Advance Access Publication Date: Day Month Year
Applications Note

Genetics and Population Analysis

Sparse Project VCF: efficient encoding of population genotype matrices

Michael F. Lin [1],*, Xiaodong Bai [2], William J. Salerno [2] and Jeffrey G. Reid [2]

[1] mlin.net LLC, San Jose, CA 95113, USA and
[2] Regeneron Genetics Center, Tarrytown, NY 10591, USA.

*To whom correspondence should be addressed.

Associate Editor: XXXXXXX

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Abstract

Summary: Variant Call Format (VCF), the prevailing representation for germline genotypes in population sequencing, suffers rapid size growth as larger cohorts are sequenced and more rare variants are discovered. We present Sparse Project VCF (spVCF), an evolution of VCF with judicious entropy reduction and run-length encoding, delivering ∼10X size reduction for modern studies with practically minimal information loss. spVCF interoperates with VCF efficiently, including tabix-based random access.
Availability and Implementation: Freely available at github.com/mlin/spVCF
Contact: dna@mlin.net

1 Introduction

Variant Call Format (VCF) is the prevailing representation for germline variants discovered by high-throughput sequencing (Danecek *et al.*, 2011). In addition to capturing variants sequenced in one study participant, VCF can represent the genotypes for many participants at all discovered variant loci. This "Project VCF" (pVCF) form is a 2-D matrix with loci down the rows and participants across the columns, filled in with each called genotype and associated quality-control (QC) measures, such as read depths, read strand ratios, and genotype likelihoods.

As the number of study participants N grows (columns), more variant loci are also discovered (rows), leading to super-linear growth of the pVCF genotype matrix. And, because cohort sequencing discovers mostly rare variants, this matrix consists largely of reference-identical genotypes and their high-entropy QC measures. In recent experiments with human whole-exome sequencing (WES), doubling N from 25 000 to 50 000 also increased the pVCF locus count by 43%, and 96% of all loci had non-reference allele frequency below 0.1% (Lin *et al.*, 2018). Empirically, `vcf.gz` file sizes in WES and whole-genome sequencing (WGS) are growing roughly with $N^{1.5}$ in the largest studies as of this writing ($N \approx 100\,000$). Unchecked, we project $N = 1\,000\,000$ WGS will yield petabytes of *compressed* pVCF.

2 Approach

We sought an incremental path to ameliorate the QC entropy and size growth problems in existing pVCF-based pipelines, which may be reluctant to adopt fundamentally different formats or data models addressing these challenges (Layer *et al.*, 2015; Li, 2015; Stilp *et al.*, 2017; LeFaive, 2017; Danek and Deorowicz, 2018; Klarqvist, 2018). To this end, we developed an evolution of VCF, Sparse Project VCF (spVCF), which begins with the same data model and text format, and adds three simple ideas (**Fig. 1**):

(1) *Squeezing: judiciously reducing QC entropy*. In any cell with a reference-identical (or non-called) genotype and QC measures indicating zero reads supporting a variant (typically Allele Depth `AD` $= d, 0$ for any d, but this depends on the upstream tools), we discard all fields except the genotype and the read depth `DP`, which we also round down to a power of two (0, 1, 2, 4, 8, 16, ...; configurable).

This QC squeezing convention, inspired by similar techniques for read quality scores (Fritz *et al.*, 2011; Illumina, 2014; Jun *et al.*, 2015; Bonfield *et al.*, 2018), preserves full detail in all cells indicating any appreciable evidence for a variant, even if a variant genotype is not actually called; in other cells, it maintains a discretized lower bound on the reference depth. We contend that this convention keeps nearly all *useful* information, removing uninformative fluctuations in QC measures.

(2) *Succinct, lossless encoding for runs of reference-identical cells* along both matrix dimensions. First we replace the contents of a reference-identical (or non-called) cell with a quotation mark `"` if it's identical to the cell above it, compressing runs down the column for each sample. Then we run-length encode these quotation marks across the rows, so for example a stretch of 42 marks across a row is written `<tab>"42` instead of repeating `<tab>"` forty-two times.

Even without QC squeezing, identical runs down pVCF columns are a common byproduct of "gVCF merging" tools such as GATK (DePristo *et al.*, 2011; Poplin *et al.*, 2018) GenotypeGVCFs and GLnexus (Lin *et al.*, 2018), when they analyze many closely-spaced loci in large cohorts. The QC squeezing not only reduces the data size prior to run-encoding, but also synergistically lengthens the available runs.

(A) pVCF

```
#CHROM  POS REF ALT FORMAT       Alyssa                          Ben                     Cy
22     1000 A   G   GT:DP:AD:PL  0/0:35:35,0:0,117,402           0/0:29:29,0:0,109,387    0/0:22:22,0:0,63,188
22     1012 CT  C   GT:DP:AD:PL  0/0:35:35,0:0,117,402           0/0:31:31,0:0,117,396    0/1:28:17,11:74,0,188
22     1018 G   A   GT:DP:AD:PL  0/0:35:35,0:0,117,402           0/0:31:31,0:0,117,396    1/1:27:0,27:312,87,0
22     1074 T   C,G GT:DP:AD:PL  0/0:33:33,0,0:0,48,62,52,71,94  ./.:0:0,0:.,.,.,.,.,.    1/2:42:4,20,18:93,83,76,87,0,77
```

(B) spVCF

```
#CHROM  POS REF ALT FORMAT       Alyssa                          Ben                     Cy
22     1000 A   G   GT:DP:AD:PL  0/0:32                          0/0:16                   0/0:16
22     1012 CT  C   GT:DP:AD:PL            "2                                             0/1:28:17,11:74,0,188
22     1018 G   A   GT:DP:AD:PL            "2                                             1/1:27:0,27:312,87,0
22     1074 T   C,G GT:DP:AD:PL            "                     ./.:0                    1/2:42:4,20,18:93,83,76,87,0,77
```

Fig. 1. spVCF encoding example. (A) Illustrative pVCF of four variant loci in three sequenced study participants, with matrix entries encoding called genotypes and several numeric QC measures. Some required VCF fields are omitted for brevity. (B) spVCF encoding of the same example. QC values for reference-identical and non-called cells are reduced to a power-of-two lower bound on read depth `DP`. Runs of identical entries down columns are abbreviated using quotation marks, then runs of these marks across rows are length-encoded. Cy's entries are shown column-aligned for clarity; the encoded text matrix is ragged.

(3) *Checkpointing to facilitate random access* by genome range (row) within a spVCF file. While all *variant* genotype cells are readily accessible from a given spVCF row, fully decoding the reference-identical and non-called cells would require information from an unpredictable number of prior rows. To expedite random access, the spVCF encoder periodically skips run-encoding a row, instead emitting a row identical to the squeezed pVCF. Subsequent run-encoded rows can be decoded by looking back no farther than this *checkpoint* row. Every run-encoded row has an additional informational field with the position of the previous checkpoint. Genome range access proceeds by locating the first desired row, following its pointer back to a checkpoint, and reversing the run-encoding from the checkpoint through the desired row(s).

3 Reference implementation

Our Unix command-line tool `spvcf` provides efficient transcoding between pVCF and spVCF, typically arranged in a shell pipeline to `gunzip` the input and `bgzip` the output. Different invocations of the tool can cause it to (i) squeeze and run-encode pVCF to spVCF, (ii) run-encode pVCF losslessly without squeezing, (iii) squeeze pVCF without run-encoding (producing valid pVCF that is typically much smaller, albeit not as small as spVCF), or (iv) decode spVCF back to pVCF.

If a spVCF file is compressed using `bgzip`, then `tabix` can create a random-access index for it (Li, 2011), as the encoding does not affect the necessary locus-level VCF fields. A subcommand of `spvcf` used instead of `tabix` can then access rows by genome position, consulting the checkpoints to formulate a spVCF "slice" that can be decoded standalone. The encoder checkpoints at a regular, configurable period and at the start of each chromosome; more-strategic checkpointing might improve compression slightly in the future.

The Apache-licensed code, compiled Linux executable, and detailed format documentation are available from: `github.com/mlin/spVCF`

4 Applied tests

We tested spVCF on two sizeable WES studies using different upstream variant-calling pipelines.

First, using $N = 50\,000$ WES from the DiscovEHR study (Dewey *et al.*, 2016), we reduced a GATK-based pVCF file with 620 782 chromosome 2 variant loci from 79GiB `vcf.gz` to a 5.2GiB `spvcf.gz` file, 15X size reduction. Most of this (6.9X) was achieved by the QC squeezing, while the run-encoding contributed an additional 2.2X. Experiments with nested subsets of these $N = 50\,000$ WES indicate `spvcf.gz` file sizes growing roughly with $N^{1.1}$, compared to $N^{1.5}$ for the original `pvcf.gz`. (VCF's binary equivalent, BCF, reduces this example by 1.2X losslessly and exhibits the same $N^{1.5}$ scaling.)

Second, with $N = 49\,960$ WES from UK Biobank (Bycroft *et al.*, 2018; Van Hout *et al.*, 2019), the 75GiB `vcf.gz` for chromosome 1 reduced to 9.2GiB `spvcf.gz`, 8.2X reduction (4.1X from QC squeezing and 2.0X from run-encoding). This dataset was produced using an upstream pipeline that already omits genotype likelihoods in most reference-identical cells, leaving less to be removed by QC squeezing compared to DiscovEHR; spVCF delivered marked size reduction nonetheless.

In these tests, `spvcf` squeezed and run-encoded the uncompressed pVCF at more than twice the speed of `bgzip` compressing the same input (each on a single x86-64 thread; both tools also have multithread modes). The decoder, with inputs and outputs both much smaller than the original pVCF, is several times faster still. Thus, it would be practical – and possibly advantageous – to store spVCF and decode it to pVCF only transiently, whenever downstream analyses require it. The smaller squeezed pVCF also tends to speed up tools consuming it.

5 Discussion

spVCF's interoperability with VCF – resulting from its identical data model and performant transcoder – makes it a practical "next step" for storage and transfer in ongoing cohort sequencing projects. At $N \approx 50\,000$, most size reduction results from QC squeezing rather than sparse run-encoding. We expect run-encoding's relative contribution to increase with N as variant loci become more closely spaced, extending runs with similar read depth. spVCF's better-controlled growth – though still slightly super-linear in N, owing to residual depth fluctuations – clears the way to scale up the VCF data model to $N = 1\,000\,000$ WGS studies in the near future.

Decoding spVCF to pVCF for downstream analysis implies runtime scaling with the less-favorable pVCF growth trend. In principle, many downstream analyses can be computed from the run-encoded spVCF directly, albeit with specialized coding. Upstream, we plan to improve GLnexus scalability by generating spVCF directly without materializing pVCF. Meanwhile many investigators – motivated by advances in linked- and long-read sequencing – are developing haplotype-centric paradigms which may eventually replace VCF.

Acknowledgements

We thank contributors to the Global Alliance for Genomics and Health, Large-Scale Genomics Work Stream for motivating discussions and early feedback on this work; particularly Albert Smith, Yossi Farjoun, Louis Bergelson, Chris Vittal, Cotton Seed, Cristina Gonzalez, Petr Danecek, Marcus Klarqvist, Rishi Nag, Richard Durbin, Thomas Keane, and Ewan Birney.

References

Bonfield, J. K. *et al.* (2018). Crumble: reference free lossy compression of sequence quality values. *Bioinformatics*, **35**(2), 337–339. URL: https://dx.doi.org/10.1093/bioinformatics/bty608, doi:10.1093/bioinformatics/bty608.

Bycroft, C. *et al.* (2018). The uk biobank resource with deep phenotyping and genomic data. *Nature*, **562**(7726), 203.

Danecek, P. *et al.* (2011). The variant call format and VCFtools. *Bioinformatics*, **27**(15), 2156–2158. URL: https://dx.doi.org/10.1093/bioinformatics/btr330, doi:10.1093/bioinformatics/btr330.

Danek, A. and Deorowicz, S. (2018). GTC: how to maintain huge genotype collections in a compressed form. *Bioinformatics*, **34**(11), 1834–1840. URL: https://dx.doi.org/10.1093/bioinformatics/bty023, doi:10.1093/bioinformatics/bty023.

DePristo, M. A. *et al.* (2011). A framework for variation discovery and genotyping using next-generation dna sequencing data. *Nature genetics*, **43**(5), 491.

Dewey, F. E. *et al.* (2016). Distribution and clinical impact of functional variants in 50,726 whole-exome sequences from the discovehr study. *Science*, **354**(6319). URL: http://science.sciencemag.org/content/354/6319/aaf6814, doi:10.1126/science.aaf6814.

Fritz, M. H.-Y. *et al.* (2011). Efficient storage of high throughput dna sequencing data using reference-based compression. *Genome Research*, **21**(5), 734–740. URL: http://genome.cshlp.org/content/21/5/734.abstract, doi:10.1101/gr.114819.110.

Illumina (2014). Reducing whole-genome data storage footprint. Accessed: 2019-02-26. URL: https://www.illumina.com/documents/products/whitepapers/whitepaper_datacompression.pdf.

Jun, G. *et al.* (2015). An efficient and scalable analysis framework for variant extraction and refinement from population scale dna sequence data. *Genome Research*. URL: http://genome.cshlp.org/content/early/2015/04/14/gr.176552.114.abstract, doi:10.1101/gr.176552.114.

Klarqvist, M. D. R. (2018). Tachyon: High-level api for storing and querying sequence variant data. Accessed: 2019-03-24. URL: https://github.com/mklarqvist/tachyon.

Layer, R. M. *et al.* (2015). Efficient genotype compression and analysis of large genetic-variation data sets. *Nature methods*, **13**(1), 63.

LeFaive, J. (2017). Sparse allele vectors specification. Accessed: 2019-02-26. URL: https://github.com/statgen/savvy/blob/d11d790/sav_spec.md.

Li, H. (2011). Tabix: fast retrieval of sequence features from generic TAB-delimited files. *Bioinformatics*, **27**(5), 718–719. URL: https://dx.doi.org/10.1093/bioinformatics/btq671, doi:10.1093/bioinformatics/btq671.

Li, H. (2015). Bgt: efficient and flexible genotype query across many samples. *Bioinformatics*, **32**(4), 590–592.

Lin, M. F. *et al.* (2018). Glnexus: joint variant calling for large cohort sequencing. *bioRxiv*. URL: https://www.biorxiv.org/content/early/2018/06/11/343970, doi:10.1101/343970.

Poplin, R. *et al.* (2018). Scaling accurate genetic variant discovery to tens of thousands of samples. *bioRxiv*. URL: https://www.biorxiv.org/content/early/2018/07/24/201178, doi:10.1101/201178.

Stilp, A. *et al.* (2017). SeqArray—a storage-efficient high-performance data format for WGS variant calls. *Bioinformatics*, **33**(15), 2251–2257. URL: https://dx.doi.org/10.1093/bioinformatics/btx145, doi:10.1093/bioinformatics/btx145.

Van Hout, C. V. *et al.* (2019). Whole exome sequencing and characterization of coding variation in 49,960 individuals in the uk biobank. *bioRxiv*. URL: https://www.biorxiv.org/content/early/2019/03/09/572347, doi:10.1101/572347.