VCF Toolz Documentation

Release 1.2.0

Steve Davis

Jun 02, 2023

Contents

Contents:

CHAPTER 1

VCF Toolz

Tools for working with Variant Call Format files.

VCF Toolz was developed by the United States Food and Drug Administration, Center for Food Safety and Applied Nutrition.

- Free software
- Documentation: https://vcftoolz.readthedocs.io
- Source Code: https://github.com/CFSAN-Biostatistics/vcftoolz
- PyPI Distribution: https://pypi.python.org/pypi/vcftoolz

1.1 Features

- Compares the snps in two or more VCF files.
- Lists the snps that are unique to each VCF file with full genotype information per snp.
- Lists the snps that are missing from each VCF file if present in at least two other VCF files.
- Generates Venn diagrams of positions and snps in the VCF files.
- Reports precision, recall, and F1 score when the truth is known.
- Reports the effectiveness of filtered variants when the truth is known.
- Reformat the VCF file in a tall-narrow format for easy viewing and diffs.
- Count samples, positions, calls, snps, indels, other variants, missing calls, and filter reasons.
- Plot calls along the length of the genome and show the location of filtered calls.

1.2 Citing VCF Toolz

To cite VCF Toolz, please reference the VCF Toolz paper:

https://doi.org/10.21105/joss.01144

1.3 License

See the LICENSE file included in the VCF Toolz distribution.

CHAPTER 2

Installation

At the command line:

```
$ pip install --user vcftoolz
```

Update your .bashrc file with the path to user-installed python packages:

```
export PATH=~/.local/bin:$PATH
```

Or, if you have virtualenvwrapper installed:

```
$ mkvirtualenv vcftoolz
$ pip install vcftoolz
```

2.1 Upgrading VCF Toolz

If you previously installed with pip, you can upgrade to the newest version from the command line:

```
$ pip install --user --upgrade vcftoolz
```

2.2 Uninstalling VCF Toolz

If you installed with pip, you can uninstall from the command line:

```
$ pip uninstall vcftoolz
```

CHAPTER 3

Usage

3.1 Narrow

Some VCF files can be very wide and difficult to view with genotypes for many samples. Using VCF Toolz, you can reformat the data into a tall and narrow format for easier viewing and to facilitate automated diff of files. This is not a standard file format. Output is written to stdout.

To reformat the data in narrow format:

```
vcftoolz narrow file.vcf > narrow.tsv
```

The resulting file looks like this:

```
Sample       CHROM       POS      REF  ALT  GT  SDP  RD   AD  RDF  RDR  ADF  ADR  FT
CFSAN000189  CP006053.1  115714   C    T    0   102  102  0   49   53   0    0    PASS
CFSAN000191  CP006053.1  115714   C    T    0   59   59   0   37   22   0    0    PASS
CFSAN000211  CP006053.1  115714   C    T    0   64   64   0   30   34   0    0    PASS
CFSAN000661  CP006053.1  115714   C    T    1   32   0    32  0    0    11   21   PASS
CFSAN000189  CP006053.1  147186   G    T    0   105  105  0   63   42   0    0    PASS
CFSAN000191  CP006053.1  147186   G    T    1   51   0    51  0    0    35   16   PASS
CFSAN000211  CP006053.1  147186   G    T    0   51   51   0   28   23   0    0    PASS
CFSAN000661  CP006053.1  147186   G    T    0   43   43   0   25   18   0    0    PASS
CFSAN000189  CP006053.1  190695   A    G    0   107  107  0   75   32   0    0    PASS
CFSAN000191  CP006053.1  190695   A    G    1   38   0    37  0    0    26   11   PASS
CFSAN000211  CP006053.1  190695   A    G    1   82   0    82  0    0    55   27   PASS
CFSAN000661  CP006053.1  190695   A    G    0   28   28   0   16   12   0    0    PASS
CFSAN000189  CP006053.1  1834926  T    .    .   .    .    .   .    .    .    .    .
CFSAN000191  CP006053.1  1834926  T    .    .   .    .    .   .    .    .    .    .
CFSAN000211  CP006053.1  1834926  T    .    .   13   0    13  0    0    0    13   ␣
↪Region
CFSAN000661  CP006053.1  1834926  T    .    .   .    .    .   .    .    .    .    .
```

By default, all genotype calls are included in the output. Command line options allow you to exclude some calls. To exclude calls matching the reference, missing calls, and filtered calls:

```
vcftoolz narrow --exclude_ref --exclude_missing --exclude_filtered file.vcf > narrow.
↪tsv
```

To see a full list of options for excluding genotype calls:

```
vcftoolz narrow --help
```

3.2 Compare

You can find the similarites and differences between VCF files with the `compare` command. It works with multiple files and multiple samples per file.

To compare VCF files:

```
vcftoolz compare file1.vcf file2.vcf file3.vcf > summary.txt
```

By default, all genotype calls are included in the output. Command line options allow you to exclude some calls. To exclude calls matching the reference, missing calls, and filtered calls:

```
vcftoolz compare --exclude_ref --exclude_missing --exclude_filtered file1.vcf file2.
↪vcf file3.vcf > summary.txt
```

To see a full list of options for excluding genotype calls:

```
vcftoolz compare --help
```

3.2.1 Summary report

A comparison report is written to stdout listing the differences between the VCF files. The report lists the following:

- Variant positions found only in one of the VCF files
- Sample variants found only in one of the VCF files
- Variant positions missing in each VCF file when found in at least 2 other VCF files
- Sample variants missing in each VCF file when found in at least 2 other VCF files

When run with the `--truth` option, the first VCF file is regarded as correct. The following additional metrics are generated:

- Number of true positives, false negatives, false positives
- Precision, recall, F1 score
- Number of true negative variants correctly removed by filtering
- Number of false negative variants incorrectly removed by filtering

3.2.2 Venn diagrams

When 2 or 3 VCF files are compared, the following Venn diagrams are generated:

- venn2.positions.pdf - Venn diagram of variant positions between all pairs of VCF files
- venn2.snps.pdf - Venn diagram of sample variants between all pairs of VCF files

When 3, 4, 5, or 6 VCF files are compared, one of the following Venn diagrams is generated:

- venn3.pdf - Side by side Venn diagrams of variant positions and sample variants
- venn4.pdf - Side by side Venn diagrams of variant positions and sample variants
- venn5.pdf - Side by side Venn diagrams of variant positions and sample variants
- venn6.pdf - Side by side Venn diagrams of variant positions and sample variants

3.2.3 Tabulated results

A detailed spreadsheet can be generated with the `--tabulate` command line option. The spreadsheet contains one row per position and one column per sample. The rows are ordered in such a way that variants found in multiple files are grouped together. In other words, each section of the Venn diagram will have corresponding rows grouped together in the spreadsheet.

The spreadsheet looks like this:

Chrom	Pos	Ref	CFSAN000189	CFSAN000191	CFSAN000211	CFSAN000661
CP006053.1	115714	C	0	0	0	T-111
CP006053.1	147186	G	0	T-111	0	0
CP006053.1	190695	A	0	G-011	G-111	0
CP006053.1	221363	G	0	0	A-111	0
CP006053.1	268434	C	0	T-011	T-111	0
CP006053.1	309014	C	0	T-111	0	0
CP006053.1	309657	C	0	T-111	0	0
CP006053.1	388919	A	0	G-111	G-111	0
CP006053.1	432510	T	0	0	0	G-111

Each cell of the table contains the alternate allele and a string of zeros and ones to indicate which VCF files contain the variant. For example:

- 0 indicates none of the VCF files contained a variant at this position for this sample
- G-111 indicates an alternate allele G found in all 3 VCF files specified on the command line.
- T-011 indicates an alternate allele T found in the 2nd and 3rd VCF files specified on the command line.

3.3 Count

The count command counts samples, positions, calls, snps, indels, other variants, missing calls, and filter reasons, while allowing you to restrict which calls are eligible for counting.

To count metrics in the VCF file:

```
vcftoolz count file.vcf
```

Output is written to stdout:

```
  3 samples
282 positions
846 calls
  0 heterozygous calls
846 homozygous calls
194 reference calls
194 snp calls
  0 indel calls
  0 other variant calls
266 missing calls
388 calls PASS filter
192 calls failing any filter
187 calls failing filter Region
  3 calls failing filter VarFreq60
  1 calls failing filter RawDpth
  1 calls failing filter Depth3
```

By default, all genotype calls are included in the output. To count only homozygous snps:

```
vcftoolz count --exclude_indels --exclude_vars --exclude_refs --exclude_hetero --
↪exclude_filtered --exclude_missing file.vcf
```

To see a full list of options for excluding genotype calls:

```
vcftoolz count --help
```

3.4 Plot

The plot command shows a bar-chart of the locations of calls along the length of the genome. This is intended for VCF files with multiple samples. The height of the bars indicates the number of samples having calls at each position. In addition, a cumulative sum of calls is plotted (right axis). The cumsum line is useful to detect areas of high-density snps. By default, every CHROM is included in the plot. The x-axis depicts the genome with contigs arranged in order from largest to smallest. A command line option lets you plot a single CHROM, which might be useful to zoom-in on a particular region. The plot command also emits a list of contigs to stdout with the start and end position of each. If the VCF file contains failed calls, there will be a separate plot depicting the failed calls for every failure reason found in the VCF file. Note that some calls can be failed for more than one reason and therefore some calls will be drawn more than once.

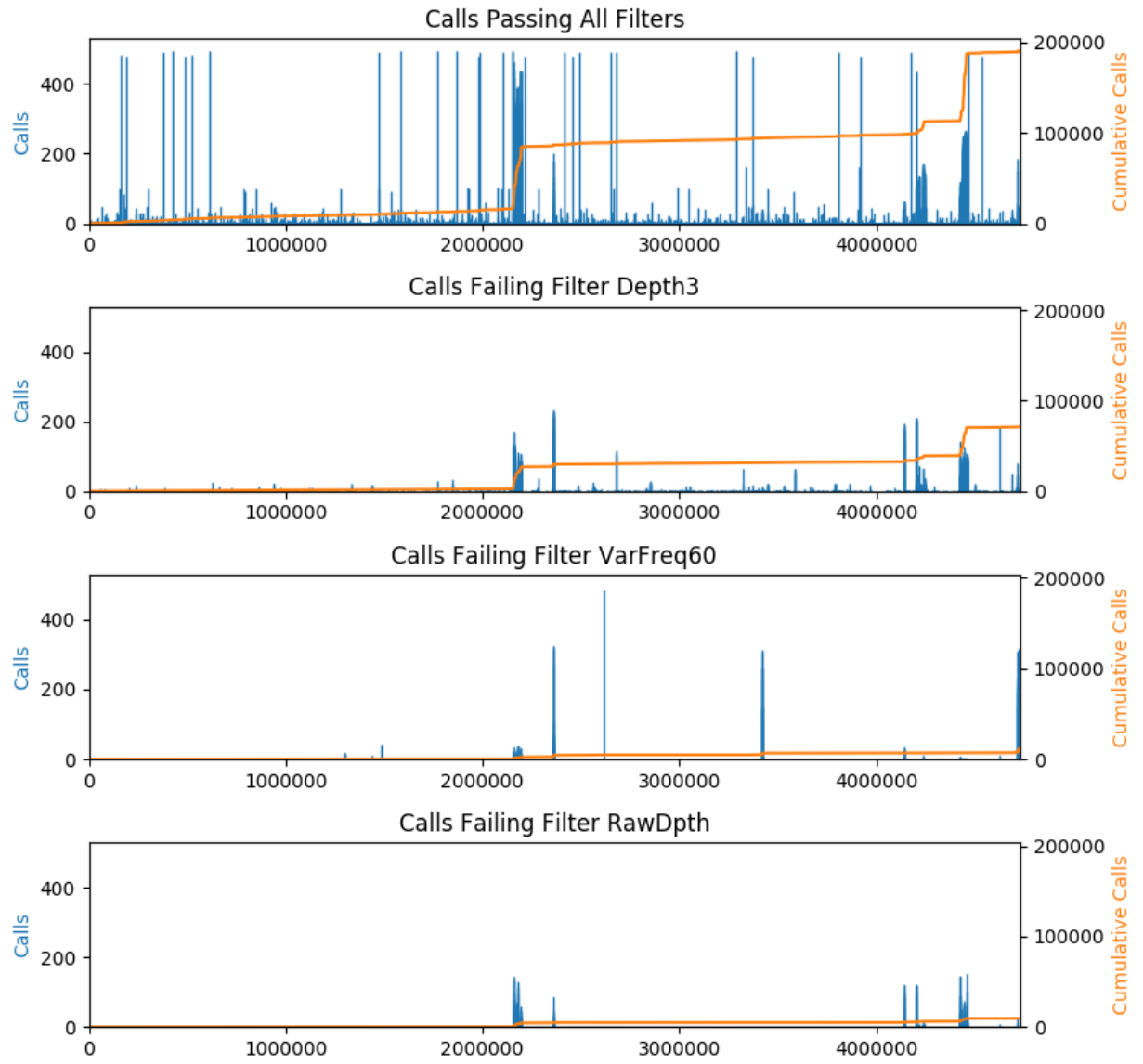
To plot calls:

```
vcftoolz plot --exclude_refs --exclude_missing file.vcf reference.fasta image.png
```

By default, all genotype calls are included in the output. Command line options allow you to exclude some calls. The plot command displays additional useful information when it finds filtered calls. For that reason, excluding filtered calls is not recommended.

To see a full list of options for excluding genotype calls:

```
vcftoolz plot --help
```


Calls Passing All Filters
Calls Failing Filter Depth3
Calls Failing Filter VarFreq60
Calls Failing Filter RawDpth
Calls
Cumulative Calls
0
200
400
0
100000
200000
0
1000000
2000000
3000000
4000000

CHAPTER 4

Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

4.1 Types of Contributions

4.1.1 Report Bugs

Report bugs at https://github.com/CFSAN-Biostatistics/vcftoolz/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

4.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" is open to whoever wants to implement it.

4.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with "feature" is open to whoever wants to implement it.

4.1.4 Write Documentation

VCF Toolz could always use more documentation, whether as part of the official VCF Toolz docs, in docstrings, or even on the web in blog posts, articles, and such.

4.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/CFSAN-Biostatistics/vcftoolz/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

4.2 Get Started!

Ready to contribute? Here's how to set up *vcftoolz* for local development.

1. Fork the *vcftoolz* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/vcftoolz.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv vcftoolz
$ cd vcftoolz/
$ pip install sphinx_rtd_theme    # the documentation uses the ReadTheDocs theme
$ pip install pytest
$ python setup.py develop
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

   Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ flake8 vcftoolz tests
$ pytest -v
$ tox
```

   To get flake8 and tox, just pip install them into your virtualenv.

6. Update the documentation and review the changes locally with sphinx:

```
$ cd docs
$ sphinx-build -b html . ./_build
$ xdg-open _build/index.html
```

7. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

8. Submit a pull request through the GitHub website.

4.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 2.7, 3.5, 3.6, 3.7, and for PyPI.

4.4 Tips

To run a subset of tests:

```
$ pytest -v tests/test_vcftoolz.py
```

CHAPTER 5

Credits

5.1 Development Lead

- Steve Davis

5.2 CFSAN Bioinformatics Team

- Steve Davis

5.3 External Contributors

None yet. Why not be the first?

CHAPTER 6

History

6.1 1.2.3 (2023-06-02)

- Fixed imports in our fork of pyvenn

6.2 1.2.2 (2023-06-01)

- Fixed broken Pyvenn dependency

6.3 1.2.1 (2023-05-09)

- Update PyVCF to PyVCF3

6.4 1.2.0 (2019-04-04)

- Fix defect in narrow command wrongly printing ALT=. when GT=.
- Add the `count` command to count samples, positions, calls, snps, indels, other variants, filtered calls, missing calls, and filter reasons.
- Add the `plot` command to plot calls along the length of the genome and show the location of filtered calls.
- Change the text of the compare report to refer to “Calls”, not “Sample snps”.
- Drop support for Python 3.4, which is not supported by matplotlib.
- Add support for Python 3.7.

6.5 1.1.1 (2019-03-26)

- Replace None with '.' when printing call data.
- Support VCF files with multiple alternate alleles per position.

6.6 1.1.0 (2019-02-06)

- Support reading gzip compressed vcf files.

6.7 1.0.0 (2018-11-20)

- First public release.

CHAPTER 7

Indices and tables

- genindex
- search