vcferr: Development, Validation, and Application of a SNP Genotyping Error Simulation Framework

VP Nagraj
Signature Science, LLC
Charlottesville, VA

Matthew Scholz
Signature Science, LLC
Charlottesville, VA

Shakeel Jessa
Signature Science, LLC
Austin, TX

Jianye Ge
University of North Texas Health Science Center
Fort Worth, Texas

August Woerner
University of North Texas Health Science Center
Fort Worth, Texas

Meng Huang
University of North Texas Health Science Center
Fort Worth, Texas

Bruce Budowle
University of North Texas Health Science Center
Fort Worth, Texas

Stephen D. Turner
Signature Science, LLC
Charlottesville, VA

March 25, 2022

Abstract

Motivation: Genotyping error can impact downstream SNP-based analyses. Simulating various modes and and levels of error can help investigators better understand potential biases caused by miscalled genotypes.

Results: We have developed and validated `vcferr`, a tool to probabilistically simulate genotyping error and missingness in VCF files. We demonstrate how `vcferr` could be used to address a research question by introducing varying levels of error of different type into a sample in a simulated pedigree, and assessed how kinship analysis degrades as a function of kind and type of error.

Software Availability: `vcferr` is available for installation on PyPi (`https://pypi.org/project/vcferr/`). The software is released under the MIT license with source code available on GitHub (`https://github.com/signaturescience/vcferr`).

1 Introduction

Genotyping error is known to occur on genomic sequencing and array platforms (Pompanon et al. 2005). Miscalled genotypes can manifest in a variety of ways, including drop in or drop out of heterozygous calls. SNP-based analysis tools may be sensitive to incorrect calls, and the impact of such error could be differential depending on the downstream application. To facilitate better understanding of how inaccurate genotype calls might bias SNP-based analyses, we aimed to develop a mechanism to simulate genotyping error. Here we describe our efforts to develop, validate, and demonstrate usage of `vcferr`, a novel, user-friendly framework for probabilistically simulating genotyping error and missingness in VCF files (Danecek et al. 2011).

2 Approach

`vcferr` is written in Python and distributed to run directly from a command line interface. The tool is built on top of `pysam` (Danecek et al. 2021), which internally manages reading and writing VCF file contents. When run on an input VCF, `vcferr` will load all variant records for a given sample. The tool checks each genotype, and randomly draws from a list of possible genotypes (heterozygous, homozygous for the alternate allele, homozygous for the reference allele, missing) with each element weighted by error rates specified by the user. The processing runs iteratively for every site in the input VCF, with the output streamed or optionally written to a new output VCF file. The following error and missingness models are available:

- `rarr` = Heterozygous drop out to homozygous reference: `0/1` or `1/0` to `0/0`
- `aara` = Homozygous alternate to heterozygous: `1/1` to `0/1`
- `rrra` = Homozygous reference to heterozygous drop in: `0/0` to `0/1`
- `raaa` = Heterozygous to homozygous alternate: `0/1` or `1/0` to `1/1`
- `aarr` = Homozygous alternate to homozygous reference: `1/1` to `0/0`
- `rraa` = Homozygous reference to homozygous alternate: `0/0` to `1/1`
- `ramm` = Heterozygous to missing: `0/1` or `1/0` to `./.`
- `rrmm` = Homozygous reference to missing: `0/0` to `./.`
- `aamm` = Homozygous alternate to missing: `1/1` to `./.`

We validated `vcferr` by reviewing concordance metrics between VCF files before and after error simulation. To perform this validation we used data from the 1000 Genomes Project (Auton et al. 2015) and companion error assessment tool we developed which wraps `bcftools stats` (Danecek et al. 2021) and calculates the proportion of specific mismatches above between samples in two VCF files (see Supplementary Information, S1). First, we retrieved all genotype calls for chromosome 22 from the 1000 Genomes FTP. Next, we simulated a set of error rates for one sample. We observed that the proportion of mismatches reported by our `bcftools`-based error assessment procedure aligned as expected to each of the specified error probabilities. We also confirmed that the simulation of error in one sample did not trigger any mismatches for other samples in the VCF. The chromosome 22 VCF includes 1,103,547 positions and 2504 samples, and genotype error simulation took ~5 minutes on a single CPU. The code used to fully reproduce this validation analysis is available in the supplement (S2).

3 Usage

`vcferr` is packaged in Python and available to be installed from PyPi via `pip`. Once installed, the tool can be used through the command line without running the Python interpreter. `vcferr` requires a path to an input VCF from which error will be simulated as the first positional argument. The tool also requires a specification for `--sample` for the ID of the sample to be simulated as it appears in the VCF header. All other error flags are optional, as is `--output-vcf` which if used will define the path to which the VCF should be written. If no value is passed to a given error or missingness option, then the probability will default to 0. The following is a minimal example of usage:

```
pip install vcferr
VCF="/path/to/input/vcf.gz"
vcferr $VCF --sample='sampleX' --p_rarr=0.05 --p_raaa=0.01 --output_vcf=sampleX.err.vcf.gz
```

To demonstrate how `vcferr` could be used to address a research question, we conducted an analysis of the impact of genotyping error on pairwise kinship estimation. We first generated simulated pedigree data using `ped-sim` (Caballero et al. 2019) with founders drawn from the 1000 Genomes GBR population. The

simulated data was masked to a subset of 590,241 sites, including content from the Illumina Global Screening Array. With all data originally simulated to include no error, we next identified an individual for which we would inject genotyping error. We used `vcferr` to iterate through each error mode from 0 to 20% error at 1% steps, holding all other error modes constant at 0. Next, we obtained KING robust (Manichaikul et al. 2010) kinship coefficients from `akt` (Arthur et al. 2016) for all pairwise comparisons that included the error-simulated individual in the resulting simulation VCFs. We then used the kinship coefficients to infer relatedness out to the third-degree and compared accuracy of degree inference to truth from the original simulated pedigree. The results of this analysis are summarized in Figure 1. In short, certain kinds of error dramatically decrease accuracy of relatedness inference with the KING robust estimator. Note that the sample for which errors were simulated included 430,056 homozygous reference sites, 105,413 heterozygous sites, and 54,772 homozygous alternate site, resulting in varying absolute counts of errors being simulated at each 0 to 20% step (as expected in any sample with genotypes from microarray or sequencing). By leveraging `vcferr` to inject a range of genotyping error scenarios, we can identify the scenarios that are most likely to impact kinship analysis. The code used in this demonstration analysis is available in the supplement (S3).

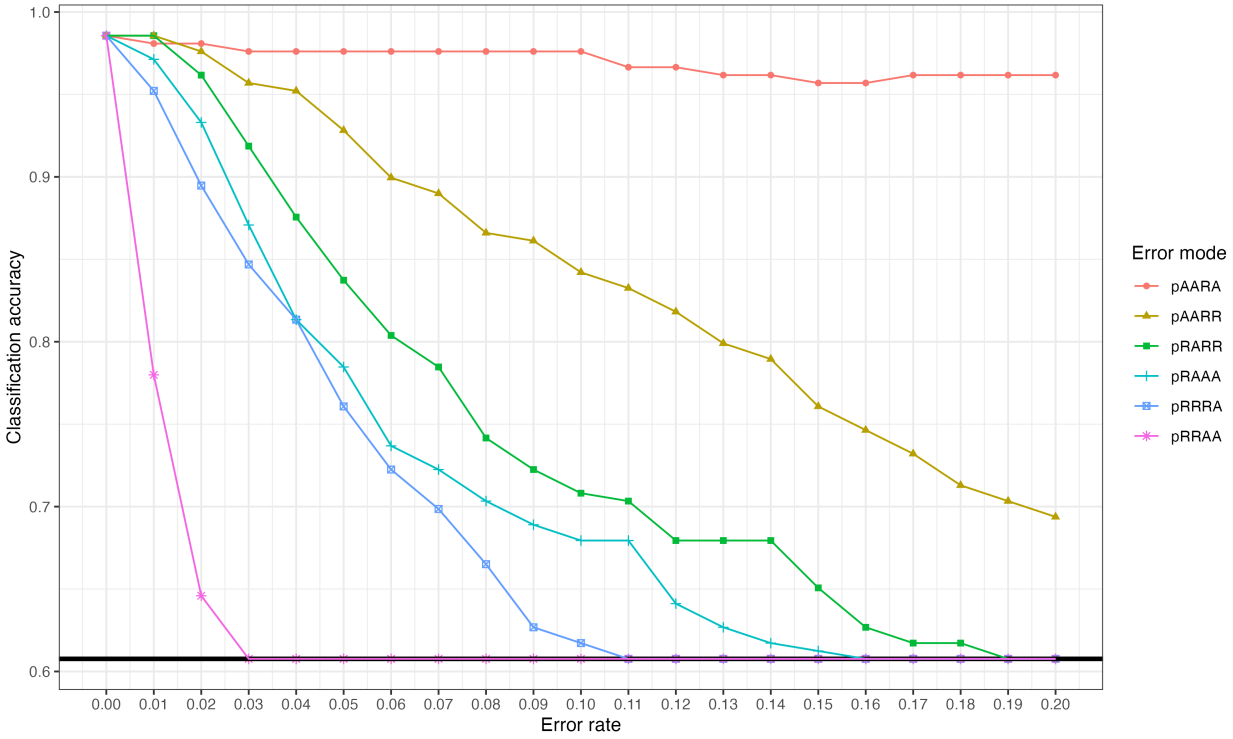

Figure 1: Impact of genotyping error on inferred degree classification accuracy. Each line corresponds to a type of error, with error rates from 0 to 0.2 stepped at 0.01 increments while holding all other types of error constant at 0. The solid black line at bottom represents the accuracy at guessing, which is the floor for relationship degree classification.

4 Discussion

`vcferr` provides a flexible, user-friendly interface to simulate genotyping error in VCF files. While tools exist to simulate entire VCFs with error (e.g. `ped-sim` (Caballero et al. 2019)), to our knowledge at the time of writing there are currently no other tools that can simulate arbitrary individuals in the VCF at varying types and levels of error while leaving all other samples as-is. We anticipate that `vcferr` will be useful for researchers in multiple areas, including forensic genomics and medical genetics.

References

Arthur, Rudy, Ole Schulz-Trieglaff, Anthony J Cox, and Jared O'Connell. 2016. "AKT: ancestry and kinship toolkit." *Bioinformatics* 33 (1): 142–44. https://doi.org/10.1093/bioinformatics/btw576.

Auton, Adam, Gonçalo R. Abecasis, David M. Altshuler, Richard M. Durbin, Gonçalo R. Abecasis, David R. Bentley, Aravinda Chakravarti, et al. 2015. "A Global Reference for Human Genetic Variation." *Nature* 526 (7571): 68–74. https://doi.org/10.1038/nature15393.

Caballero, Madison, Daniel N. Seidman, Ying Qiao, Jens Sannerud, Thomas D. Dyer, Donna M. Lehman, Joanne E. Curran, et al. 2019. "Crossover Interference and Sex-Specific Genetic Maps Shape Identical by Descent Sharing in Close Relatives." *PLOS Genetics* 15 (12): e1007979. https://doi.org/10.1371/journal.pgen.1007979.

Danecek, Petr, Adam Auton, Goncalo Abecasis, Cornelis A. Albers, Eric Banks, Mark A. DePristo, Robert E. Handsaker, et al. 2011. "The variant call format and VCFtools." *Bioinformatics* 27 (15): 2156–58. https://doi.org/10.1093/bioinformatics/btr330.

Danecek, Petr, James K. Bonfield, Jennifer Liddle, John Marshall, Valeriu Ohan, Martin O. Pollard, Andrew Whitwham, et al. 2021. "Twelve Years of SAMtools and BCFtools." *GigaScience* 10 (2): giab008. https://doi.org/10.1093/gigascience/giab008.

Manichaikul, Ani, Josyf C. Mychaleckyj, Stephen S. Rich, Kathy Daly, Michèle Sale, and Wei-Min Chen. 2010. "Robust relationship inference in genome-wide association studies." *Bioinformatics* 26 (22): 2867–73. https://doi.org/10.1093/bioinformatics/btq559.

Pompanon, François, Aurélie Bonin, Eva Bellemain, and Pierre Taberlet. 2005. "Genotyping Errors: Causes, Consequences and Solutions." *Nature Reviews. Genetics* 6 (11): 847–59. https://doi.org/10.1038/nrg1707.