] B. Bulik-Sullivan *et al.*, “LD score regression distinguishes confounding from polygenicity in genome-wide association studies,” *Nat. Genet.*, 2015.

[5] J. Yang, J. Zeng, M. E. Goddard, N. R. Wray, and P. M. Visscher, “Concepts, estimation and interpretation of SNP-based heritability,” *Nature Genetics*, vol. 49, no. 9. Nature Publishing Group, pp. 1304–1310, 01-Sep-2017.

[6] S. Purcell *et al.*, “PLINK: A tool set for whole-genome association and population-based linkage analyses,” *Am. J. Hum. Genet.*, 2007.

[7] J. Yang, S. H. Lee, M. E. Goddard, and P. M. Visscher, “GCTA: A tool for genome-wide complex trait analysis,” *Am. J. Hum. Genet.*, vol. 88, no. 1, pp. 76–82, Jan. 2011.

[8] P. R. Loh *et al.*, “Efficient Bayesian mixed-model analysis increases association power in large cohorts,” *Nat. Genet.*, vol. 47, no. 3, pp. 284–290, Feb. 2015.

[9] X. Zhou and M. Stephens, “Genome-wide efficient mixed-model analysis for association studies,” *Nat. Genet.*, vol. 44, no. 7, pp. 821–824, Jul. 2012.

[10] C. J. Willer, Y. Li, and G. R. Abecasis, “METAL: fast and efficient meta-analysis of genomewide association scans,” *Bioinforma. Appl. NOTE*, vol. 26, no. 17, pp. 2190–2191, 2010.

[11] F. P. Hartwig, N. M. Davies, G. Hemani, and G. D. Smith, “Two-sample Mendelian randomization: avoiding the downsides of a powerful, widely applicable but potentially fallible technique,” *Int. J. Epidemiol.*, pp. 1717–1726, 2016.

[12] A. Buniello *et al.*, “The NHGRI-EBI GWAS Catalog of published genome-wide association studies, targeted arrays and summary statistics 2019,” *Nucleic Acids Res.*, vol. 47, no. D1, pp. D1005–D1012, Jan. 2019.

[13] Z. Zhu *et al.*, “Integration of summary data from GWAS and eQTL studies predicts complex trait gene targets,” *Nat. Genet.*, vol. 48, no. 5, pp. 481–487, May 2016.

[14] “The Variant Call Format (VCF) Version 4.2 Specification,” 2019.

[15] P. Danecek *et al.*, “The variant call format and VCFtools,” *Bioinformatics*, vol. 27, no. 15, pp. 2156–2158, Aug. 2011.

[16] H. Li, “A statistical framework for SNP calling, mutation discovery, association mapping and population genetical parameter estimation from sequencing data.,” *Bioinformatics*, vol. 27, no. 21, pp. 2987–93, Nov. 2011.

[17] “bioforensics/rsidx: Library for indexing VCF files for random access searches by rsID.” [Online]. Available: https://github.com/bioforensics/rsidx. [Accessed: 05-Mar-2020].

[18] “UK Biobank — Neale lab.” [Online]. Available: http://www.nealelab.is/uk-biobank/. [Accessed: 25-Feb-2020].

[19] H. Li *et al.*, “The Sequence Alignment/Map format and SAMtools,” *Bioinforma. Appl. NOTE*, vol. 25, no. 16, pp. 2078–2079, 2009.

[20] A. McKenna *et al.*, “The genome analysis toolkit: A MapReduce framework for analyzing next-generation DNA sequencing data,” *Genome Res.*, 2010.

[21] “broadinstitute/picard: A set of command line tools (in Java) for manipulating high-throughput sequencing (HTS) data and formats such as SAM/BAM/CRAM and VCF.” [Online]. Available: https://github.com/broadinstitute/picard. [Accessed: 25-Feb-2020].

[22] A. R. Quinlan and I. M. Hall, “BEDTools: a flexible suite of utilities for comparing genomic features,” *Bioinforma. Appl. NOTE*, vol. 26, no. 6, pp. 841–842, 2010.

[23] “GenomicsDB/GenomicsDB: Highly performant data storage in C++ for importing, querying and transforming variant data with Java/Spark. Used in gatk4.” [Online]. Available: https://github.com/GenomicsDB/GenomicsDB. [Accessed: 25-Feb-2020].

[24] K. Voss, J. Gentry, and G. Van Der Auwera, “GATK4 + WDL + Cromwell,” *F1000Research*, vol. 6, p. 4, Aug. 2017.

Table 1. List of requirements for summary statistics storage format

|  |  |
| --- | --- |
| # | Requirement |
| 1 | Human readable and easy to parse |
| 2 | Unambiguous interpretation of the data |
| 3 | Unambiguous representation of bi-allelic, multi-allelic and insertion-deletion variants |
| 4 | Genomic information can be validated |
| 5 | Flexibility on which GWAS fields are recorded and enforcement of essential fields |
| 6 | Capacity to store meta-data about the study or studies |
| 7 | Allows multiple studies to be stored together |
| 8 | Rapid querying by dbSNP identifier, genomic position range or GWAS summary data values |
| 9 | File compression |
| 10 | Readable by existing open-source tools |
| 11 | Amenable to cloud-based streaming and database storage |

GWAS, genome-wide association study. dbSNP, database of single-nucleotide polymorphisms.

Table 2. Open-source tools for working with the summary statistics VCF format

|  |  |  |  |
| --- | --- | --- | --- |
| Program | Purpose | Implementation | Source code link |
| gwas2vcf | Mapping tab separated GWAS summary statistics and EBI format to VCF | Python3 (Docker) | <https://github.com/mrcieu/gwas2vcf> |
| gwas2vcfweb (<http://64.227.44.193:8400/>) | Front-end and queue schedular for gwas2vcf | Python3, Cromwell[24]  (Docker) | <https://github.com/mrcieu/gwas2vcfweb> |
| R/gwasvcf | Library for querying and reading GWAS VCF files | R | <https://github.com/mrcieu/gwasvcf> |
| pygwasvcf | Library for querying and reading GWAS VCF files | Python3 | <https://github.com/mrcieu/pygwasvcf> |
| R/gwasglue | Library for processing GWAS summary statistics ready for secondary analysis | R | <https://github.com/mrcieu/gwasglue> |
| LD Score Regression[4] | Estimating genetic correlation and heritability | Python | <http://github.com/explodecomputer/ldsc> |

GWAS, genome-wide association study. LD, linkage disequilibrium. VCF, variant call format. EBI, European Bioinformatics Institute.

Figure 1. VCF format adapted to store GWAS summary statistics

A screenshot of a social media post

Description automatically generated

Variant call file format for storing GWAS summary statistics. The file contains metadata, variant-level content and variant-trait association statistics. Each field is defined in the metadata including variable type and number. The format can accommodate multiple traits/studies or one per file as required.

Figure 2. Performance comparison for querying summary statistics in plain text and VCF format using dbSNP identifier

A close up of a piece of paper

Description automatically generated

Mean query time (seconds; repetitions n=5) for extracting a single variant using the dbSNP identifier from summary statistics stored in tab-separated text or VCF. AWK, grep, bcftools [16] and rsidx [17] were evaluated using uncompressed and GZIP/BGZIP [16] compressed files. Error bars represent the 95% confidence interval.

Figure 3. Performance comparison for querying summary statistics in plain text and VCF format using chromosome position

A screenshot of a cell phone

Description automatically generated

Mean query time (seconds; repetitions n=5) for extracting a single variant using chromosome position from summary statistics stored in tab-separated text or VCF. AWK, grep, and bcftools [16] were evaluated using uncompressed and GZIP/BGZIP [16] compressed files. Error bars represent the 95% confidence interval.

Figure 4. Performance comparison for querying summary statistics in plain text and VCF format using genomic interval

A close up of a piece of paper

Description automatically generated

Mean query time (seconds; repetitions n=5) for extracting variants within a genomic interval using chromosome position from summary statistics stored in tab-separated text or VCF. AWK and bcftools [16] were evaluated using uncompressed and GZIP/BGZIP [16] compressed files. Error bars represent the 95% confidence interval.

Figure 5. Performance comparison for querying summary statistics in plain text and VCF format using trait association P value threshold

A screenshot of a cell phone

Description automatically generated

Mean query time (seconds; repetitions n=5) for extracting variants by trait association P value from summary statistics stored in tab-separated text or VCF. AWK and bcftools [16] were evaluated using uncompressed and GZIP/BGZIP [16] compressed files. Error bars represent the 95% confidence interval.