*Bioinformatics*, 34(24), 2018, 4241–4247
doi: 10.1093/bioinformatics/bty443
Advance Access Publication Date: 1 June 2018
Original Paper

OXFORD

Genetics and population analysis

# Comparing complex variants in family trios

**Berke Ç. Toptaş[1,2,*], Goran Rakocevic[1,2], Péter Kómár[1,2] and Deniz Kural[1,2]**

[1]R&D Seven Bridges Genomics and [2]R&D Totient, Cambridge, MA 02142, USA

*To whom correspondence should be addressed.
Associate Editor: Oliver Stegle

Received on February 1, 2018; revised on May 3, 2018; editorial decision on May 24, 2018; accepted on May 29, 2018

## Abstract

**Motivation:** Several tools exist to count Mendelian violations in family trios by comparing variants at the same genomic positions. This naive variant comparison, however, fails to assess regions where multiple variants need to be examined together, resulting in reduced accuracy of existing Mendelian violation checking tools.
**Results:** We introduce VBT, a trio concordance analysis tool, which identifies Mendelian violations by approximately solving the 3-way variant matching problem to resolve variant representation differences in family trios. We show that VBT outperforms previous trio comparison methods by accuracy.
**Availability and implementation:** VBT is implemented in C++ and source code is available under GNU GPLv3 license at the following URL: https://github.com/sbg/VBT-TrioAnalysis.git.
**Contact:** cagkantoptas@gmail.com
**Supplementary information:** Supplementary data are available at *Bioinformatics* online.

## 1 Introduction

Recent technological advancements enabled a rapid progress in our understanding and characterization of the human genome, assessment of the scale and extent of genomic variation present in human genome (The 1000 Genome Project Consortium *et al.*, 2015) as well as creation of a vast body of knowledge related to the functioning of human body and rare diseases (Jamuar and Tan, 2015). Pedigree-based genetics plays a crucial role in uncovering the genetic origins of diseases, where family trios are analyzed, and genomic variants that disagree with Mendel's law of segregation are identified.

A particularly important use case of trio analysis is the identification of de novo mutations which have repeatedly been implicated in rare and complex diseases (Deciphering Developmental Disorders Study, 2017; Hidalgo *et al.*, 2016). De novo mutations occur with relatively low frequencies ($1.2 \times 10^{-8}$) (Conrad *et al.*, 2011; Kong *et al.*, 2012) compared with the typical amount of variants a person has. Therefore, accurate strategies are essential for identification of such variants which typically starts with assessing Mendelian Inheritance rules of the calls from family trios followed by using sophisticated statistical models [DeNovoGear (Ramu *et al.*, 2013), PhaseByTransmission (Francioli *et al.*, 2017), TrioDeNovo (Wei et al., 2015)].

Such trio analysis can also be used for truth-free benchmarking of variant calling pipelines (Douglas *et al.*, 2002; Komar and Deniz 2017, 2017; Nutsua *et al.*, 2015; Pilipenko *et al.*, 2014). In many cases, Mendelian violations are considered as sequencing/variant calling errors due to the low mutation rate. Trio concordance analysis is useful where no truth-set exists and allows using variants from all regions of the genome as opposed to current whole genome truth-sets which are limited to a set of high confidence regions in a few samples, excluding many regions of the genome (Zook *et al.*, 2014). Improved truth-free benchmarking will also guide the development of future genome analysis pipelines such as graph genome pipeline (Rakocevic *et al.*, 2017).

Several tools exist (RTG-mendelian, GATK-SelectVariants, Vcftools-mendel) that count Mendelian violations using naive locus-by-locus variant comparison. In this approach, each record in the merged trio vcf is processed independently, and only variants with coinciding reference positions are analyzed together. This method fails to provide an accurate analysis in cases where multiple records are affecting the same locus.

Here we address a problem during the identification of Mendelian violations in the data from a family trio, one which arises

from varying variant representations. Regions with several overlapping variants often have a number of different ways in which they can be represented, all of which conform to the widely accepted VCF standard (Danecek *et al.*, 2011); the same is true for most variants which are complex in nature, and even some simple indels (Fig. 1a). The choice of which of the possible representations is produced often depends on the variant context (other nearby variants) and the set of sequencing reads used to identify the variant. If this choice happens to be different between different members of the pedigree, comparing the three sets of calls position by position will result in detection of Mendelian violations, even though the underlying haplotypes are Mendelian compliant (Fig. 1b).

Problems related to variant representation has been recognized in the context of benchmarking NGS data processing methods, and numerous approaches have been developed for comparing two sets of results for a single sample [SMaSH (Talwalkar *et al.*, 2014), Vcfeval (Cleary *et al.*, 2015), VarMatch (Sun and Medvedev, 2016)]. However, none of these tools are capable of resolving the issue with data from a family trio.

One way to unify variant representations across a family trio is to use a joint variant caller such as GATK GenotypeGVCFs. This tool aggregates variants of multiple samples by combining overlapping trio records and re-genotyping them. Although joint calling resolves many representation issues, it is still unable to merge complex overlapping indels affecting the same site. This method also requires GATK (McKenna *et al.*, 2010) HaplotypeCaller as variant caller which eliminates the benchmarking purpose of trio analysis.

In this paper, we present VBT, a Mendelian violation detection tool that uses an advanced variant comparison to deal with ambiguities arising from different variant representations. VBT extends the variant comparison algorithm of vcfeval (Cleary *et al.*, 2015) for trio concordance analysis. We show that VBT outperforms all previous trio comparison methods regarding the accuracy of detecting Mendelian violations.

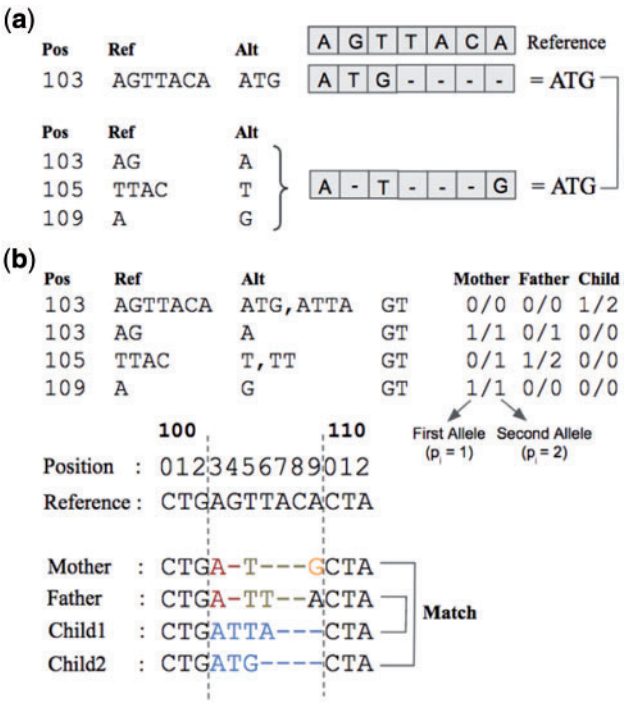


**Fig. 1.** (**a**) Representation difference in indels. The variant in position 103 is represented as a single indel in first vcf and 2 indels + 1 SNP in the second vcf. After they are applied on the reference sequence, it is seen that they are equivalent. (**b**) A toy example of variant representation difference in family trios. Naive trio comparison tools mark all four records as a Mendelian violation. However, a consistent combination can be found if they are processed together

## 2 Materials and methods

### 2.1 Pairwise variant comparison and extension for Mendelian violation identification

The pairwise comparison algorithm of vcfeval eliminates variant representation differences in VCFs by applying variants back to the reference sequence. For each of baseline (gold standard) and called (test) diploid variant sets (we refer variants as diploid single-sample VCF entities which have two alleles), two haplotype sequence (i.e. Haplotype A and Haplotype B) are formed by applying alleles of variants to the reference sequence in such a way that, haplotype A(B) of baseline set becomes identical to haplotype A(B) of called set. Many ways of forming such haplotype sequences are possible using different subsets of baseline and called variants and using different phasings. Vcfeval aims to identify the largest baseline and called variant subset out of possible combinations that satisfy the equality condition.

For a diploid variant set V from a single sample VCF, we define a phasing vector, $P_V = \{p_1, p_2, .. p_{|V|}\} \in \{1, 2\}^{|V|}$, where the $i$th value (1 or 2) indicates whether the first or second allele (Fig. 1b) of the $i$th variant is selected for the maternal haplotype. Similarly $P_{V'}$ denotes the opposite phasing vector $\{3\text{-}p_1, 3\text{-}p_2, .., 3\text{-}p_{|V|}\}$, which indicates the alleles on the paternal haplotype, not selected by $P_V$. A haplotype function h(V, $P_V$) is defined (Cleary *et al.*, 2015) to produce the haplotype sequence (i.e. outputs a single string) obtained by applying all variants of V to the reference sequence using the $P_V$ phasing vector. vcfeval defines the variant matching problem as finding the optimal sets of variants $X^{opt}$, $Y^{opt}$, and their corresponding phasing vectors $P_X{}^{opt}$, $P_Y{}^{opt}$, that solves the following optimization problem:

$$\begin{aligned} X^{opt},\ P_X{}^{opt} = \ & \arg\max\ I[h(X,\ P_X),\ h(Y,\ P_Y)] \\ Y^{opt},\ P_Y{}^{opt} \qquad & I[h(X,\ P_{X'}),\ h(Y,\ P_{Y'})]\ |X| \\ & X \subseteq B,\ Y \subseteq C \\ & P_X \in \{1,2\}^{|X|},\ P_Y \in \{1,2\}^{|Y|} \end{aligned} \tag{1}$$

where B and C denote baseline and called variant sets, and $X^{opt}$ and $Y^{opt}$ are the sets of variants which maximize the number of matching variants in baseline and called variant set. I[seq1, seq2] is the indicator function that performs a simple string comparison operation which outputs 1 if seq1=seq2, and 0 otherwise.

For VBT, we aim to extend this definition for family trios to detect Mendelian violations. Instead of baseline and called variant sets, we use mother, father and child variant sets denoted by M, F and C, respectively. We define the trio matching problem as finding the optimal sets $X^{opt}$, $Y^{opt}$, $Z^{opt}$ and their corresponding phasing vectors $P_X{}^{opt}$, $P_Y{}^{opt}$, $P_Z{}^{opt}$ that solve the following optimization problem:

$$\begin{aligned} X^{opt},\ Y^{opt},\ Z^{opt} = \ & \arg\max\ I[h(X, P_X),\ h(Z, P_Z)] \\ P_X{}^{opt},\ P_Y{}^{opt},\ P_Z{}^{opt} \qquad & I[h(Y, P_Y),\ h(Z, P_{Z'})]\ |Z| \\ & X \subseteq M,\ Y \subseteq F,\ Z \subseteq C \\ & P_X \in \{1,2\}^{|X|},\ P_Y \in \{1,2\}^{|Y|},\ P_Z \in \{1,2\}^{|Z|} \end{aligned} \tag{2}$$

By following the Mendelian Inheritance rules, Eq. (2) seeks for the optimum mother, father and child variant sets where the

maternal haplotype sequence of the child sample is identical to one of two mother haplotype sequences, and paternal haplotype sequence of the child sample is identical to one of two father haplotype sequences. $X^{opt}$, $Y^{opt}$ and $Z^{opt}$ denote the set of Mendelian-consistent variants in the mother, father and child samples, and the remaining variants $M\backslash X^{opt}$, $F\backslash Y^{opt}$ and $C\backslash Z^{opt}$ are marked as Mendelian violations. VBT uses a heuristic that approximates Eq. (2) described in the following section.

## 2.2 Separate haplotype sequence construction and Same Allele Match Elimination

An alternative approach of identifying Mendelian violations in trio data is constructing maternal and paternal haplotype sequences of the child sample separately. To construct the maternal sequence of the child sample, we can search for the optimum mother and child variant sets which share a single allele rather than two. Similarly, we can construct the paternal sequence using the shared-allele father and child variant sets. Then we can take the intersection of the child variants from mother–child and father–child shared allele searching. Since all of the variants in the intersection child variant set share one allele each with parent samples, we can assess these child variants as Mendelian consistent and the remaining child variants as Mendelian violations.

One problem with the separate haplotype sequence construction is, we need to guarantee that the child's haplotypes use opposite phases $P_Z$ and $P_{Z'}$. For heterozygous child variants, if one of the two alleles is not present in either the mother–child or the father–child sequences, they should be reported as a Mendelian violation. For instance, if the alleles of mother is A/A, father is C/A and child is A/G for a multi-allelic SNP variant at the same position, then the child's variant matches with both parents' variants with allele A. On the other hand, Allele G of child is not present in any of the parents. Although pairwise comparisons with both parents indicate one matching allele of the child, this locus is a Mendelian violation because the same phase is used for both matches. In most cases, we can indeed mark it as such. We call this condition *same allele matching*.

One point, one need to be careful about during elimination of same allele matching condition is that, in a family trio, child variants often match to parent variants with both of their alleles. For these child variants, any of the two alleles can be present in the final haplotype sequence. For example, if the alleles of mother is A/T, father is C/T and child is A/T for a variant, both 'A' and 'T' can be selected as a shared allele between mother and child, however, if the 'T' allele is selected, then same allele matching condition occurs whereas selecting 'A' as the shared allele resolves the problem. During haplotype construction, the allele selection of the child variant between parent samples is unknown. Therefore, in cases where both alleles of the parent and child are matching, the allele is chosen arbitrarily. To identify which parent variants are sharing both alleles with child variants and eliminate the wrong phasing selection, we applied 2-stage variant comparison where we start searching for shared genotypes [i.e. Eq. (1)] between parent and child followed by shared allele searching. We write the equations of shared genotype searching stage as:

$$\begin{aligned} X^{opt},\ P_X{}^{opt} = \ & \arg\max\ I[h(X,\ P_X),\ h(Z,\ P_Z)] \\ Z1^{opt},\ P_{Z1}{}^{opt} \qquad & I[h(X,\ P_{X'}),\ h(Z,\ P_{Z'})]\ |Z| \\ & X \subseteq M,\ Z \subseteq C \\ & P_X \in \{1,2\}^{|X|},\ P_Z \in \{1,2\}^{|Z|} \end{aligned} \tag{3}$$

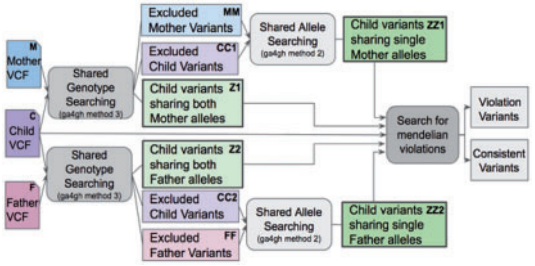


**Fig. 2.** VBT pipeline using vcfeval best path algorithm and GA4GH benchmarking standard methods (Krusche *et al.*, 2018). Included variants are present in the best common path between parent and child while excluded variants are eliminated from that path

$$\begin{aligned} Y^{opt},\ P_Y{}^{opt} = \ & \arg\max\ I[h(Y,\ P_Y),\ h(Z,\ P_Z)] \\ Z2^{opt},\ P_{Z2}{}^{opt} \qquad & I[h(Y,\ P_{Y'}),\ h(Z,\ P_{Z'})]\ |Z| \\ & Y \subseteq F,\ Z \subseteq C \\ & P_Y \in \{1,2\}^{|Y|},\ P_Z \in \{1,2\}^{|Z|} \end{aligned} \tag{4}$$

where $Z1^{opt}$ and $Z2^{opt}$ are child variants sharing both alleles with mother and father variants respectively. From the remaining variants $M\backslash X^{opt}$ (=: MM), $F\backslash Y^{opt}$ (=: FF), $C\backslash Z1^{opt}$ (=: CC1) and $C\backslash Z2^{opt}$ (=: CC2); we obtain all child variants sharing a single allele by maximizing the number of variants in constructing a single haplotype sequence, ignoring the alternative phases of variant sets:

$$\begin{aligned} XX^{opt},\ P_{XX}{}^{opt} = \ & \arg\max\ I[h(XX,\ P_{XX}), h(ZZ1,\ P_{ZZ1})]\ |ZZ1| \\ ZZ1^{opt},\ P_{ZZ1}{}^{opt} \qquad & XX \subseteq MM,\ ZZ1 \subseteq CC1 \\ & P_{XX},\ P_{ZZ1} \end{aligned} \tag{5}$$

$$\begin{aligned} YY^{opt},\ P_{YY}{}^{opt} = \ & \arg\max\ I[h(YY,\ P_{YY}),\ h(ZZ2,\ P_{ZZ2})]\ |ZZ2| \\ ZZ2^{opt},\ P_{ZZ2}{}^{opt} \qquad & YY \subseteq FF,\ ZZ2 \subseteq CC2 \\ & P_{YY},\ P_{ZZ2} \end{aligned} \tag{6}$$

where, during maximization, $P_{XX}$, $P_{YY}$, $P_{ZZ1}$ and $P_{ZZ2}$ are required to be such that the reference allele ('0') is never used in any comparison. I.e. if a variant has the genotype 1|0, the corresponding phasing is not allowed to take the value 2, because that would correspond to the '0' allele.

The reason for not allowing the reference allele to be used in Eqs. (5) and (6) is the ambiguity caused by the equal representation of excluded variant and included reference allele. If we allow reference alleles in the haplotype function for Eqs. (5) and (6), child variants having reference phasing would always be included regardless of the corresponding parent variant. For example, if genotype of the mother is 0/2, father is 2/2 and child is 0/1 for a variant call, mother and child variants would be included in mother–child side because they share '0' allele. In father–child side, there is no shared allele, but once the father variant is excluded, that position becomes reference and child variant alone could be included with '0' allele. In the end, child variant would be present on both mother and father final haplotypes and would be marked as Mendelian consistent, while it is a violation in reality. With the above restriction on the phasing vectors, we eliminate this mistake.

We solve the maximization problem given in Eqs. (3), (4), (5) and (6) using dynamic programming solution of Cleary *et al.*, (2015) where haplotype sequences of samples are procedurally generated, and the total combination count is kept small using clever dynamic pruning methods. We also introduce a small change to the algorithm to improve reference overlapping variant comparison which is described in the Supplementary Text, Section 2.

Using Eqs. (3), (4), (5) and (6), we construct the VBT pipeline as shown in Figure 2 to obtain our four child variant sets $Z1^{opt}$, $Z2^{opt}$,

$ZZ1^{opt}$, $ZZ2^{opt}$ with their phasing information $P_{Z1}{}^{opt}$, $P_{Z2}{}^{opt}$, $P_{ZZ1}{}^{opt}$, $P_{ZZ2}{}^{opt}$. Using these 4 variant set, we identify Mendelian consistent and violation child variants with the accurate checking of same allele matching condition with Algorithm 1:

**Algorithm 1** Same Allele Match Elimination

**procedure** GETVIOLATIONS
**Input:** $Z1^{opt}$, $P_{Z1}{}^{opt}$, $Z2^{opt}$, $P_{Z2}{}^{opt}$, $ZZ1^{opt}$, $P_{ZZ1}{}^{opt}$, $ZZ2^{opt}$, $P_{ZZ2}{}^{opt}$
**Output:** *ConsistentChildList*, *ViolationChildList*
1 CVars_MC = $Z1^{opt} \cup ZZ1^{opt}$, CVars_FC = $Z2^{opt} \cup ZZ2^{opt}$
2 CPhases_MC = $P_{Z1}{}^{opt} \cup P_{ZZ1}{}^{opt}$, CPhases_FC = $P_{Z2}{}^{opt} \cup P_{ZZ2}{}^{opt}$
3 SortByIndex(CVars_MC, CPhases_MC)
4 SortByIndex(CVars_FC, CPhases_FC)
5 **FOR** varM in CVars_MC, phaseM in CPhases_MC, varF in CVars_FC, phaseF in CPhases_FC
6 **IF** varM.Index = varF.Index
7 **IF** IsHomozygous(varM)
8 **ADD** varM to *ConsistentChildList*
9 **ELSE IF** `phaseM` ≠ `phaseF`
10 **ADD** varM to *ConsistentChildList*
11 **ELSE IF** `varM` ∈ `Z1` **OR** `varF` ∈ `Z2`
12 **ADD** varM to *ConsistentChildList*
13 **ELSE**
14 **ADD** varM to *ViolationChildList*
15 **ENDIF**
16 next(varM), next(varF), next(phaseM), next(phaseF)
17 **ELSE IF** varM.Index < varF.Index
18 **ADD** varM to *ViolationChildList*
19 next(varM), next(phaseM)
20 **ELSE**
21 **ADD** varF to *ViolationChildList*
22 next(varF), next(phaseF)
23 **ENDIF**
24 **ENDFOR**

In Algorithm 1, in order to identify child variants that share an allele with both parents, we first merge the shared genotype and shared allele child variant sets by keeping the information of belonging sets for each variant at lines (1) and (2) of the pseudocode. Then we sort the merged child variant sets by variant indexes (order in VCF) at lines (3) and (4). At line (7), we check the condition where child variant is homozygous and same allele matching condition is ignored since both phasings can be used for child variant to break the same allele matching condition. At line (9), we check whether heterozygous child variants match with parents with different phasings. At line (11), we check if child variant matches to parent with both alleles so that alternative phasing can also be used to avoid same allele matching condition. We use *next* command to iterate to the following variant/phase at the line (16), (19) and (22). In the end, we obtain the list of Mendelian violations and consistent child variants for the input sets $Z1^{opt}$, $Z2^{opt}$, $ZZ1^{opt}$ and $ZZ2^{opt}$.

In Eqs. (5) and (6), reference alleles of child variants are ignored during maximization calculation. As a result, child variants that are matching one parent with non-reference allele and the other parent with reference allele are marked as Mendelian violation after processing variants with Algorithm 1. To identify and correct the decision of these child variants, we use the following equations:

$$K_{MOTHER} = \{r \in Z2^{opt} \cup ZZ2^{opt},\ A_r(1) = a_{REF} \vee A_r(2) = a_{REF:}\ h(X^{opt} \cup XX^{opt},\ P_X{}^{opt} \cup P_{XX}{}^{opt})\ [s_r.e_r] = Ref[s_r.e_r]\} \quad (7)$$

$$K_{FATHER} = \{r \in Z1^{opt} \cup ZZ1^{opt},\ A_r(1) = a_{REF} \vee A_r(2) = a_{REF:}\ h(Y^{opt} \cup YY^{opt},\ P_Y{}^{opt} \cup P_{YY}{}^{opt})\ [s_r.e_r] = Ref[s_r.e_r]\} \quad (8)$$

where $s_r$ and $e_r$ denote the start and end position of variant r and *Ref* denotes the reference sequence string. Indexing of the haplotype sequence is inherited from the reference sequence. $A_r(k)$ is the allele function that represents the allele of variant r with the phase selection of $k \in \{1, 2\}$ and $a_{REF}$ is the reference allele of a variant. $K_{MOTHER}$ and $K_{FATHER}$ are the sets of consistent child variants that share a reference allele with Mother and Father variants, respectively, and they are inserted into the set of consistent variants. The remaining unprocessed child variants (i.e. $C \backslash (Z1^{opt} \cup Z2^{opt} \cup ZZ1^{opt} \cup ZZ2^{opt})$) are inserted into the set of violations.

Once we obtain decisions of all child variants, we merge mother, father and child VCF as a trio by merging variants at the same position. Then, we apply three post-processing steps on the merged VCF:

1. Assign Mendelian decision to sites where the child has no variant (i.e. homozygous ref child variants in the merged trio). For each hom-ref child variant, final haplotype sequences of both mother and father are checked. If none of the parent haplotype sequences is equal to reference at the child variant's location, then the variant is marked as a violation.
2. Consolidate the decision for variants affecting the same position in the final haplotype sequence. Consistent VCF record decisions are changed to a violation if there is at least one overlapping violation VCF record.
3. Exclude sites where *nocall* is reported by at least one family member. *Nocall* variants are sites where insufficient information is available to determine genotypes, and they are usually represented as './.' at genotype (GT) column of VCF records.

## 2.3 Violation metrics

With naive line-by-line violation identification, a VCF entity is considered as a Mendelian violation if the genotypes of the mother, father and child variant call are inconsistent according to the Mendelian inheritance rules. After we extend the Mendelian identification from single entity comparison to multiple entities, the ambiguity of proper violation identification metrics arises.

In VBT, we use two different metrics to identify violations. The first method is identifying violations based on single entities which are similar to the naive approach. The difference is that, when multiple VCF records are affecting the same site, we unify their Mendelian decisions. For example, if child sample has a hom-alt deletion and parent samples have heterozygous SNPs in the range of deletion, naive method marks the hom-alt child deletion only as violation, whereas VBT marks all variants resides in the deletion site as violations.

In addition to the entity based violation identification, VBT also reports region based violations. In entity based evaluation, if a site

has a Mendelian violation, all VCF entities reside in that site are counted as violations.

| Chr | Pos | Ref | Alt | Format | Mother | Father | Child |
|---|---|---|---|---|---|---|---|
| 1 | 100 | GATAC | G, GA | GT | 0/2 | 0/0 | 0/1 |
| 1 | 102 | TAC | T | GT | 0/0 | 0/0 | 0/1 |
| 1 | 103 | A | C | GT | 0/0 | 1/1 | 0/0 |

In the above example, all three variants are counted as Mendelian violations using entity based metrics. However, all three variants could be represented at the same position and could be counted as a single violation. In that case, when comparing Mendelian violation rate of different variant calling pipelines, the variant caller which calls the above variants as a single entity would have the advantage. We eliminate this by dividing variant sets into small independent regions. We use *syncpoints* (Cleary *et al.*, 2015) which are genomic positions that delimit regions where variants within the two syncpoint can be processed independently. *Syncpoints* are incidental results of vcfeval and they are used to determine variant weightings as a post-processing stage. In VBT, we use syncpoints to divide variants into small regions where each region can contain one or more variants. If a region includes at least one Mendelian violation, we count that region as a violation and Mendelian consistent otherwise.

### 2.4 Evaluation methods and data

A truth set for trio analysis does not exist for direct result comparison. Instead, we use two alternative testing methods to compare VBT and existing tools. For our experiments, we use high coverage alignments of Central European (CEU) individuals (NA12878, NA12891 and NA12892) which are available at 1000 genomes phase 3 FTP server (ftp://ftp.1000genomes.ebi.ac.uk/vol1/ftp/phase3/data/).

For our first testing experiment, we construct trios from single individual samples by changing their variant representations. We use FreeBayes (Garrison and Marth, 2012) to generate unnormalized VCF files. Then we use Vt norm (Tan *et al.*, 2015) to alter variant representations of VCF. By using vcftools merge, we merge two identical unnormalized VCFs (playing the roles of mother and father samples) and one normalized VCF (playing the role of child sample). Since all trio samples belong to the same individual, we expect to see zero Mendelian violations by all Mendelian violation checking tools.

For the second testing experiment, we implement a Mendelian violation validator that checks all possible combinations of variant phasings in a given set of small regions. For this experiment, we use region-based assessment. As we discussed in violation metrics section, we want to count any number of mistakes has been made for a region as one since we do not want to penalize one tool to another due to representation differences. We obtain the regions by merging *syncpoints* yielded by mother–child and father–child variant comparisons. Our validation pipeline performs a comparative analysis of unique Mendelian violations between two Mendelian violation checking tools. That is why, we first select the regions where either VBT or naive tools find a Mendelian Violation. We ignore the regions in which both tools found no Mendelian violations.

For each selected region, we discard the variants that are marked as a Mendelian violation and, using the remaining variants; we seek for a Mendelian consistent combination. If no consistent combination can be found, the region is marked as *missing MV* for that tool. If a consistent combination can be found for both VBT and the naive method for a region, then we compare the reported violation counts for that region in order to check whether there are extra Mendelian violations reported. We accept the decision of tool with less Mendelian violation as correct and mark that region for the other tool as *extra MV*. A more detailed overview of our validation pipeline can be found in Supplementary Text, Section 3.

**Table 1.** Violation counts of different tools where the input trio is constructed from a single sample with different variant representations

| Input sample | VBT | Naive | PBT |
|---|---|---|---|
| NA12878 as trio | 0 | 76 867 | 72 111 |
| NA12891 as trio | 0 | 78 854 | 73 396 |
| NA12892 as trio | 0 | 76 176 | 72 674 |

*Note*: Naive check tools include RTG-mendelian, GATK -SelectVariants and Vcftools-mendel. For PhaseByTransmission (PBT), the sum of violation count and corrected genotype count is used in this table.

## 3 Results

VBT resolves variant representation differences in family trios efficiently by maximizing matching child variants with mother and father separately instead of using the ideal trio comparison function (eq. 2). This enables covering nearly all regions in datasets and provides VBT a reasonable running time, which varies between 6 and 8 min on Amazon c4.4xlarge instance (Intel Xeon E5 2.9 GHz, 16 vCPU, 30 GiB Memory) depending on complex region count for the whole human genome that contains 6.1 million vcf record.

In our first test scenario, we use the trios we generated from a single CEU sample to show that naive trio comparison tools produce wrong Mendelian decisions due to variant representations. We compared VBT, naive (line-by-line) Mendelian error checking tools (RTG-mendelian, GATK-SelectVariants, Vcftools-mendel) and PhaseByTransmission (PBT). For PBT, we used both $2 \times 10^{-2}$ and $10^{-8}$ as mutation rates and obtained the same number of *corrections* plus *mutations*. As seen in Table 1, VBT correctly outputs no violations for all three test data while the other tools output more than seventy thousand violations.

In the second experiment, we used CEU samples to compare trio concordance rate of different variant callers, FreeBayes (fb), Unified Genotyper (ug) and HaplotypeCaller (hc). In addition, we apply normalization (vt norm) on FreeBayes outputs and add it as fourth testing set to see if normalization can reduce errors of naive comparison tools. As a final testing trio, we produce gVCFs using HaplotypeCaller and jointly call them using GATK GenotypeGVCFs. We used vcftools (v0.1.14) to merge VCF files of individual samples except for the jointly called HaplotypeCaller trio VCF.

After we generate 5 trio VCFs using different variant calling pipelines, we ran VBT and naive checking tool for each trio and compare their result using our Mendelian violation validator. Table 2 shows the numbers of total violation regions, falsely identified violation regions and missed violations regions of the two methods, and for the five different variant calling pipelines. For all five testing pipeline, VBT has over 99% precision and recall values (Fig. 3). The precision of naive tools and VBT is closer for HaplotypeCaller and jointly-called HaplotypeCaller because the representations of called variants are more similar across the samples compared to other variant callers. It is also important to note

**Table 2.** Violation validation results of different variant calling pipelines using CEU trio

| | | VBT | | naïve | |
|---|---|---|---|---|---|
| Pipeline | Total MV Regions | Extra MV Regions | Missing MV Regions | Extra MV Regions | Missing MV Regions |
| **Fb** | 160 756 | 229 | 604 | 21 237 | 18 212 |
| **Ug** | 119 359 | 134 | 692 | 6457 | 16 986 |
| **Hc** | 72 671 | 516 | 229 | 32 | 6215 |
| **fb+norm** | 161 252 | 39 | 1052 | 23 077 | 15 380 |
| **hc joint** | 39 342 | 287 | 60 | 218 | 1600 |

*Note*: Comparison results of FreeBayes (fb), HaplotypeCaller (hc) and UnifiedGenotyper (ug). Autosomes only. No filtration is applied to the data. Regions may contain zero or multiple Mendelian violations.

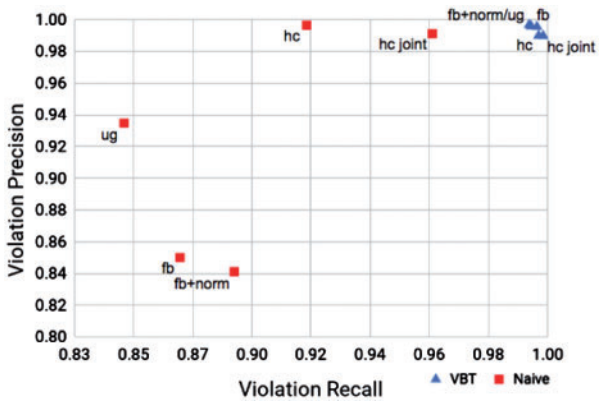


**Fig. 3.** Violation Precision versus Recall plot of VBT and Naive tools for 5 different variant calling pipelines. Recall is defined as *(Total MV Region—Extra MV—Missing MV)/(Total MV Region—Extra MV)* and precision is defined as *(Total MV Region—Extra MV—Missing MV)/(Total MV Region—Missing MV)* according to Table 2

that, total violation counts in experiment 2 depend on the variant caller, and for de novo mutation analysis, using a dedicated de novo caller may produce numbers closer to the expected counts.

## 4 Discussion

In this work, we presented VBT, a Mendelian violation detection tool that is capable of comparing complex indels in family trios. We showed with our test scenarios that, VBT has better accuracy than the existing tools.

We propose a trio comparison extension for pairwise comparison algorithm of vcfeval and we approximately solve this problem by processing mother–child and father–child haplotype sequence chains separately. VBT has two error modes which drop the precision and recall by a small amount visible on Table 2: First error mode is not knowing the 1-to-1 matches of variants across different samples. When we assess parent variant decisions using child decisions in the post-processing stage, we try to match parent and child variants using a position overlap [i.e. used in Eq. (7) and Eq. (8)] which leads to an error in repetitive regions. Our approximation to the trio comparison problem introduces the second error mode. Obtained local best paths from mother–child and father–child duos are not always identical to the global optimum of the trio, and in the merging step, this causes a small error rate. We further explain the error modes with examples in Supplementary Text, Section 3.4. To solve the second error mode of VBT, direct implementation of our initial method [i.e. Eq. (2)] could be used. However, the first error mode also remains as a problem for this method and even worse sensitivity could be shown due to the larger number of variants required to be post-processed after Eq. (2).

VBT's accuracy can further be improved by correcting wrong/missed decisions by comparing vcf output with the naive comparison as a post-processing step. In regions where the naive method and VBT disagrees, a nonlinear violation check can be performed by generating all possible subsequences for that region, similarly to our violation validation pipeline. This would not increase overall running time considerably because the slow nonlinear checking method would be invoked only for regions where the naive method disagrees with VBT. As a result, VBT would serve as a cost-efficient detector informing us whenever the naive comparison methods are not enough.

VBT does not alter variant representations during or after variant comparison. Instead, we keep the original variant representations and add additional info tag that whether a variant is a Mendelian violation or not. The advantage of this is the ability to track variants for benchmarking purposes. The disadvantage is that existing tools which require trio analysis such as PhaseByTransmission and DeNovoGear are not able to use VBT output directly and need to read Mendelian decision annotation from output vcf records. The reason why we do not alter variant representations (i.e. merge multiple rows as a single entity) is the ambiguity in violation regions since the correct phasings of variants are unknown and several merging options are possible. Therefore, for de novo mutation identification tools, we add an input mode (–output-violation-regions) that allows VBT to output all Mendelian violation regions as a BED file. Using this BED file, we can identify the violation regions missed by naive the method. Besides, variants in these violation regions then can be either merged with a custom choice or all combinations can be generated for de Novo mutation analysis.

## Acknowledgements

We would like to thank Maxime Huvet, Maria C. Suciu and Amit Jain for trio benchmarking discussions and thank Sun-Gou Ji, Yilong Li, Morten Källberg, Gülfem Demir, James Spencer and Kaushik Ghose for their valuable comments on VBT.

## Funding

This work was supported in part by UK Department of Health grant SBRI Genomics Competition: Enabling Technologies for Genomic Sequence Data Analysis and Interpretation administered by Genomics England.

*Conflict of Interest*: none declared.

## References

Genome Project Consortium *et al.* (2015) A global reference for human genetic variation. *Nature*, **526**, 68–74.

Cleary,J.G. *et al*. (2015) Comparing variant call files for performance benchmarking of next-generation sequencing variant calling pipelines. *bioRxiv*, doi: 10.1101/023754/

Conrad,D.F. *et al*. (2011) Variation in genome-wide mutation rates within and between human families. *Nat. Genet.*, **43**, 712–714.

Danecek,P. *et al*. (2011) The Variant Call Format and VCFtools. *Bioinformatics*, **27**, 2156–2158.

Deciphering Developmental Disorders Study (2017) Prevalence and architecture of de novo mutations in developmental disorders. *Nature*, **542**, 433–438.

Douglas,J.A. *et al*. (2002) Probability of detection of genotyping errors and mutations as inheritance inconsistencies in nuclear-family data. *Am. J. Hum. Genet.*, **70**, 487–495.

Francioli,L.C. *et al*. (2017) A framework for the detection of de novo mutations in family-based sequencing data. *Eur. J. Hum. Genet.*, **25**, 227–233.

Garrison,E. and Marth,G. (2012) Haplotype-based variant detection from short-read sequencing. *arXiv*, arXiv: 1207.3907

Hidalgo,R.A. *et al*. (2016) New insights into the generation and role of de novo mutation in health and disease. *Genome Biol.*, **17**, 241.

Jamuar,S.S. and Tan,E.C. (2015) Clinical application of next generation sequencing for Mendelian diseases. *Hum. Genomics*, **9**, 10.

Kong,A. *et al*. (2012) Rate of de novo mutations and the importance of father's age to disease risk. *Nature*, **488**, 471–475.

Komar,P. and Deniz,K. (2017) geck: trio-based comparative benchmarking of variant calls. *bioRxiv*, doi: 10.1101/208116

Krusche,P. *et al*. (2018) Best practices for benchmarking small variant calls in human genomes. *bioRxiv*, doi: 10.1101/270157

McKenna,A. *et al*. (2010) The Genome Analysis Toolkit: a MapReduce framework for analyzing next-generation DNA sequencing data. *Genome Res.*, **20**, 1297–1303.

Nutsua,M.E. *et al*. (2015) Family-based benchmarking of copy number variation detection software. *PLoS One*, doi: 10.1371/journal.pone.0133465

Pilipenko,V.V. *et al*. (2014) Using Mendelian inheritance errors as quality control criteria in whole genome sequencing dataset. In *BMC Proceedings*, Vol. 8, pp. 1.

Rakocevic,G. *et al*. (2017) Fast and accurate genomic analyses using genome graphs. *bioRxiv*, doi: 10.1101/194530

Ramu,A. *et al*. (2013) DeNovoGear: de novo indel and point mutation discovery and phasing. *Nat. Methods*, **10**, 985–987.

Sun,C. and Medvedev,P. (2016) VarMatch: robust matching of small variant datasets using flexible scoring schemes. *Bioinformatics*, **33**, 1301–1308.

Talwalkar,A. *et al*. (2014) SM A SH: a benchmarking toolkit for human genome variant calling. *Bioinformatics*, **30**, 2787–2795.

Tan,A. *et al*. (2015) Unified representation of genetic variants. *Bioinformatics*, **31**, 2202–2204.

Wei,Q. *et al*. (2015) A Bayesian framework for de novo mutation calling in parents-offspring trios. *Bioinformatics*, **31**, 1375–1381.

Zook,J.M. *et al*. (2014) Integrating human sequence data sets provides a resource of benchmark SNP and indel genotype calls. *Nat. Biotechnol.*, **32**, 246–251.