Sequence analysis

BCFtools/csq: Haplotype-aware variant consequences

Petr Danecek * **and Shane A. McCarthy**

Wellcome Trust Sanger Institute, Wellcome Genome Campus, Hinxton, CB10 1SA, UK

*To whom correspondence should be addressed.

Associate Editor: XXXXXXX

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Abstract

Motivation: Prediction of functional variant consequences is an important part of sequencing pipelines, allowing the categorization and prioritization of genetic variants for follow up analysis. However, current predictors analyze variants as isolated events, which can lead to incorrect predictions when adjacent variants alter the same codon, or when a frame-shifting indel is followed by a frame-restoring indel. Exploiting known haplotype information when making consequence predictions can resolve these issues.
Results: BCFtools/csq is a fast program for haplotype-aware consequence calling which can take into account known phase. Consequence predictions are changed for 2839 of 4934 compound variants found in the 81.7M variants in the 1000 Genomes Project data, with an average of 139 compound variants per haplotype. Predictions match existing tools when run in localized mode, but the program is an order of magnitude faster and requires an order of magnitude less memory.
Availability: The program is freely available for commercial and non-commercial use in the BCFtools package which is available for download from `http://samtools.github.io/bcftools`
Contact: pd3@sanger.ac.uk

1 Introduction

With the rapidly growing number of sequenced exome and whole-genome samples, it is important to be able to quickly sift through the vast amount of data for variants of most interest. A key step in this process is to take sequencing variants and provide functional effect annotations. For clinical, evolutionary and genotype-phenotype studies, accurate prediction of functional consequences can be critical to downstream interpretation. There are several popular existing programs for predicting the effect of variants such as the Ensembl Variant Effect Predictor (VEP) (McLaren *et al*., 2016), SnpEff (Cingolani *et al*., 2012) or ANNOVAR (Wang *et al*., 2010). One significant limitation is that they are single-record based and, as shown in Figure 1, this can lead to incorrect annotation when surrounding in-phase variants are taken into account.

With recent experimental and computational advancements, phased haplotypes over tens of kilobases are becoming routinely available through the reduced cost of long-range sequencing technologies (Zheng *et al*., 2016) and the increased accuracy of statistical phasing algorithms (Sharp *et al*., 2016; Loh *et al*., 2016) due to the increased sample cohort sizes (McCarthy *et al*., 2016). We present a new variant consequence predictor implemented in BCFtools/csq that can exploit this information.

2 Methods

For haplotype-aware calling, a phased VCF, a GFF3 file with gene predictions and a reference FASTA file are required. The program begins by parsing gene predictions in the GFF3 file, then streams through the VCF file using a fast region lookup at each site to find overlaps with regions of supported genomic types (exons, CDS, UTRs, or general transcripts). Active transcripts that overlap variants being annotated are maintained on a heap data structure. For each transcript we build a haplotype tree which includes phased genotypes present across all samples. The nodes in this tree correspond to VCF records with as many child nodes as there are alleles. In the worst case scenario of each sample having two unique haplotypes, the number of leaves in the haplotype tree does not grow exponentially but stops at the total number of unique haplotypes present in the samples. Thus each internal node of the tree corresponds to a set of haplotypes with the same prefix and the leaf nodes correspond to a set of haplotypes shared by multiple samples. Once all variants from a transcript are retrieved from the VCF, the consequences are determined on a spliced transcript sequence and reported in the VCF.

Representing the consequences is itself a challenge as there can be many samples in the VCF, each with different haplotypes, thus making the prediction non-local. Moreover, diploid samples have two haplotypes and at each position there can be multiple overlapping transcripts. To represent this rich information and keep the output compact, all unique consequences are recorded in a per-site INFO tag with structure similar to existing annotators. Consequences for each haplotype are recorded in a per-sample FORMAT tag as a bitmask of indexes into the list of consequences recorded in the INFO tag. The bitmask interleaves each haplotype so that when stored in BCF (binary VCF) format, only 8 bits per sample are required for most sites. The bitmask can be translated into a human readable

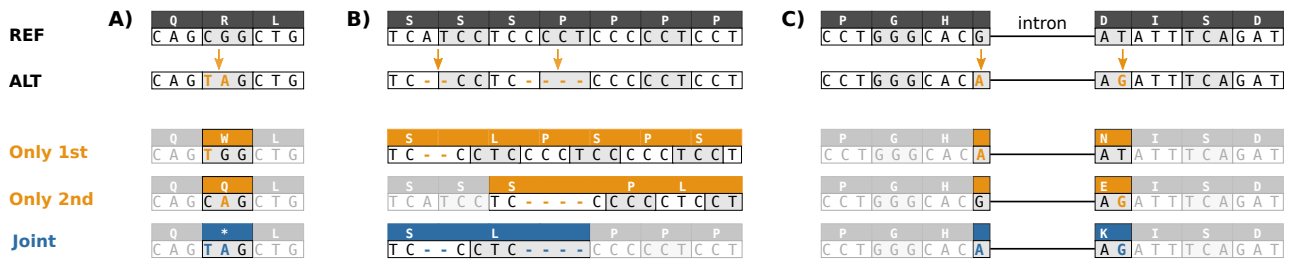

Fig. 1. Three types of compound variants that lead to incorrect consequence prediction when handled in a localized manner, i.e. each variant separately rather than jointly. A) Multiple SNVs in the same codon result in a TAG stop codon rather than an amino acid change. B) A deletion locally predicted as frame-shifting is followed by a frame-restoring variant. Two amino acids are deleted and one changed, the functional consequence on protein function is likely much less severe. C) Two SNVs separated by an intron occur within the same codon in the spliced transcript. Unchanged areas are shaded for readability. All three examples were encountered in real data.

form using the BCFtools/query command. Consequences of compound variants linking multiple sites are reported at one of the sites only with others referencing this record by position.

3 Results

3.1 Accuracy

Accuracy was tested by running in localized mode and comparing against one of the existing local consquence callers (VEP) using gold-standard segregation-phased NA12878 data (Cleary *et al*., 2014). With each site treated independently, we expect good agreement with VEP at all sites. Indeed, only 11 out of 1.6M predictions differed within coding regions. See the Supplement (S2) for further details about these differences.

3.2 Performance

Performance was compared to VEP (McLaren *et al*., 2016), SnpEff (Cingolani *et al*., 2012) and ANNOVAR (Wang *et al*., 2010) running on the same NA12878 data. BCFtools/csq, in localized and haplotype-aware mode, was faster by an order of magnitude than the fastest of the programs and required an order of magnitude less memory, see Suppl. Table S1. For haplotype-aware calling, memory and time both scale linearly with number of sites in the transcript buffer and number of samples, see Suppl. Fig. S2-S5.

3.3 Compound variants in 1000 Genomes

Applied to the 1000 Genomes Phase 3 data, haplotype-aware consequence calling modifies the predictions for 2839 of 4934 compound variants, summarised in Table 1. On average, we observe 139 compound variants per haplotype (Suppl. Fig. S1), recover 28 variants incorrectly predicted as deleterious, and identify 2 newly deleterious compound variants.

Table 1. Summary of BCFtools/csq consequence type changes from localized (rows) to haplotype-aware (columns) calling in 1000 Genomes data. Blue / orange background indicates a change to a less / more severe prediction. Only variants with modified predictions are included in the table.

	haplotype aware:	synonymous	missense	stop lost	stop gained	inframe
localized:	synonymous	67	2453	2	16	28
	missense	43	6796	1	142	52
	stop lost	0	4	17	0	0
	stop gained	0	271	0	30	3
	frameshift	0	3	0	1	249

4 Discussion

Correctly classifying the functional consequence of variants in the context of nearby variants in known phase can change the interpretation of their effect. Variants previously flagged as benign or less severe may now be flagged as deleterious and vice versa. In a rare disease sequencing study, for example, this may have a significant impact as these functional annotations may determine which variants to follow up for further study.

Previous work by Wei *et al*. (2015) focussed on single nucleotide variants and, to estimate haplotypes, required access to the BAM alignment files. Our approach ignores haplotype calling, leaving that as a problem to be solved by other means. Instead, we focus on providing fast consequence prediction taking into account all variation within a transcript.

The standard programs have rich functionality beyond the reporting of variant consequence, and the aim of BCFtools/csq is not to compete with that. Instead, we propose haplotype-aware calling is included in annotation pipelines for enhanced downstream analysis.

Acknowledgements

The authors thank Monica Abrudan, Daniel Gaffney, Thomas Keane, William McLaren and Kim Wong for helpful discussions and ideas.

Funding

The work was supported by the Wellcome Trust (WT098051) and a grant co-funded by the Wellcome Trust and Medical Research Council (WT098503).

References

Cingolani P, *et al*. (2012) A program for annotating and predicting the effects of single nucleotide polymorphisms, SnpEff, *Fly (Austin)*, **6(2)**, 80-92.

Cleary JG, *et al* (2014) Joint variant and de novo mutation identification on pedigrees from high-throughput sequencing data, *J. Comput. Biol.*, **21(6)**, 405-419.

Loh P, *et al*. (2016) Reference-based phasing using the Haplotype Reference Consortium panel, *Nat. Genetics.*, **48(11)**, 1443-1448.

McCarthy S, *et al*. (2016) A reference panel of 64,976 haplotypes for genotype imputation, *Nat. Genetics.*, **48(10)**, 1279-83.

McLaren W, *et al*. (2016) The Ensembl Variant Effect Predictor, *Genome Biol.*, **17(1)**, 122.

Sharp K, *et al*. (2016) Phasing for medical sequencing using rare variants and large haplotype reference panels, *Bioinformatics.*, **32(13)**, 1974-80.

Wang K, Li M, Hakonarson H (2010) ANNOVAR: functional annotation of genetic variants from high-throughput sequencing data, *Nucleic Acids Res.*, **38(16)**, e164.

Wei L *et al*. (2015) MAC: identifying and correcting annotation for multi-nucleotide variations, *BMC Genomics.*, **16**, 569.

Zheng GX, *et al*. (2016) Haplotyping germline and cancer genomes with high-throughput linked-read sequencing, *Nat. Biotechnol.*, **34(3)**, 303-11.

Supplementary Material

S1 Comparison and evaluation data

We downloaded two datasets for comparison and evaluation.

- **NA12878**: The Cleary *et al*. (2014) segregation-phased NA12878 data was downloaded from the below URL. Only the 4,452,828 sites labelled as `FILTER=PASS` were used.
  `https://s3-us-west-2.amazonaws.com/10x.files/samples/genome/NA12878_WGS/NA12878_WGS_phased_variants.vcf.gz`
- **1000 Genomes**: The 1000 Genomes Phase 3 (v5a) haplotype data was downloaded from the below URL. Evaluations were performed on the the autosomes, chr1-22.
  `ftp://ftp.1000genomes.ebi.ac.uk/vol1/ftp/release/20130502/`

S2 Differences between BCFtools/csq and VEP

With BCFtool/csq run in localized mode on the NA12878 data above, only 11 out of 1.6M predictions differed within coding regions. Six cases were due to ambiguity of insertions occurring at a splice site, which can be interpreted either as part of the exon or outside of it. In these cases, VEP called a frameshift whereas BCFtools called splice acceptor or donor. Note that these all occurred in extremely short introns (2-3bp) bringing into question the accuracy of gene predictions for these transcripts. Another 4 cases were incorrect predictions by VEP, for example a missense event in incomplete CDS misclassified as "start lost", and 1 difference was due to an ambiguous alignment of a 14bp deletion in a repeat region.

Details of these differences are given in the supplementary file `VEP-vs-BCFtools.txt`. The script used to determine these differences is the supplementary file `diff-bt-vep`.

S3 Compound variants in NA12878 and 1000 Genomes

In haplotype-aware mode, on the NA12878 data above, BCFtools/csq found 40 frame-recovered indels with altered sequence shorter than 30bp, 13 stop gains recovered as missense events, and 5 novel stops. The list of all compound variants found in NA12878 is given in the supplementary file `cmpnd.NA12878.txt`.

The list of all compound variants found in the 1000 Genomes dataset is given in the supplementary file `cmpnd.1000GP.txt.gz`. These are summarized in Table 1 and Suppl. Fig. S1. Note that the summary number of compound variants (4934) is different from the number of variants in Table 1 because in the table each compound variant has been split into its constituents to allow direct comparison between the compound variant consequences called in haplotype-aware mode and the single variant consequences from the localized mode.

The script used to generate these lists of compound variants is the supplementary file `prn-cmpnd`.

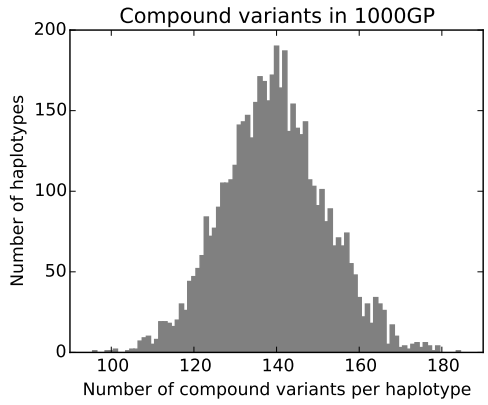

Fig. S1. Number of compound variants in 1000 Genomes Project data.

S4 Performance and scaling

Table S1. Performance comparison of BCFtools/csq with three popular consequence callers using a single-sample VCF with 4.5M sites. Note that at the time of writing, a new version of VEP is being prepared which brings the running time down and makes the performance of VEP comparable to the other programs (William McLaren, personal communication).

	VEP	snpEff	ANNOVAR	csq (local)	csq (haplotype)
CPU time	6 hrs	35 min	24 min	101 sec	92 sec
Memory	17 GB	7.8 GB	3.9 GB	207 MB	208 MB

The commands and versions used in the above evaluation are listed below.

VEP (v82)

```
perl variant_effect_predictor_v82.pl \
  --db_version 82 -t SO --format vcf --cache \
  --dir vep_cache --offline --species human \
  --symbol --biotype --vcf --no_stats \
  --assembly GRCh37 --no_progress --quiet
```

snpEff (v4.2)

```
java -Xmx14g -jar snpEff.jar -noStats -noLof \
  -noInteraction -noNextProt -noMotif GRCh37.75
```

ANNOVAR (2016Feb01)

```
table_annovar.pl --vcfinput --tempdir tmp \
  --outfile NA12878.annovar --buildver hg19 \
  --protocol ensGene --operation g \
  NA12878.vcf annovar/humandb
```

BCFtools/csq (local) (v1.3.1-179-gd7f6692)

```
bcftools csq -f hs37d5.fa -g GRCh37.82.gff3.gz \
  -l -s -
```

BCFtools/csq (haplotype) (v1.3.1-179-gd7f6692)

```
bcftools csq -f hs37d5.fa -g GRCh37.82.gff3.gz \
  -p s -n 64
```

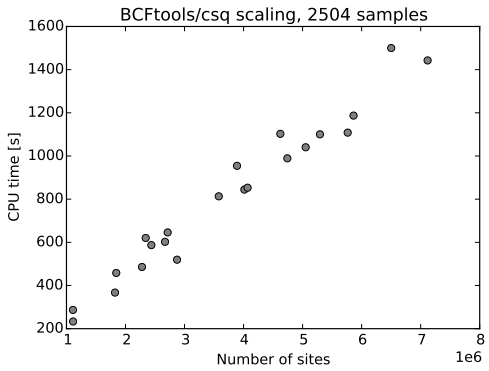

Fig. S2. Linear scaling of BCFtools/csq with the number of sites. Tests were performed on 1000 Genomes Phase 3 data.

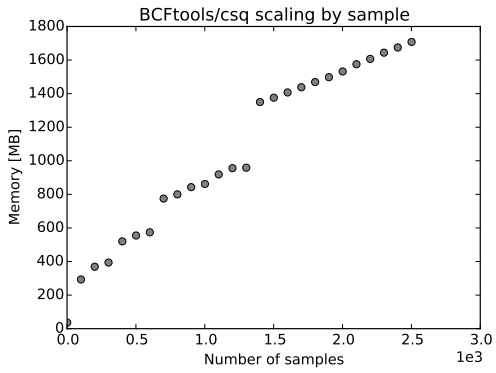

Fig. S4. Linear scaling of memory required by BCFtools/csq with the number of samples. Tests were performed on 1000 Genomes Phase 3 data, chromosome 1.

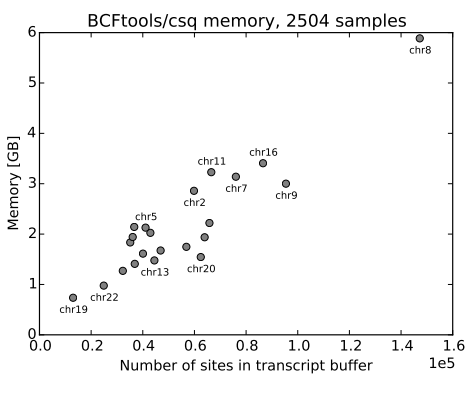

Fig. S3. The maximum amount of memory required by the BCFtools/csq haplotype-aware calling is linear with the number of sites that needs to be kept in memory before flushing the transcript. Tests were performed on 1000 Genomes Phase 3 data.

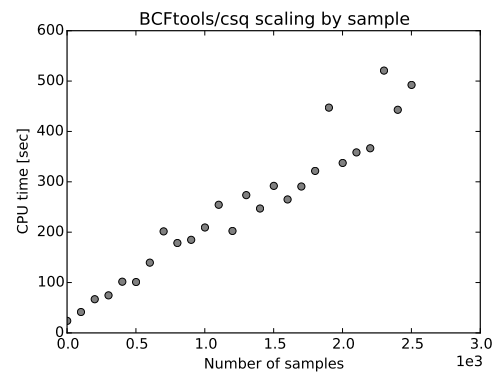

Fig. S5. The CPU time by the number of samples. Tests were performed on 1000 Genomes Phase 3 data, chromosome 1.