Conversion of text formatted population variants (pVCF) to binary formats (PLINK and BGEN) for 200k GraphTyper call

Executive summary

- 200k joint variant calls were converted from pVCF to BGEN and PLINK (BED/BIM/FAM formats)
- 60648 pVCF files processed, producing ~0.5PB of intermediate files (multiallelic split and normalized VCFs + unmerged BED) and ~30TB final data (chromosome merged PLINK and BGEN files)
- The pipeline complexity can be significantly reduced by migrating to PLINK 2.0 (pgen/pvar/psam) format, which supports multiallelic variants and produces much smaller files
- GraphTyper calls on 200k cohort introduced some large, very complex multiallelic variants (>2GB of data in single pVCF line), which lead to issues with bcftools and PLINK
- The file size significantly increased in comparison to the 150k joint-call run (median of 9.36GB for 200k vs. 5.73GB for 150k), requiring machines with more operating memory to successfully finish the conversion task

Scope

This document contains a protocol for converting the pVCF files to binary formats. The protocol is optimized to run on the RAP platform provided by DNA Nexus. The protocol used for 200k is a re-implementation of the PLINK 1.9 format (bed/bim/fam) protocol used for the 150k cohort in the DNA Nexus platform workflow. The workflow code is available in WDL format.

Steps of the protocol

- Variant normalization and splitting multiallelic variants
- Conversion to PLINK (bed) format
- QC for converted files
- Merging PLINK (bed) files for each chromosome
- Conversion of chromosome PLINK (bed) to BGEN format

Software prerequisites

- **dx utility [API v1.0.0, client v0.317.0]** – required for accessing the project file system and submitting jobs on the RAP platform
- plink [**PLINK v1.90b6.27 64-bit (10 Dec 2022)] –** required for pVCF to BED format conversion
- **plink2 [PLINK v2.00a3.6LM AVX2 Intel (14 Aug 2022)] –** required for merging BED files and conversion to BGEN format
- bcftools [**v1.16]** – required for normalization, splitting multiallelic variants
- bcftools [**v1.16m]** – a custom build, a modified version of bcftools that accepts VCF lines >2GB and <~4GB – if this version fails the pipeline fallbacks on default version that omits problematic lines

Input files

The input variant calls are stored in pVCF files. The project consists of 60,648 pVCF files and a total of 533.62TB of compressed text data. The file sizes range from 1.55 MB to 31 GB with a median size of 9.36 GB. The distribution of file sizes is shown in **Figure 1**.

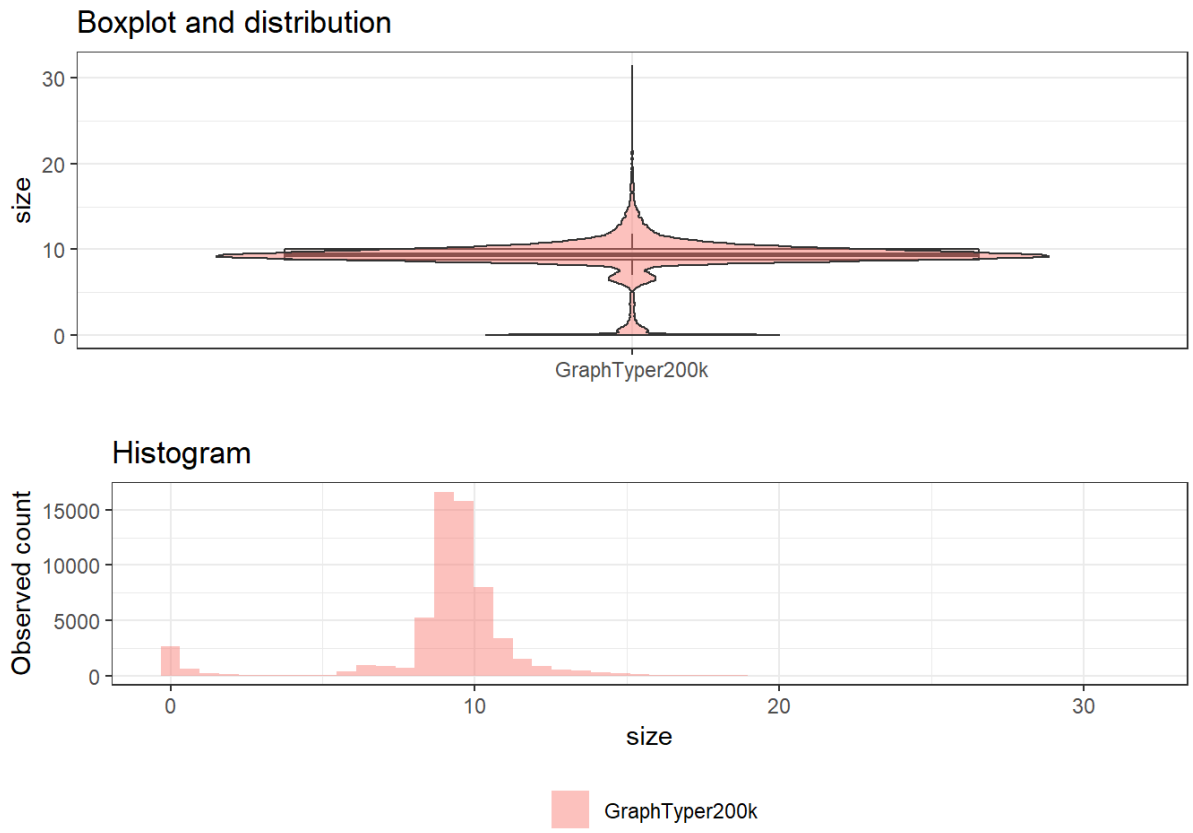

Figure 1: Distribution (A) and histogram (B) of pVCFs sizes [GB].

The smallest file size (1.55 MB) indicates that the pVCF file contains only a header. In a previous implementation, we will make use of this information to quickly identify empty pVCF files and omit them from being processed. However, because of a different number of individuals, the size of "empty" pVCF changes between releases. A new effective algorithm was implemented to identify "empty" pVCFs based on the absence of data lines. This approach would work universally, without the need to a priori determine the header size.
Skipping header-only pVCF is important because processing empty files result in a PLINK executable error and incur the cost of setting up the instance and starting the execution.

Output files

- PLINK BED files (26.84 TB)
- BGEN files (0.87 TB)

Note: BGEN files are Zlib compressed, as produced by PLINK 2. The compression method can be changed to zsdt with qctools. This process will be lengthy – the test jobs on full chromosomes did not finish within 24h. I recommend living zlib as a compression standard for WGS data. The full listings of files are available in CSV format and as appendixes at the end of this document. There are

also some concerns in the community about zsdt compatibility with commonly used tools and frameworks, e.g. Hail (see discussion here: https://discuss.hail.is/t/index-bgen-zlib-compression-exception/2652).

The conversion protocol

This protocol is implemented as DNA Nexus workflow, and is available as a WDL file in the repository, along with additional documentation and Docker build files. However, these steps can be executed on any Unix-compatible machine (Linux, MacOS, WSL) and will produce the same result. Corresponding WDL task names are shown in square brackets.

1) Check if pVCF file have data (discard header-only, "empty" files) [vcf_has_data]

```
cat "~{vcf}" 2>/dev/null | zgrep "^[^#]" 2>/dev/null | head -1 | wc -l
```

This efficient bash script returns 1 if file has data and 0 if file is empty.

2) Split multialleic variants and normalise pVCF

```
bcftools norm "~{vcf}" \
-f /data/GRCh38_full_analysis_set_plus_decoy_hla.fa \
-m -any -Oz --no-version --threads 4 \
-o "~{prefix}".norm.vcf.gz || \
bcftools_vanilla norm "~{vcf}" \
-f /data/GRCh38_full_analysis_set_plus_decoy_hla.fa \
-m -any -Oz --no-version --threads 4 \
-o "~{prefix}".norm.vcf.gz 2> "~{prefix}".norm.stderr.txt || \
true
```

This script uses a modified "bcftools" binary to attempt to process the lines >2GB. If the line largely exceeds 2GB (~4GB or bigger) this script fill fails and the pipe will fall back on standard bvcftools implementation ("bcftools_vanilla"). This script will omit problematic lines (reporting them to stdout, captured by workflow logging) and output the process file. Final "|| true" will guarantee the workflow will continue even if bcftools return output to stderr. The workflow will only fail on no VCF file is produced, or conversion abnormalities are detected by the PLINK validation step.

3) Convert normalized pVCF to PLINK

```
plink --make-bed \
--vcf "~{nornvcf}" \
--keep-allele-order \
--vcf-idspace-to _ \
--double-id \
--allow-extra-chr 0 \
--vcf-half-call m \
--out "~{prefix}"
```

4) Validate PLINK file

```
plink2 --validate --bed "~{bed}" --bim "~{bim}" --fam "~{fam}" >
"~{prefix}".BedValid.txt
```

5) Merge PLINK files

```
plink2 --pmerge-list "~{prefix}"_merge_list.txt --make-bed --out
"~{prefix}" --multiallelics-already-joined --max-alleles 2
```

This script requires the creation of “"~{prefix}"_merge_list.txt” table. This file is generated automatically with DNA Nexus workflow, using following code:

```
cat ~{write_tsv(transpose([select_all(bed), select_all(bim),
select_all(fam)]))} > "~{prefix}"_merge_list.txt
```

If running outside of RAP/WDL build system you need to manually list these files, following the manual for “plink2 --pmerge-list”: https://www.cog-genomics.org/plink/2.0/data#pmerge

6) Convert PLINK to BGEN

```
plink2 --bed "~{bed}" --bim "~{bim}" --fam "~{fam}" --export bgen-1.2
bits=8 ref-first --out "~{prefix}".zlib
```

RAP workflow (WDL) implementation

The WDL implementation provides an efficient, reproducible, complete conversion protocol implementation. It takes a list of pVCF files (RAP file-IDs) for a single chromosome as an input (up to ~5000 pVCF files) and returns merged PLINK and BGEN files alongside validation outputs and a list of failed (~4GB or larger) pVCF lines.

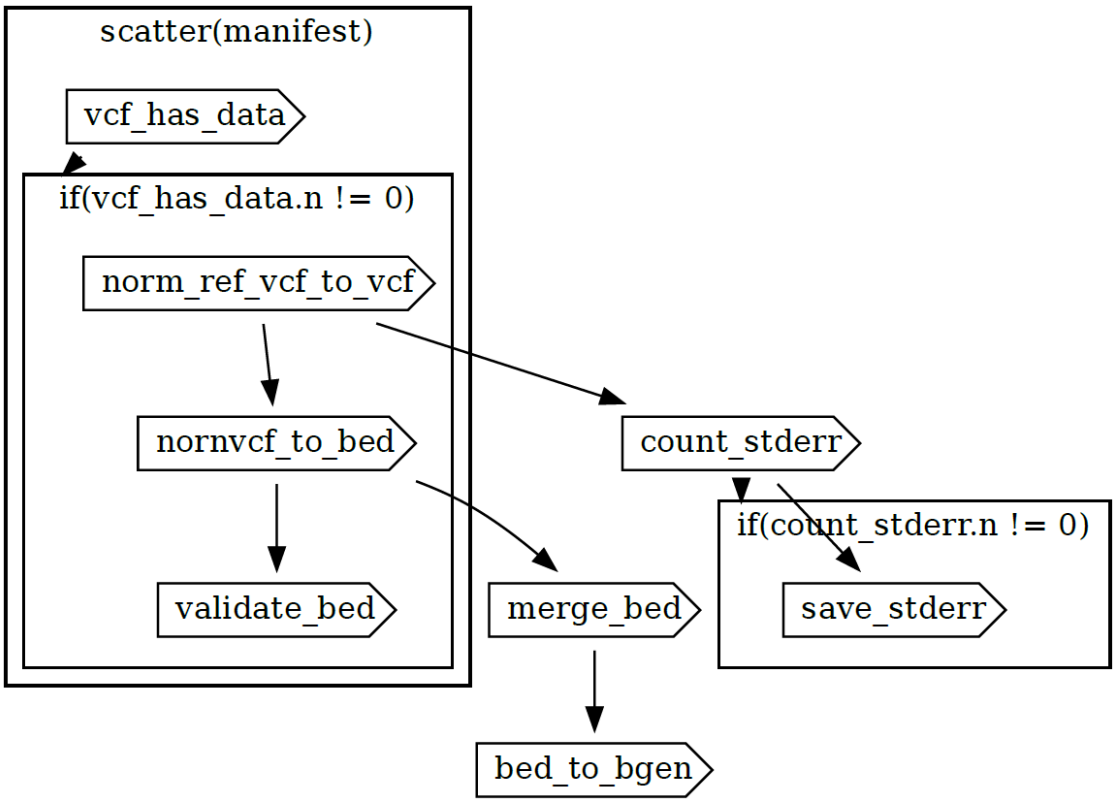

Figure 2: Block diagram representing RAP workflow (auto-generated from WDL)