Best Practices for Benchmarking Germline Small Variant Calls in Human Genomes

Authors

Peter Krusche[1], Len Trigg[2], Paul C. Boutros[3], Christopher E. Mason[4], Francisco M. De La Vega[5], Benjamin L. Moore[1], Mar Gonzalez-Porta[1], Michael A. Eberle[6], Zivana Tezak[7], Samir Lababidi[8], Rebecca Truty[9], George Asimenos[10], Birgit Funke[11], Mark Fleharty[12], Marc Salit[13,*], Justin M Zook[14,*], and the Global Alliance for Genomics and Health Benchmarking Team

[1]Illumina Cambridge Ltd, Chesterford Research Park, Little Chesterford, Saffron Walden, Essex, CB10 1XL, UK
[2]Real Time Genomics, Level 1, South Bloc, 19 Knox St. Hamilton. New Zealand
[3]Ontario Institute for Cancer Research, Toronto, Ontario, Canada
[4]Weill Cornell Medicine, New York, NY
[5]Department of Biomedical Data Science, Stanford University School of Medicine, Stanford, CA
[6]Illumina Inc., 5200 Illumina Way, San Diego, CA 92122
[7]Center for Devices and Radiological Health, FDA, Silver Spring, MD
[8]Office of Health Informatics, Office of the Commissioner, FDA, Silver Spring, MD
[9]Invitae, 1400 16th St, San Francisco, CA 94103
[10]DNAnexus, 730 Market St Suite 2100, San Francisco, CA 94103
[11]Veritas Genetics, 99 Conifer Hill Dr, Danvers, MA 01923
[12]Broad Institute, 415 Main Street, Cambridge, MA 02142
[13]Material Measurement Laboratory, Joint Initiative for Metrology in Biology, National Institute of Standards and Technology, Stanford, CA
[14]Material Measurement Laboratory, National Institute of Standards and Technology, 100 Bureau Dr, MS8301, Gaithersburg, MD 20899

*These authors contributed equally to this work
Contact Information: Justin Zook, jzook@nist.gov

Abstract

Assessing accuracy of NGS variant calling is immensely facilitated by a robust benchmarking strategy and tools to carry it out in a standard way. Benchmarking variant calls requires careful attention to definitions of performance metrics, sophisticated comparison approaches, and stratification by variant type and genome context. The Global Alliance for Genomics and Health (GA4GH) Benchmarking Team has developed standardized performance metrics and tools for benchmarking germline small variant calls. This team includes representatives from sequencing technology developers, government agencies, academic bioinformatics researchers, clinical laboratories, and commercial technology and bioinformatics developers for whom benchmarking variant calls is essential to their work. Benchmarking variant calls is a challenging problem for many reasons:

- Evaluating variant calls requires complex matching algorithms and standardized counting because the same variant may be represented differently in truth and query callsets.
- Defining and interpreting resulting metrics such as precision (aka positive predictive value = TP/(TP+FP)) and recall (aka sensitivity = TP/(TP+FN)) requires standardization to draw robust conclusions about comparative performance for different variant calling methods.
- Performance of NGS methods can vary depending on variant types and genome context; and as a result understanding performance requires meaningful stratification.
- High-confidence variant calls and regions that can be used as “truth” to accurately identify false positives and negatives are difficult to define, and reliable calls for the most challenging regions and variants remain out of reach.

We have made significant progress on standardizing comparison methods, metric definitions and reporting, as well as developing and using truth sets. Our methods are publicly available on GitHub (https://github.com/ga4gh/benchmarking-tools) and in a web-based app on precisionFDA, which allow users to compare their variant calls against truth sets and to obtain a standardized report on their variant calling performance. Our methods have been piloted in the precisionFDA variant calling challenges to identify the best-in-class variant calling methods within high-confidence regions. Finally, we recommend a set of best practices for using our tools and critically evaluating the results.

Introduction

Next generation sequencing (NGS) technologies and analysis methods have rapidly evolved and are increasingly being used in research and clinical settings. The ability to detect DNA variants began in the last third of the 20th century when recombinant DNA technology facilitated the identification and characterization of human genes. Due to the high cost of sequencing technologies, early diagnostic applications were limited to screening patient samples for established pathogenic variants. Clinical heterogeneity and overlapping presentations can complicate accurate diagnosis based on clinical symptoms alone, which often resulted in the need for sequential testing approaches (diagnostic odysseys). While some focused tests are still in use today, large gene panels including tens to hundreds of genes, often accommodating sets of diseases with clinical overlap, are the most common application for NGS today, with exome and genome sequencing rapidly gaining popularity in the research and medical genetics communities.[1,2] An output of these tests is a list of variant calls and their genotypes, often in variant call format (VCF), and benchmarking these calls is an important part of analytical validation.

Robust, sophisticated, and standardized benchmarking methods are critical to enable development, optimization, and demonstration of performance for these sequencing and analysis tools. This is especially important for clinical laboratories developing sequencing based tests for medical care. Efforts such as the *Genome in a Bottle Consortium* and *Platinum Genomes Project* have developed small variant "truth" sets for several well-characterized human genomes from publicly available cell lines and DNA.[3–6] A "truth" set was also recently developed from a "synthetic-diploid" mixture of two haploid hyditaform mole cell lines that are not currently in a public repository.[7] A framework for benchmarking simple variant calls in the exome was developed previously as a web-based tool GCAT.[8] However, comparing variant calls from any particular sequencing pipeline to a truth set is not a trivial exercise. First, benchmarking must consider that variants may be represented in multiple ways in the commonly used variant call format (VCF).[9–12] When comparing VCF files record by record, many of the putative differences are simply different representations of the same variant. Secondly, definitions for performance metrics such as true positive (TP), false positive (FP), and false negative (FN), which are key for the interpretation of the benchmarking results, are not yet standardized. Lastly, due to the complexity of the human genome, performance can vary across variant types and genomic regions, which inevitably increases the number of benchmarking statistics to report.

In the context of performance metrics, two critical performance parameters that are traditionally required for clinical tests are sensitivity (the ability to detect variants that are known to be present or "absence of false negatives", which we call recall in this work) and specificity (the ability to correctly identify the absence of variants or "absence of false positives", which we replace with "precision" in the work).[13] The shift from focused genotyping tests to DNA sequencing enables the detection of novel sequence variants, which has fundamental implications on how these diagnostic performance parameters need to be determined. Early professional guidelines call for the use of samples with and without known pathogenic variants to determine sensitivity and specificity, which was appropriate when genetic testing interrogated only targeted, previously

identified variants. This approach remains valid for sequencing based testing, but now constitutes an incomplete evaluation, since it does not address the ability to detect novel variants. To predict performance for novel variants, it is important to maximize the number and variety of variants that can be compared to a "gold standard" in order to establish statistical confidence values for different types of variants and genome contexts, which can then be extrapolated to all sequenced bases.[14–17] While this problem already existed for Sanger sequencing tests, the power and scope of NGS technologies presents a different scale of challenges for fit-for-purpose test validation. Laboratories that performed Sanger sequencing prior to transitioning to NGS were often able to utilize previously analyzed specimens to establish analytical performance of NGS tests; however, this approach is practically limiting, poses severe challenges for other laboratories, and is completely infeasible as test sizes increase from a few genes to the exome or genome. Guidelines were recently published for validating clinical bioinformatics assays.[18] These guidelines highlight the utility of reference materials for benchmarking variant calls, as well as the importance of stratifying performance by variant type and genome context.

To address the needs for using reference materials to benchmark variant calls in a standardized, robust manner, we present the work of the Global Alliance for Genomics and Health (GA4GH) Benchmarking Team. This team, open to all interested parties, includes broad stakeholder representation from research institutes and academia, sequencing technology companies, government agencies, and clinical laboratories, with the common goal of driving towards the standardisation of variant calling benchmarking. In particular, we describe the available reference materials and tools to benchmark variant calls, and provide best practices for using these resources and interpreting benchmarking results.

Comparison Method Standardization

Our goal was to standardize the variant benchmarking process such that (1) the methods used to compare callsets assess the accuracy of the variant and genotype calls independent of different representations of the same variant, (2) primary performance metrics are represented in the most commonly used binary classification form (i.e., TP, FP, FN, and statistics derived from these), (3) calculation of performance metrics is standardized such that they can be compared more easily across methods, and (4) performance metrics can be stratified by variant type and genome context.

In this section we discuss the technical challenges presented by comparing VCF files accurately and describe our solution to implement such comparisons. We focus on the use case where we have a call set that can be used as "truth" (e.g. Genome in a Bottle or Platinum Genomes) and would like to benchmark a single-sample query VCF against this dataset. The inputs to this comparison are a *truth callset* (in VCF format), and a set of *confident regions* (in BED format) for the truth set. The confident regions indicate the locations of the genome where, when comparing to the truth callset, variants that do not match the truth callset should be false positives and variants missed in the truth callset should be false negatives. Furthermore, our inputs include a

query callset in (g)VCF format, a reference FASTA file and optionally stratification and subset regions to break out variant calling performance in particular regions of the genome or to restrict comparisons to a genomic subset (e.g. exons / regions captured by targeted sequencing). For more details see SI A.

We developed a framework for standardized benchmarking of variant calls (Fig. 1), which addresses the challenges discussed in detail in the following sections.

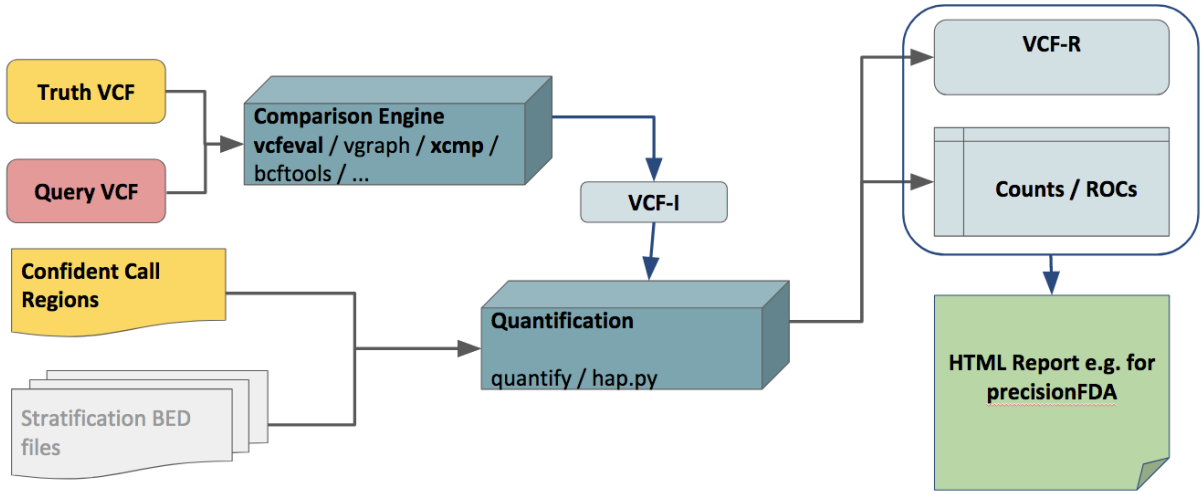


Figure 1: The GA4GH Benchmarking Team's reference implementation of a comparison framework. The framework takes in a Truth VCF, Query VCF, confident call regions for the Truth and/or Query, and optionally BED files to stratify performance by genome context. A standardized intermediate output (VCF-I) from the comparison engines allows them to be interchanged and for TP, FP, and FN to be quantified in a standard way.

Variant representation

The challenge with comparing two or more VCF files is handling complex variant representations correctly. In a VCF file, we describe two haplotype sequences by means of REF-ALT pairs and genotypes. These variant calls do not always uniquely represent the same haplotype sequences. Alignments are not always unique even when using a fixed set of gap and substitution scores; different variant calling methods may produce different variant representations. While some of these differences can be handled using pre-processing of VCF files (*e.g.* variant trimming and left-shifting), others cannot be fixed easily. As a result we cannot compare VCF files accurately by comparing VCF records and genotypes directly. Approaches were developed to standardize indel representation by means of left-shifting and trimming the indel alleles.[19,20] These methods determine the left-most and right-most positions at which a particular indel could be represented in a VCF file (Fig. 2a). These methods work well when considering each VCF record independently. However, when multiple VCF records are used to represent a complex haplotype, more sophisticated comparison methods are required (Fig. 2b-d).

Variant representation differences broadly fall into the following categories.

1. *Reference-trimming of alleles*: Variant alleles may include reference bases at the beginning or at the end. This is required to represent insertions in a VCF file, which always include a single padding base (the reference base just before the insertion), and is also used by most variant calling methods to represent deletions. Some methods add reference padding to explicitly assert surrounding bases have been observed to be homozygous reference.
2. *Left-shifting and right-shifting of alleles*: variant calls in repetitive sequence may have different possible starting positions after trimming. The simplest example of this scenario is change in length in a homopolymer: this can be represented as an insertion/deletion which may be anchored at any position within the homopolymer (Fig 2a). Interestingly, the unstated convention for VCF is that variants are left-aligned, whereas HGVS guidelines state that variants should be right-aligned in the context of the transcript (i.e., represent the 3' most possibility), which may be either left or right aligned in the genomic context depending on which strand the gene is encoded on. More complex repeats lead to additional complexity.[9–11]
3. *Allele decomposition and phasing*: Complex alleles may be represented as a single VCF record, or split into multiple records with different starting positions. An insertion can often be represented as one large insertion or multiple smaller insertions (Fig 2b). Similarly, multi-nucleotide polymorphisms (MNPs) can be represented either as a single MNP record, or alternatively as separate SNVs (Fig 2c). The MNP representation may be chosen since it explicitly encodes variant phasing and makes it clear that two SNVs have occurred on the same haplotype, which can have a significant impact on clinical interpretation, as illustrated in Supplementary Figure 1. In the SNV representation the same information may be encoded using the PS format field in the VCF file, but the practical implementation varies between different variant calling methods. Also, although including local phasing is strongly encouraged, phasing information may not be present in the VCF file, in which case we can match the MNP to the SNVs by decomposing it, but we cannot recompose the SNVs into the MNP if they are unphased and heterozygous.
4. *Generic ambiguous alignment*: Cases 1 and 2 are specific examples for ambiguous alignments between reference sequence and the observed haplotype in a sample. However, these are not the only case where different sequence alignments with the same maximum score may exist. In low-complexity sequence, more difficult scenarios may arise. One example is the case where we have a SNV adjacent to a deletion. The SNV may be called either at the start or at the end of the deletion, and the choice is up to the variant caller or the aligner. An example is shown in Fig. 2d. Complex variants in repetitive regions often cannot be "normalized" (i.e., converted into a standard representation) by any existing tools, so that sophisticated variant comparison tools like those developed in this work are necessary.

When benchmarking, the above variant representation differences can also give rise to different notions of giving partial credit for variant calls. One example is where we may have called only one SNV in an MNP with the correct genotype. When assigning TP/FP/FN status on a per-VCF-

record basis, a variant caller that chooses to represent calls using single MNP records would not get credit for calling this SNV correctly since the overall MNP record does not reproduce the correct haplotype. Another example would be phasing switch-errors: a choice needs to be made whether to use phasing-aware benchmarking for a particular evaluation. Handling these cases is important since adding phasing information provides additional information to the users of a variant caller, but may lead to FPs / FNs when running a benchmarking comparison when comparing to a method which does not provide phasing information and outputs all alleles in decomposed form for maximum credit in the benchmarking comparison.

A variety of approaches have been recently developed to address the challenges in variant representation.[9–11,21,22] Real Time Genomics (RTG) developed the comparison tool *vcfeval*, which introduced the idea of comparing variants at the level of the genomic haplotypes that the variants represent as a way to overcome the problems associated with comparing complex variants, where alternative yet equivalent variant representations can confound direct comparison methods.[9] Variant "normalization" tools help to represent variants in a standardized way (e.g., by left-shifting indels in repeats), but they demonstrated that "variant normalization" approaches alone were not able to reconcile different representations of many complex variants. In contrast, global optimization permits evaluation of alternate representations that minimize the number of discrepancies between truth and test set caused by differences in representations of the same variant. Similarly, *VarMatch* was developed to resolve alternate representations of complex variants, with additional ability to tune the matching parameters depending on the application.[10] Finally, *hap.py* includes a comparison tool to perform haplotype-based comparison of complex variants in addition to sophisticated functionality to stratify variant calls by type or region.[21]

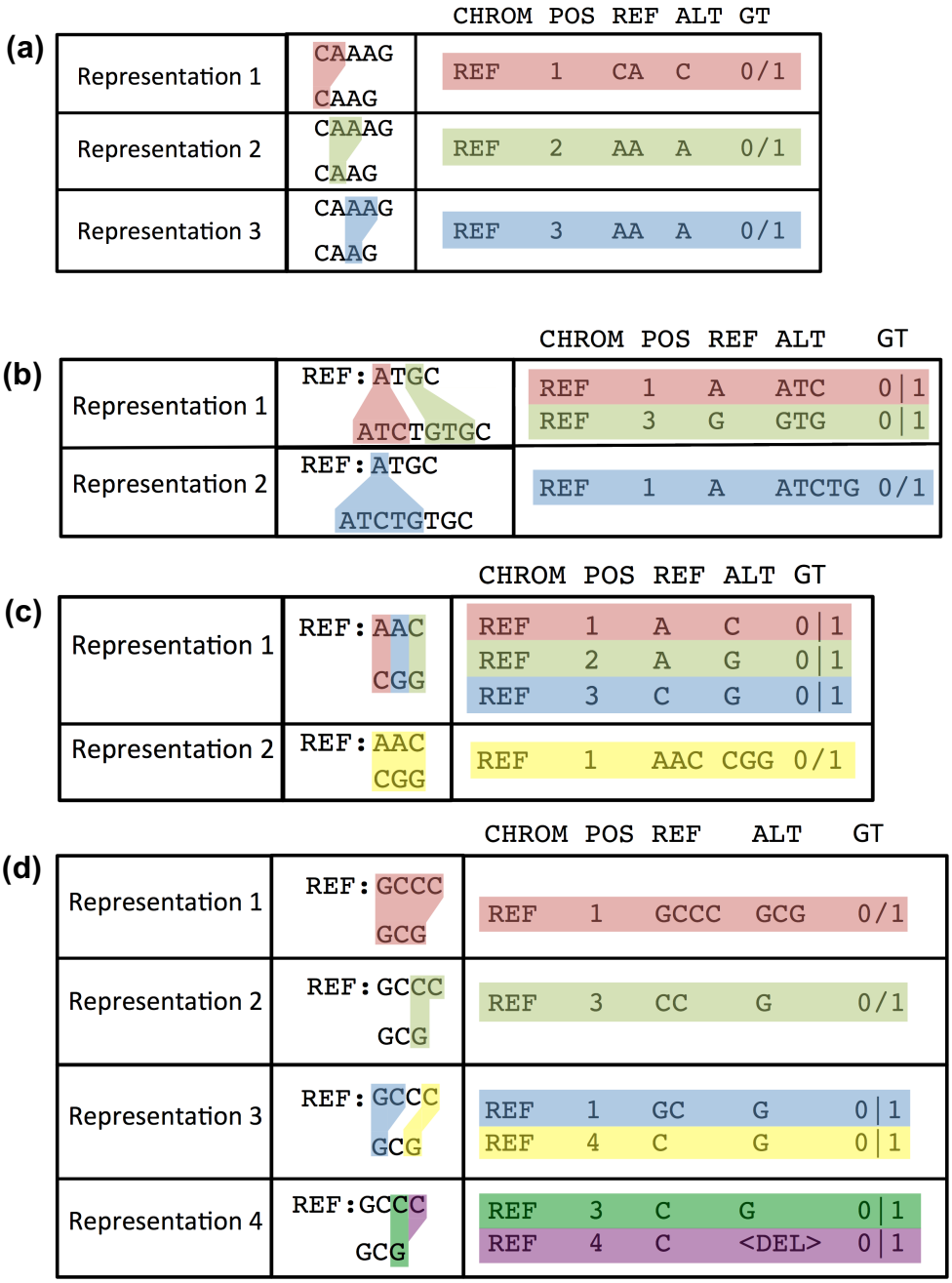


Fig. 2: Four examples of cases where variants can be represented in multiple forms in VCF format. (a) Three representations of a deletion in a homopolymer. (b) The insertion can be represented as one 4-bp insertion or two 2-bp insertions. (c) An MNP can be represented as 3 SNVs or one larger substitution. (d) Four different representations of a complex variant. Note that representations include phasing information in these examples where it is necessary to unambiguously describe the variant. If phasing was not described for these variants, it would impossible to normalize their representations, but our sophisticated variant comparison tools can determine that they could describe the same two haplotypes.

Variant counting

The GA4GH Benchmarking Team developed consensus definitions and recommendations for expressing performance metrics for small germline variant calls. Assessing the performance of variant callers does not easily lend itself to the typical binary classification performance assessment model of simply determining true and false "positives" and “negatives”. Several characteristics of the genome do not fit well in a binary classification model:

1. More than two possible genotypes exist at any given location. For SNVs (if ignoring phasing), any location can have one of 10 different true genotypes (*i.e.*, A/A, A/C, A/G, A/T, C/C, C/G, …). For indels and complex variants, an infinite number of possible genotypes exists (*e.g.*, any length of insertion).
2. A number of variant callers distinguish between "no-calls" and homozygous reference calls at some genome positions or regions. Some variant callers even output partial no-calls, calling one allele but not the other. “No-calls” at a true variant site could be treated as false negatives or be excluded from counting.
3. In addition to the challenges comparing different representations of complex variants (*i.e.*, nearby SNVs and/or indels) discussed above, there are challenges in standardizing counting of these variants. Complex variants can be treated as a single positive event or as multiple distinct SNV and indel events when counting the number of TP, FP, and FN variants. In addition, only part of a complex variant may be called, which poses challenges in defining TP, FP, and FN.
4. Methods for assessing accuracy of phasing have not been fully developed or standardized, but accurate phasing can be critical, particularly when multiple heterozygous variants exist in a small region (*e.g.*, complex variants).

Matching Stringencies

Due to the inherent complexity of the human genome, TP, FP, and FN can be defined in different ways. Our reference implementation for benchmarking uses a tiered definition of variant matches, a standardized VCF format for outputting matched variant calls, and a common counting and stratification tool (see SI A).

We consider the following types of variant matches from most to least stringent, with “Genotype match” being used by our current tools to calculate TP, FP, and FN:

- *Genotype match*: Variant sets in truth and query are considered TPs when their unphased genotypes and alleles can be phased to produce a matching pair of haplotype sequences for a diploid genome. Each truth (and query) variant may be replayed onto one of two truth (or query) haplotypes. A maximal subset of variants that is replayed to produce matching haplotype sequences forms the TP variants, query variants outside this set are FP, truth variants outside this set are FN. The method only considers haploid or diploid samples but could be extended to higher ploidy also. Enumerating the possible assignments for haplotype generation is computationally expensive. Vcfeval solves this problem using

global optimization methods supplemented with heuristic pruning. Genotype match statistics are the default TP, FP, and FN output by our tools. Genotype matching has been implemented in the *hap.py* tool *xcmp* and in *vcfeval*.

- *Allele match*: Truth and query alleles are counted as TP_AM if they contain any of the same (trimmed and left-shifted) alleles. This method is more specific than local matching (e.g. repeat expansions must be called with the correct length in order to get an allele match), but could also be susceptible to spurious mismatches when truth and query variant alleles are decomposed differently. Genotype mismatches (FP.GT in Table 1) are considered TPs in this matching method. We indicate allele matches in scenarios where variants can be matched when ignoring the genotype. Allele match statistics (TP_AM, FP_AM, and FN_AM) can be calculated from the GA4GH outputs (which require genotypes to match): TP_AM=QUERY.TP+FP.GT; FP_AM=QUERY.FP-FP.GT; FN_AM=TRUTH.FN-FP.GT. Allele matching has been implemented in the *hap.py* tool *scmp-somatic* and in *vcfeval* with the --squash-ploidy option.
    - Note that *vcfeval* --squash-ploidy and *scmp-somatic* differ. *scmp-somatic* checks if the VCF records give the same alleles after normalization and trimming. This will match alleles that overlap on the reference as long as they can be matched directly after left-shifting and trimming. When comparing somatic variant calls, this is probably the best option since technically, every variant could be on a different (low-frequency) haplotype. *vcfeval* --squash-ploidy does haplotype-based comparison but assumes all variants are hom-alt and there is only one haplotype. This will match different representations unless they overlap on the reference (which is also possible using *xcmp* via the force-gt command line option in *hap.py* which changes the GTs before comparing).
- *Local match*: Truth and query variants are counted as TP_LM if their reference span intervals are closer than a pre-defined local matching distance, i.e. all yellow categories in Table 1 are considered TPs, including “F” matches that are within a specified number of basepairs. This approach has previously been implemented.[7,21] An advantage of this matching method is that it is robust towards representational differences. A drawback for many applications is that it does not measure allele or genotype accuracy. We use local matches as the lowest tier of matching to label variants which are close-by but cannot be matched with other methods. Local match statistics (TP_LM, FP_LM, and FN_LM) can be calculated from the GA4GH outputs (which require genotypes to match): TP_LM=QUERY.TP+FP.GT+FP.AL; FP_LM=QUERY.FP-FP.GT-FP.AL; FN_LM=TRUTH.FN-FP.GT-FP.AL. If only local matching is required, this has been implemented in the *hap.py* tool *scmp-distancebased.*

A fourth, most stringent matching, which is not yet fully implemented in the GA4GH framework, requires phasing information to match:

- *Phased genotype match*: When VCF files specify phasing information, we can compare on a haplotype level: variants will only be matched if they produce matching haplotype sequences under phasing constraints. Both vcfeval and hap.py’s xcmp method support phased matching when both callsets include variants that are globally phased (*i.e.* specify a paternal and maternal haplotype for each chromosome). To our knowledge, no current

comparison method supports phasesets and local phasing to compare variants. Moreover, assessing phasing requires us to consider not only phasing variant accuracy, but also completeness of phasing coverage. In our current methods we do not implement phased genotype matching beyond the basic support provided by vcfeval and xcmp.

Table 1: Contingency table describing the GA4GH definitions of true positive (TP), false positive (FP), false negative (FN), allele mismatch (FP.AL), genotype mismatch (FP.GT), and unknown (UNK). Matches counted as FP.GT and FP.AL are additionally counted as both FP and FN, since our tool's default matching stringency requires genotypes to match. Query variants outside the Truth bed file are counted as UNK.

		Truth				
	Genotype	ref/ref	ref/var1	var1/var2	var1/var1	Outside bed
Query	ref/ref	-	FN	FN	FN	-
	ref/var1	FP	TP	FP.GT	FP.GT	UNK
	ref/var2	-	FP.AL	FP.GT	FP.AL	-
	ref/var3	-	-	FP.AL	-	-
	var1/var2	FP	FP.GT	TP	FP.GT	UNK
	var1/var3	-	-	FP.GT	-	-
	var2/var3	-	FP.AL	FP.GT	FP.AL	-
	var3/var4	-	-	FP.AL	-	-
	var1/var1	FP	FP.GT	FP.GT	TP	UNK
	var2/var2	-	FP.AL	FP.GT	FP.AL	-
	var3/var3	-	-	FP.AL	-	-

Defining True Positives, False Positives, and False Negatives

In Table 1, we enumerate the types of matches that are clear TP, FP, and FN as well as various kinds of partial matches that may be considered TP, FP, and/or FN depending on the matching stringency, and how they are counted by our tools. Our tools calculate TP, FP, and FN requiring the genotype to match, but output additional statistics related to how many of the FPs and FNs are allele matches (FP.GT) or local matches (FP.AL). Note that we have chosen not to include true negatives (or consequently specificity) in our standardized definitions. This is due to the challenge in defining the number of true negatives, particularly around complex variants. In addition, precision is often a more useful metric than specificity due to the very large proportion of true negative positions in the genome.

Another key question is how to count both matching and mismatching variant calls when they are differently represented in the truth dataset and a query. When representing MNPs as multiple

SNVs, we may count one variant call for each SNV, or only one call in total for the MNP record. Similar considerations apply to counting complex records. We approach variant counting as follows:

- We count the truth and query VCF files separately. A set of truth records may be represented by a different set of query records.
- To get comparable recall, we count both TPs and FNs in their truth representation. When comparing different variant calling results to the same truthset, these counts will be based on the same variant representation.
- Precision is assessed using the query representation of variants. We give a relative precision to the number of truth variants in query representation. If a variant caller is consistent about the way it represents variants, this approach mitigates counting-related performance differences.
- We implement a "partial credit" mode in which we trim, left-shift and decompose all query variant calls before comparison. This resolves the MNP vs. SNV comparison issues and also simplifies the variant types we use for stratification, rather than having a category of complex variant calls which has results that are difficult to interpret, we account for every atomic indel and SNV call independently.
- Variants are stratified into a canonical set of types and subtypes (see SI B).
- When stratification regions are applied, we match variants by their trimmed reference span. If any part of a deletion overlaps the stratification region, it is counted as part of that stratum. Insertions receive special treatment by requiring both the base before and the base after to be captured. Importantly, this stratification is performed after comparison to deal appropriately with representation issues.

Benchmarking metrics report

To reconcile the comparison methods and metrics discussed above into a simple summary, we have implemented in *hap.py* a standardized report that can be generated from the tabular output of the benchmarking workflow.[21] This report displays the metrics we believe are most important in an accessible fashion (Tier 1 metrics), while also allowing to examine the data in more detail (Tier 2 metrics). An example for the metrics and plots displayed in such a report is shown in Fig. 3.

From the TP, FP, and FN counts defined in Table 1, we calculate:

METRIC.PRECISION = QUERY.TP / (QUERY.TP + QUERY.FP)
METRIC.RECALL = TRUTH.TP / (TRUTH.TP + TRUTH.FN)

We use the count of TPs based on the query representation to calculate precision, and we use the count of TPs based on the Truth representation to calculate recall, in order to account best for cases where the Truth may tend to split a complex variant into multiple varaints and the Query may combine them into a single variant, or vice versa. Definitions and formulas for all performance metrics are detailed in Supplementary Table 1.

An alternative to precision is false positive rate (FPR) = FP / megabase.[7] It can easily be obtained from GA4GH/hap.py extended csv by taking FP / 1e6 * Subset.Size (or Subset.IS_CONF.Size, the number of confident bases in each stratification region). Precision approximates the probability that a given query call is true, while FPR approximates the probability of making a spurious call. Note that we do not define “True negatives” or “specificity” because these are not cleanly applicable to genome sequencing. For example, there are an infinite number of possible indels in the genome, so there are an infinite number of true negatives for any assay.

In addition, the GA4GH Benchmarking framework is able to produce precision-recall curves, which are graphical plots that illustrate the performance of a variant quality score of a test call set as its discrimination threshold is varied, compared to the reference call set (see Figure 4). The curve is created by plotting the precision against the recall at various quality score threshold settings. Commonly used quality scores include QUAL, GQ (genotype quality), DP (depth of coverage), and machine-learning derived scores such as VQSLOD and AVR. Because some methods use multiple annotations for filtering, precision-recall curves can be generated for a particular quality score either before or after removing filtered sites. Examining the precision-recall curves for various call-sets has two main advantages. Firstly, it allows the user to consider how accuracy is affected through the precision/recall trade-off. Secondly, different call sets may have effectively selected different precision/recall trade-off criteria, so simply comparing full call set metrics may reflect more about the different trade-off points than the call sets themselves at some shared trade-off criteria.

In the following section, we will discuss example comparisons performed using the standardized benchmarking workflow and comparison methods.

(a) SNPs

Show additional metrics | Show ALL calls | Filter Rows

Method	Comparison	Recall	Precision	Frac. NA	F-Score	#QUERY	TP	FN	FP
BWAGATKfilt	hs37d5-vcfeval-nopartialcredit	99.52%	99.92%	14.36%	0.997	3731816	3193768	15548	2423

INDELS

Show additional metrics | Show ALL calls | Filter Rows

Method	Comparison	Recall	Precision	Frac. NA	F-Score	#QUERY	TP	FN	FP
BWAGATKfilt	hs37d5-vcfeval-nopartialcredit	98.84%	99.19%	44.31%	0.990	862169	475993	5563	3870

(b) SNPs

Hide additional metrics | Show ALL calls | Filter Rows

Method	Comparison	Subset	Subtype	Recall	Precision	Frac. NA	F-Score	#TRUTH	#QUERY	TP	FN	FP	GT Mismatches	Allele Mismatches	UNK
BWAGATKfilt	hs37d5-vcfeval-nopartialcredit	*	*	99.52%	99.92%	14.36%	0.997	3209316	3731816	3193768	15548	2423	159	0	535779
BWAGATKfilt	hs37d5-vcfeval-nopartialcredit	TS_boundary	*	98.65%	82.56%	95.80%	0.899	74	2047	73	1	15	12	0	1961
BWAGATKfilt	hs37d5-vcfeval-nopartialcredit	TS_contained	*	99.52%	99.92%	0.00%	0.997	3209242	3195951	3193695	15547	2408	147	0	0
BWAGATKfilt	hs37d5-vcfeval-nopartialcredit	lowcmp_AllRepeats_51to200bp_gt95identity_merged	*	99.63%	99.81%	70.21%	0.997	4318	14468	4302	16	8	4	0	10158
BWAGATKfilt	hs37d5-vcfeval-nopartialcredit	lowcmp_AllRepeats_gt200bp_gt95identity_merged	*	0.00%	(n/a)	100.00%	(n/a)	0	9678	0	0	0	0	0	9678

(c)

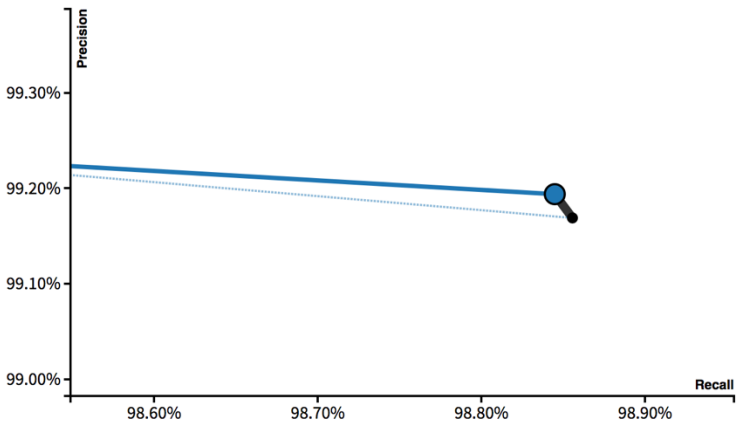


Figure 3: Example standardized HTML report output from hap.py. (a) Tier 1 high-level metrics output in the default view. (b) Tier 2 more detailed metrics and stratifications by variant type and genome context. (c) Precision-recall curve using QUAL field, where the black point is all indels, the blue point is only PASS indels, the dotted blue line is the precision-recall curve for all indels, and the solid blue line is the precision-recall curve for PASS indels.

Benchmark callsets

Benchmarking of variant calls requires a specific genome and an associated set of calls that represent the "right answers" for that genome. Such call sets have the property that they can be used as "truth" to accurately identify false positives and negatives. That is, when comparing calls from any sequencing method to this set of calls, >50% of the putative false positives and false negatives should be errors in the method being assessed. Because it is treated as the truth, this benchmark set will be referred to in this manuscript as the "truth" set, but other terms used for this include the "gold-standard" set, the "high-confidence" set, the "reference callset," or "benchmarking data."

Genome in a Bottle

The Genome in a Bottle Consortium (GIAB) is a public-private-academic consortium hosted by the National Institute of Standards and Technology (NIST) to perform authoritative characterization of a small number of human genomes to be used as benchmarks. GIAB published a benchmark set of small variant and reference calls for its pilot genome, NA12878, which characterized a high-confidence genotype for approximately 78% of the bases with sequence information (i.e., bases that are not an "N") in the human genome reference sequence (version GRCh37).[3] Since this publication, GIAB has further developed integration methods to be more reproducible, comprehensive, and accurate, and has incorporated new technologies and analysis methods. The new integration process has been used to form benchmark small variant and reference calls for approximately 90% of GRCh37 and GRCh38 for NA12878, as well as a mother-father-son trio of Ashkenazi Jewish ancestry and the son in a trio of Chinese ancestry from the Personal Genome Project (v3.3.2 at ftp://ftp-trace.ncbi.nlm.nih.gov/giab/ftp/release/).[4,6,23] The five GIAB-characterized genomes are available as NIST Reference Materials (RMs 8391, 8392, 8393, and 8398), which are extracted DNA from a single, homogenized, large growth of cells for each genome. These samples are also all available as cell lines and DNA from the Coriell Institute for Medical Research. The Personal Genome Project samples are also consented for commercial redistribution,[23] and several derived products are commercially available, including FFPE-preserved and *in vitro* mutated cell lines, or with DNA spike-ins with particular variants of clinical interest. GIAB is continuing to improve the characterization of these genomes to characterize increasingly difficult variants and regions with high-confidence.

Platinum Genomes

In addition to the benchmarking data produced by the GIAB consortium, Illumina Platinum Genomes (PG) has also created a benchmarking data set for small variants (SNVs and Indels) using the 17-member pedigree (1463) from Coriell Cell Repositories that includes the GIAB pilot sample NA12878/HG001.[5] Every sample of this pedigree was sequenced to ~50x depth on an Illumina HiSeq2000 system. Variant calls were made from this data using different combinations of aligners and variant callers. This pedigree includes 11 children of the parents (NA12877 and NA12878), producing a fully phased dataset that allows to validate the accuracy of variant calls

through genetic inheritance patterns. The HiSeq2000 sequence data used to create these benchmarking calls can be obtained from the Database of Genotypes and Phenotypes (dbGaP; https://www.ncbi.nlm.nih.gov/gap) under accession number phs001224.v1.p1. Additionally, the sequence data for six of the members of this pedigree are released through the European Nucleotide Archive (ENA; http://www.ebi.ac.uk/ena) under accession number ERP001960. The DNA and cell lines for all samples are available from the Coriell Institute for Medical Research, and DNA from a single, homogeneous batch of NA12878 is also available as NIST Reference Material 8398.

Merged PG and GiaB

Since the two resources mentioned above constitute two different methods for generating "truth" call sets for NA12878, we have merged these into a single and more comprehensive dataset. Such a "hybrid" truth set can leverage the strengths of each input, namely the diversity of technologies used as input to Genome in a Bottle and the robust validation-by-inheritance methodology employed by Platinum Genomes.

As a first pass, we have compared the call sets in NA12878 and identified the intersection as well as the ones unique to each (Supplementary Figure 2). Next, starting from the union, we have used a modified version of the k-mer validation algorithm described in [PG] to validate the merged calls (Supplementary Methods). This hybrid benchmark call set includes more total variants than either input set (67-333k additional SNVs and 85-90k additional indels), allowing us to assess more of the calls made by any sequencing pipeline without a loss in precision (see below).

This new benchmarking set represents the first step towards a more comprehensive call set that includes both "easy" to characterise variants and those that occur in difficult parts of the genome. Despite this significant advance, there remain areas for continued improvement, such as adjudication between conflicting calls and the merging of confident regions. We will continue to develop this integration method in order to further expand the breadth of coverage of this hybrid truth set resource.

Currently, neither PG nor GIAB makes high-confidence calls on chromosome Y or the mitochondrial genome. In addition, GIAB currently has chromosome X calls only for females, but PG has haploid chromosome X calls for the male NA12877 as well. Hap.py has an optional preprocessing step to guess male/female from the truth VCF. For male samples it converts haploid 1 GT calls on chrX/Y to 1/1 so that they get compared correctly by xcmp. For vcfeval, haploid 1 GT calls are treated as the same as 1/1, so this conversion is not necessary.

Synthetic Diploid

A new "synthetic-diploid" benchmark callset was created from long read assemblies of the CHM1 and CHM13 haploid cell lines, in order to benchmark small variant calls in regions difficult to analyze with short reads or in diploid genomes, which are currently excluded from the GIAB and Platinum Genomes high-confidence regions.[7] Because it is based on long reads,

performance metrics are likely less biased toward any short read sequencing technology or informatics method, and it enables benchmarking in regions difficult to map with short reads. However, because it currently contains some errors that were not corrected in the long reads, it requires a less stringent benchmarking methodology similar to the "local match" method described below. It also excludes 1bp indels from performance assessment since long read assemblies contain 1bp indel errors, and >50bp indels because these are not analyzed. Therefore, it is currently not as useful for assessing accuracy of genotypes or accuracy of the exact sequence change predicted in the REF and ALT fields. When using GA4GH tools requiring genotypes to match, the majority of FPs and FNs may not be errors in the query callset, though work is underway to improve this. Nevertheless, it is likely to be complementary to the GA4GH benchmarking strategy by enabling users to assess accuracy in more difficult regions that GIAB and Platinum Genomes currently exclude from their high confidence regions. In particular, because the truth set was not developed from short reads, and errors in the truth may be different from errors in short reads, it may better assess of relative performance between short read-based methods, particularly in more difficult genomic regions. A current limitation is that CHM1 and CHM13 cell lines are not available in a public repository.

Example comparisons

Lessons from PrecisionFDA Challenges

The PrecisionFDA team held two challenges in 2016, with participants publicly submitting results from various mapping/variant calling pipelines. While both challenges asked participants to analyze short read WGS datasets, the first challenge used a sample with high-confidence calls already available (HG001/NA12878) and the second one without high-confidence calls yet available (HG002 from GIAB, made available by GIAB upon the close of the challenge).

In the first, "Consistency" Challenge, 30x Illumina WGS of the HG001/NA12878 sample was provided from two different sequencing sites, and the VCF file results from 17 participants were assessed for reproducibility and accuracy against the GIAB v2.19 Benchmark VCF. It is possible to generate reproducible results without much variability but substantial differences from the truth. Additionally, the pipelines that generated the variant calls could be tuned to HG001, which, in many situations, was used to train or optimize pipelines.

Therefore, in the second, "Truth" Challenge, participants were asked to use their pipelines with 50x Illumina WGS to predict variants from at the time yet unknown reference sample HG0002/NA24385. Challenge results were compared using two benchmarking comparator tools, RTG Tools vcfeval for Consistency Challenge, and Vcfeval + Hap.py Comparison for Truth Challenge (more information at https://precision.fda.gov/challenges/). There were 35 entries in the Truth Challenge and the responses were submitted and ranked according to precision and recall for SNVs and indels vs. the GIAB v3.3.2 high-confidence calls for each genome (ftp://ftp-trace.ncbi.nlm.nih.gov/giab/ftp/release/). This was the first time Vcfeval + Hap.py GA4GH comparison methodology was applied at scale across the large number of entries submitted by

pipeline developers. It helped highlight the utility of the tools and the need for further development and careful interpretation of results. Based in part on feedback from the challenges, an improved benchmarking app "GA4GH Benchmarking" uploaded by user peter.krusche is now available on precisionFDA.

The results of comparisons can be informative and useful to the community, but rankings and performance metrics should be interpreted with care, as explained in the challenge results descriptions. For instance, in the Consistency Challenge the indel calling results were not calculated separately, so the indel performance was unclear considering the majority of the pipelines correctly called most of the "easy" SNVs in known high confidence regions. This became distinguishable in the Truth Challenge where separate statistics were computed for SNVs and indels, with the indels showing much larger variability when using different pipelines.

Note that both the "truth" sets and the comparison methodology in the truth challenge were newly introduced, with GA4GH comparison methodology under active development. The challenge results available on precisionFDA web page should be considered only initial evaluation, with the rich data set resulting from the challenge inviting further exploration. It is especially critical to recognize that performance metrics indicate performance for the "easier" variants and regions of the genome, so that precision and recall estimates are higher than if more difficult variants and regions were included. It is likely that some methods will perform worse than other methods for easier variants while performing better for harder variants (*e.g.*, methods using a graph reference or de novo assembly may do better in regions not assessed like the MHC or large insertions while not performing as well for easier variants because they are less mature). It is also important to manually curate a subset of FPs and FNs to ensure they are actually FPs and FNs and to understand their cause. Interestingly, stringency of matching can also significantly influence performance metrics. For example, Figure 4 shows how the number of FP indels for the assembly-based fermikit submission is much higher than the RTG submission when counting genotype errors as FPs, but the number of FPs is lower for fermikit when matching only the allele or performing distance-based matching. Additional information about relative strengths and weaknesses of the pipelines could also be gained through stratification, as discussed in the next section.

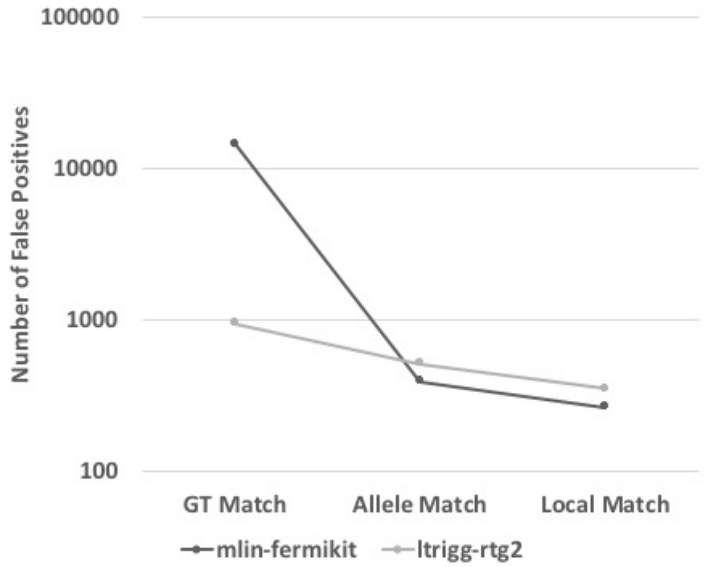


Figure 4: Matching stringency can affect relative performance of algorithms. Number of false positives for two PrecisionFDA Challenge submissions is shown for different matching stringencies, showing that the fermikit submission has many more false positives if genotype errors are counted as FPs, but that it has fewer FPs if matching only the allele or performing distance-based matching. Note that this is intended to illustrate the importance of matching stringency and is likely not indicative of the performance of these methods with optimized parameters or current versions.

Stratification illuminates challenging regions sequenced with and without PCR amplification

As an example of using stratification, we compare recall and precision for whole genome sequencing assays with and without a PCR amplification step. Table 2 shows that indel recall and precision are lower when using PCR amplification than when using PCR-free sequencing. Stratification highlights that this difference almost entirely results from PCR-related errors in homopolymers and tandem repeats, since performance is similar when excluding variants that occur within 5bp of homopolymer sequences longer than 5bp and tandem repeats longer than 10bp. Performance in regions with low GC content is similar, but PCR results in lower SNV and indel recall where GC content is > 85%.

Further stratification by type of repeat can illuminate particularly challenging genome contexts. For example, when sorting strata by recall, indels in 51-200 bp AT dinucleotide tandem repeats have substantially lower recall and precision than all other strata for both PCR and PCR-free results. Also, 86 out of 114 truth indels in 51-200 bp AT dinucleotide tandem repeats are compound heterozygous, and 89% fall outside the high-confidence regions, so our stratification and benchmarking methods help illuminate that these appear to be highly polymorphic and difficult variants to characterize.

Table 2: Recall and Precision stratified by genomic context and variant type for Illumina whole genome sequence assays with and without a PCR step

Genomic context	Type	Recall (PCR-free)	Recall (with-PCR)	Recall (PCR effect)	Precision (PCR-free)	Precision (with-PCR)	Precision (PCR effect)
All	SNV	98.4	98.4	0	99.9	99.9	0
	indel	97.1	85.8	-11.3	99.3	97.6	-1.7
Not in homo-polymers or TRs	SNV	98.5	98.6	0.1	99.9	99.9	0
	indel	98.4	98.3	-0.1	99.4	99.3	-0.1
In homo-polymers or TRs	SNV	97.2	95.6	-1.6	99.9	99.7	-0.2
	indel	96.4	78.2	-18.2	99.3	96.3	-3
GC content > 85%	SNV	94.4	84.7	-9.7	100	100	0
	indel	97.3	73.2	-24.1	97.3	96.5	-0.8
51-200 bp AT dinucleotide TRs	indel	28.0	12.0	-16	64.0	39.0	-25
All 51-200 bp dinucleotide TRs	indel	81.0	45.0	-36	94.0	84.0	-10

Benchmarking Best Practices

Box 1: GA4GH recommendations for best practices for germline variant call benchmarking

Benchmark sets	Use benchmark sets with both high-confidence variant calls as well as high-confidence regions (e.g., from GIAB or Platinum Genomes).
Stringency of variant comparison	Determine whether it is important that the genotypes match exactly, only the alleles match, or the call just needs to be near the true variant. For example, if you confirm and/or manually curate all variants to ensure you have the correct allele and genotype, then local matching may be sufficient. While the default TP, FP and FN require genotype and allele matching, the additional metrics FP.GT and FP.AL output by the GA4GH tools enable users to calculate performance at different stringencies.
Variant comparison tools	Use sophisticated variant comparison engines such as vcfeval, xcmp, or varmatch that are able to determine if different representations of the same variant are consistent with the benchmark call (examples in Fig. 1). Subsetting by high-confidence regions and, if desired, targeted regions, should only be done after comparison to avoid problems comparing variants with different representations.
Manual curation	Manually curate alignments, ideally from multiple data types, around at least a subset of putative false positive and false negative calls in order to ensure they are truly errors in the user's callset and to understand the cause(s) of errors. Report back to benchmark set developers any potential errors found in the benchmark set (e.g., using https://goo.gl/forms/ECbjHY7nhz0hrCR52 for GIAB or https://github.com/Illumina/PlatinumGenomes/issues/new for PG).
Interpret-ation of metrics	All performance metrics should only be interpreted with respect to the limitations of the variants and regions in the benchmark set. Performance is unknown for variant types and genome contexts not well represented in the benchmark set. Performance metrics are likely to be lower for more difficult variant types and regions that are not fully represented in the benchmark set, such as those in repetitive or difficult-to-map regions. When comparing methods, note that method A may perform better in the high-confidence regions, but method B may perform better for more difficult variants outside the high-confidence regions.
Stratifi-cation	Performance results should be stratified by variant type. Stratification by genomic region should also be considered to gain additional insights into strengths and limitations of the sequencing pipeline, as it can highlight regions that are not sufficiently represented. Stratification should only be done after comparison to avoid problems comparing variants with different representations.
Confidence Intervals	Confidence intervals for performance metrics such as precision and recall should be calculated. This is particularly critical for the smaller numbers of variants found when benchmarking targeted assays and/or less common stratified variant types and regions.
Additional bench-marking approaches	We recommend using other benchmarking approaches in addition to those discussed in this paper to understand performance of a pipeline, including: ● Confirming results found in samples over time ● Synthetic DNA spike-ins with challenging and common clinically relevant variants ● Engineering variants into cell lines ● Finding existing samples with challenging and common clinically relevant variants ● Simulation methods, such as read simulators, adding variants into real reads, and modifying the reference ● Run-specific metrics such as base quality score distributions, coverage distributions, etc. can also be useful to identify outlier runs

Conclusions

The GA4GH Benchmarking Team has developed a suite of tools to produce standardized performance metrics for benchmarking small germline variant calls. These sophisticated tools address challenges in standardizing metrics like recall and precision, comparing different representations of variant calls, and stratifying performance by variant type and genome context. We have developed a set of best practices for benchmarking variant calls to help users avoid common pitfalls and misinterpreting performance metrics.

Moving forward there will be a continual need for improvements in benchmarking of variant discovery methodologies. Technological evolution will enable laboratories to characterize increasingly difficult variants and genomic regions, which will require improved benchmarks. Simultaneously, this evolution can contribute to improved characterisation of reference materials. For example, the types of variants being analyzed will increase in scope: most current benchmarking focuses on relatively small variations, and entirely different techniques will be needed to consider structural variants. An algorithm that detects copy number variation needs to identify the two separate breakpoints that define the variant, but also provide an estimate of the actual number of copies present. There are thus at least three different metrics of interest for each copy number variant predicted, and it is not clear how these should be weighted to provide a singular ranking of methodologies. Similarly, performance metrics for large insertions might include accuracy of the size and sequence predictions, in addition to genotype accuracy. As a result, definitions of truth and performance metrics can become significantly more complex than for the smaller variants outlined here.

Assessment of somatic variants introduces challenges that are quite different from those presented in germline variant calling. Different benchmarking approaches are needed to handle somatic issues like assessing the accuracy of variant allele frequency or trinucleotide mutational signatures. A good germline variant caller will not perform well for somatic detection, and *vice versa*. A global consortium around benchmarking of somatic variant detection has been established, called the ICGC-TCGA DREAM Somatic Mutation Calling (SMC) group, and has been benchmarking both detection of individual variants and of broader processes like subclonal variation.[24]

Moving forward, groups will also need to modify benchmarking strategies to address changes in the way the human genome itself is represented. Today the most common way of representing the human genome involves a set of linear chromosomes (*e.g.,* the most common usage of GRCh37). There are key advantages to non-linear representations of the genome, including ability to recognize copy-number and other polymorphisms directly in the reference, and as a result more graphical structures are in development.[25] The GRCh38 build of the human genome makes a key step towards this with its use of ALT loci, which provide multiple distinct versions of specific regions of the genome.[26] These ALT loci are not well-accounted for by most aligners or the benchmarking tools we describe, and their impact on benchmarking studies is largely unexplored and likely would require a variety of samples with differing ALT alleles. It is likely that

the core representation of the genome will continue to evolve over time, and benchmarking tools will continue to evolve.

Acknowledgments

We thank GA4GH, especially Stephen Keenan, David Lloyd, and Rishi Nag, for their support in hosting and organizing the Benchmarking Team. We thank the many contributors to Benchmarking Team discussions over the past few years. Certain commercial equipment, instruments, or materials are identified to specify adequately experimental conditions or reported results. Such identification does not imply recommendation or endorsement by the National Institute of Standards and Technology or the Food and Drug Administration, nor does it imply that the equipment, instruments, or materials identified are necessarily the best available for the purpose.

References

1. Yang, Y. *et al.* Molecular Findings Among Patients Referred for Clinical Whole-Exome Sequencing. *JAMA* **312,** 1870 (2014).
2. Xue, Y., Ankala, A., Wilcox, W. R. & Hegde, M. R. Solving the molecular diagnostic testing conundrum for Mendelian disorders in the era of next-generation sequencing: single-gene, gene panel, or exome/genome sequencing. *Genet. Med.* **17,** 444–451 (2015).
3. Zook, J. M. *et al.* Integrating human sequence data sets provides a resource of benchmark SNP and indel genotype calls. *Nat. Biotechnol.* **32,** 246–51 (2014).
4. Zook, J. M. *et al.* Extensive sequencing of seven human genomes to characterize benchmark reference materials. *Sci. Data* **3,** 160025 (2016).
5. Eberle, M. A. *et al.* A reference data set of 5.4 million phased human variants validated by genetic inheritance from sequencing a three-generation 17-member pedigree. *Genome Res.* (2016). doi:10.1101/gr.210500.116
6. Zook, J. *et al.* Reproducible integration of multiple sequencing datasets to form high-confidence SNP, indel, and reference calls for five human genome reference materials. *bioRxiv* 281006 (2018). doi:10.1101/281006
7. Li, H. *et al.* New synthetic-diploid benchmark for accurate variant calling evaluation. *bioRxiv* 223297 (2017). doi:10.1101/223297
8. Highnam, G. *et al.* An analytical framework for optimizing variant discovery from personal genomes. *Nat. Commun.* **6,** 6275 (2015).
9. Cleary, J. G. *et al. Comparing Variant Call Files for Performance Benchmarking of Next-Generation Sequencing Variant Calling Pipelines*. *bioRxiv* (Cold Spring Harbor Labs Journals, 2015). doi:10.1101/023754
10. Sun, C. & Medvedev, P. VarMatch: robust matching of small variant datasets using flexible scoring schemes. *Bioinformatics* **33,** btw797 (2016).
11. Talwalkar, A. *et al.* SMaSH: A benchmarking toolkit for human genome variant calling. *Bioinformatics* **30,** 2787–2795 (2014).
12. The Variant Call Format Specification. (2017). at <https://samtools.github.io/hts-specs/VCFv4.3.pdf>
13. Roper, W. L. *et al.* Good Laboratory Practices for Molecular Genetic Testing for Heritable

Diseases and Conditions. *MMWR* **58,** (2009).
14. Mattocks, C. J. *et al.* A standardized framework for the validation and verification of clinical molecular genetic tests. *Eur. J. Hum. Genet.* **18,** 1276–1288 (2010).
15. Gargis, A. S. *et al.* Assuring the quality of next-generation sequencing in clinical laboratory practice. *Nat. Biotechnol.* **30,** 1033–1036 (2012).
16. Rehm, H. L. *et al.* ACMG clinical laboratory standards for next-generation sequencing. *Genet. Med.* **15,** 733–747 (2013).
17. Aziz, N. *et al.* College of American Pathologists' Laboratory Standards for Next-Generation Sequencing Clinical Tests. *Arch. Pathol. Lab. Med.* **139,** 481–493 (2015).
18. Roy, S. *et al.* Standards and Guidelines for Validating Next-Generation Sequencing Bioinformatics Pipelines: A Joint Recommendation of the Association for Molecular Pathology and the College of American Pathologists. *J. Mol. Diagn.* **20,** 4–27 (2018).
19. Hasan, M. S., Wu, X., Watson, L. T., Li, Z. & Zhang, L. UPS-indel: A Universal Positioning System For Indels. *bioRxiv* (2017). at <http://biorxiv.org/content/early/2017/05/03/133553>
20. Tan, A., Abecasis, G. R. & Kang, H. M. Unified representation of genetic variants. *Bioinformatics* **31,** 2202–2204 (2015).
21. Krusche, P. Haplotype comparison tools / hap.py. at <http://github.com/illumina/hap.py>
22. Jacobs, K. B. Variant Graph Comparison Tool (vgraph). at <https://github.com/bioinformed/vgraph>
23. Ball, M. P. *et al.* A public resource facilitating clinical use of genomes. *Proc. Natl. Acad. Sci. U. S. A.* **109,** 11920–7 (2012).
24. Ewing, A. D. *et al.* Combining tumor genome simulation with crowdsourcing to benchmark somatic single-nucleotide-variant detection. *Nat. Methods* **12,** 623–30 (2015).
25. Novak, A. M. *et al.* Genome Graphs. *bioRxiv* 101378 (2017). doi:10.1101/101378
26. Schneider, V. A. *et al.* Evaluation of GRCh38 and de novo haploid genome assemblies demonstrates the enduring quality of the reference assembly. *Genome Res.* **27,** 849–864 (2017).

Supplementary Information

(a)

VCF Records

CHROM	POS	REF	ALT	GT
chr1	247695097	A	C	1/1
chr1	247695098	T	.	0/0
chr1	247695099	G	A	1/1

REF : ATG in ORC2C3|NM_198074
SNP 1: CTG (missense)
SNP 2: ATA (missense)
MNP : CTA (stop codon)

(b)

VCF Records

CHROM	POS	REF	ALT	GT
chr17	7386279	T	C	1/1
chr17	7386280	G	A	1/1

REF : TGG in SLC35G6|NM_001102614.1
SNP 1: CGG (missense)
SNP 2: TAG (stop codon)
MNP : CAG (missense)

Supplementary Figure 1: Two examples in NA12878 where local phasing of variants can affect the interpretation. (a) In this case, if the SNPs are interpreted independently then they are two missense mutations, and if they are interpreted together then a stop codon has been gained. (b) In this case, if the SNPs are interpreted independently then there is one missense mutation and one gained stop codon, and if they are interpreted together then it is just a missense mutation. If these events were heterozygous without phasing information, then the interpretation would be ambiguous from the VCF.

Supplementary Table 1: Definitions and formulas for performance metrics output by GA4GH tools. Bold metrics are "Tier 1" metrics displayed by default, and others are "Tier 2" metrics output but optionally displayed.

Metric	Common Name(s)	Definition	Formula
TRUTH.TP	**True positives (Truth)**	**Number of truth calls for which there is a query call that is consistent with the truth call and its genotype**	
QUERY.TP	**True positives (Query)**	**Number of query calls for which there is a truth call that is consistent with the query call and its genotype. This can differ from TRUTH.TP if complex changes are represented as a single change in TRUTH.TP and as multiple primitive SNVs and indels in QUERY.TP, or vice versa.**	
TRUTH.FN	**False negatives**	**Number of truth calls for which there is no query call that is consistent with the truth call and its genotype**	
QUERY.FP	**False positives**	**Number of query calls for which there is no truth call that is consistent with the query call and its genotype.**	
QUERY.UNK	Number of unknown variant calls	The number of query variant calls not inside confident regions of the truth dataset	
QUERY.TOTAL		**Total number of query calls**	**QUERY.TP + QUERY.FP + QUERY.UNK**
TRUTH.TOTAL		Total number of truth calls	TRUTH.TP + TRUTH.FN
METRIC.Recall	**Recall, Sensitivity**	**Fraction of truth calls that are consistent with a query allele and genotype call within the confident regions**	**TRUTH.TP / (TRUTH.TP + TRUTH.FN)**
METRIC.Precision	**Precision, Positive predictive value**	**Fraction of query calls that are consistent with a truth allele and genotype call within the confident regions**	**QUERY.TP / (QUERY.TP + QUERY.FP)**
METRIC.Frac_NA	**Fraction not assessed**	**Fraction of query calls that are outside the confident regions of the truthset (and which could not be assessed in this benchmarking run)**	**QUERY.UNK / QUERY.TOTAL**
F-Score	**F1 Score**	**The harmonic mean between recall and precision**	**2 * METRIC.Recall* METRIC.Precision / (METRIC.Recall + METRIC.Precision)**

FP.GT	Genotype errors	This is the number of query variants with an incorrect genotype, but the correct allele (e.g., when the query GT is 1/1 and truth GT is 0/1)	
FP.AL	Allele errors	The number of query variant calls which could not be matched by genotype or by alleles, but which have a truth variant call within a specified distance	

A Benchmarking VCF file formats

Intermediate VCF Files

The intermediate VCF file is produced by a comparison engine (like xcmp / vcfeval / xcmp in hap.py).

A comparison engine should

- assess for each call in the truth and in the query whether this call is considered a match or mismatch, and ideally output if the allele matches but the genotype does not (FP.GT) or if the wrong allele is called near a true allele (FP.AL)
- output corresponding labels:
    - True Positive / `TP`: present in both truth and query
    - False Positive / `FP`: present only in the query
    - False Negative / `FN`: present only in the truth
    - Not-assessed / `N`: call was not assigned a match status
- (optionally) output additional information about each decision

Note that the comparison engine should output the FP/TP/FN/N labels separately for truth and for query variants. For simple comparison types which do not attempt to reconcile different variant representations, the assigned type for truth and query might be the same. However, more sophisticated methods (e.g. vcfeval) will be able to type truth and query variants separately.

The N label may be applied for a variety of reasons, which may be specific to the comparison engine. For example, a comparison engine might not assess input calls which had non-PASS FILTER fields, or may choose to ignore half-calls. Alternatively an engine may find that some call regions are too complex to confidently assess. Additional information may be included in the BI annotation.

A comparison engine should also preserve input INFO / FORMAT annotations to the largest degree possible (depending on the variant processing it does).

Required VCF Fields

The intermediate VCF must have two columns named TRUTH and QUERY, with these FORMAT annotations:

```
##FORMAT=<ID=BK,Number=1,Type=String,Description="Sub-type for decision (match/mismatch type)">
```

The value of BK specifies the class of match for each variant record:

1. `.`: *missing* = no match at any level tested by the comparison tool
2. `lm`: *local match* = the truth/query variant is nearby a variant in the query/truth -- if the tool outputs such match types, it should annotate the VCF header with the definition of local matches (e.g. match within a fixed window, or within the same superlocus).
3. `am`: *almatch* = the variant forms (part of) an allele match (independent of representation, i.e. one-sided haplotype match)
4. `gm`: *gtmatch* = diploid haplotypes (and genotypes) were resolved to be the same (independent of representation)

Based on the values in BK, comparison tools must assign a decision for each variant call that assigns true/false positive/negative status. This status is output in the BD format field:

```
##FORMAT=<ID=BD,Number=1,Type=String,Description="Decision for call (TP/FP/FN/N)">
```

The mapping of combinations of BK values to BD could vary for different comparison stringencies, as discussed in the Variant Counting section. Our current tools require genotypes to match for TP to be in the BD field.

Local Matching

Find all variants in the query where another variant was seen nearby in the truth.

BK	BD (Truth)	BD (Query)
.	FN	FP
lm/am/gm	TP	TP

Allele Matching

Test allele-level concordance between truth and query.

BK	BD (Truth)	BD (Query)
./lm	FN	FP
am/gm	TP	TP

Genotype Matching

Test genotype concordance between truth and query.

BK	BD (Truth)	BD (Query)
./lm	FN	FP
am	FN	FP (and FP.GT)
gm	TP	TP

Additional VCF Fields

Comparison engines may have engine-specific status information that is useful to present in the output (for example, error status, specific match sub-algorithm used, etc). This can be recorded in the optional BI annotation:

```
##FORMAT=<ID=BI,Number=1,Type=String,Description="Additional match status information">
```

We allow for comparison engines to transform the input variants for more granular accounting / comparison. To facilitate ROC creation based on such a processed variant file, we define the following FORMAT annotation to pass on a variant quality score obtained from the input variants:

```
##FORMAT=<ID=QQ,Number=1,Type=Float,Description="Variant quality for ROC creation.">
```

Since we aim to benchmark variant calls independently of their representation, we also define *superloci*. A *superlocus* is a set of variant calls that intends to fully describe the variation within a contiguous reference region that may contain complex variation with different representations. In order to group variants into blocks by superlocus, we introduce the following INFO annotation that allows us to assign a benchmarking superlocus ID to each variant record.

```
##INFO=<ID=BS,Number=1,Type=Integer,Description="Benchmarking superlocus ID for these variants.">
```

In downstream tools, this may e.g. be used to count with superlocus granularity.

Final Output VCF Files

The output VCF file is similar to the intermediate VCF file. We add one additional variant classification: in addition to `FP`/`FN`/`TP`, each variant call can also be assigned the status `UNK` for unknown / outside the regions which the truthset covers.

A simple definition for `UNK` variants is as follows: We call any variant unknown if it `BD` == `FP` and `BK` == `miss` and if the variant is outside the confident call regions of the truth set. If the truthset does not give confident call regions, no `UNK` variants are output.

We introduce the following additional fields:

```
##INFO=<ID=Regions,Number=.,Type=String,Description="Tags for confident / stratification regions.">
##FORMAT=<ID=BVT,Number=1,Type=String,Description="High-level variant type in truth/query (SNV|INDEL|COMPLEX).">
##FORMAT=<ID=BLT,Number=1,Type=String,Description="High-level location type in truth/query (het|hom|hetalt|halfcall|multiallelic|nocall).">
```

Optional: variant types seen and counted for this record.

```
##INFO=<ID=VTC,Number=.,Type=String,Description="Variant types used for counting.">
```

B Variant types

Counts and statistics are calculated for the following subsets of variants:

Type	Description
SNV	SNV or MNP variants. We count single nucleotides that have changed
INDEL	Indels and complex variants

Hap.py (and qfy.py, which is the part of the hap.py that counts variants) also computes counts for subtypes and observed genotypes in the above two categories.

Subtype	Description
*	Aggregate numbers for all subtypes
ti or tv	Transitions and transversions for SNVs
I1_5	Insertions of length 1-5
I6_15	Insertions of length 6-15
I16_PLUS	Insertions of length 16 or more
D1_5	Deletions of length 1-5
D6_15	Deletions of length 6-15
D16_PLUS	Deletions of length 16 or more
C1_5	Complex variants of length 1-5
C6_15	Complex variants of length 6-15
C16_PLUS	Complex variants of length 16 or more

Genotype	Description
*	Aggregate numbers for all genotypes
het	only heterozygous variant calls (0/1 or similar genotypes)
homalt	only homozygous alternative variant calls (1/1 or similar genotypes)
hetalt	only heterozygous alternative variant calls (1/2 or similar genotypes)

Note that currently the granularity of counting is on a per-VCF-record level. In complex/high-variability regions, the classifications above might become inaccurate due to nearby variants (e.g. an insertion with a close-by SNV would be more accurately classified as "complex" if the SNV and the insertion occur on the same haplotype).

C Benchmarking output columns

For each variant stratification subset, the GA4GH benchmarking workflow outputs a set of stratification columns, followed by metrics columns. Stratification columns may contain the placeholder "*" value, which indicates that values in this row are aggregated over all possible values of the column. In case of a subtype, this means that the counts have been summed across all subtypes. In case of QQ, it means that the counts correspond to all variants rather than only ones below a QQ threshold.

Stratification Column	**Description**
Type	Variant type (SNV / INDEL)
Subtype	Variant Subtype (ti/tv/indel length, see above)
Subset	Subset of the genome/stratification region
Filter	Variant filters: PASS, SEL, ALL, or a particular filter from the query VCF
Genotype	Genotype of benchmarked variants (het / homalt / hetalt)
QQ.Field	Which field from the original VCF was used to produce QQ values in truth and query
QQ	QQ threshold for ROC values

Metric Column	Description
METRIC.Recall	Recall for truth variant representation = TRUTH.TP / (TRUTH.TP + TRUTH.FN)
METRIC.Precision	Precision of query variants = QUERY.TP / (QUERY.TP + QUERY.FP)
METRIC.Frac_NA	Fraction of non-assessed query calls = QUERY.UNK / QUERY.TOTAL
METRIC.F1_Score	Harmonic mean of precision and recall = 2*METRIC.Recall*Metric.Precision/(METRIC.Recall + METRIC.Precision)
TRUTH.TOTAL	Total number of truth variants
TRUTH.TP	Number of true-positive calls in truth representation
TRUTH.FN	Number of false-negative calls = calls in truth without matching query call
QUERY.TOTAL	Total number of query calls
QUERY.TP	Number of true-positive calls in query representation
QUERY.FP	Number of false-positive calls in the query file (extra query calls in truth bed file)
QUERY.UNK	Number of query calls outside the confident regions
FP.gt	Number of genotype mismatches (alleles match, but different zygosity)
FP.al	Number of allele mismatches (variants matched locally but not by alleles)
TRUTH.TOTAL.TiTv_ratio	Transition / Transversion ratio for all truth variants
TRUTH.TOTAL.het_hom_ratio	Het/Hom ratio for all truth variants
TRUTH.FN.TiTv_ratio	Transition / Transversion ratio for false-negative variants
TRUTH.FN.het_hom_ratio	Het/Hom ratio for false-negative variants
TRUTH.TP.TiTv_ratio	Transition / Transversion ratio for true positive variants
TRUTH.TP.het_hom_ratio	Het/Hom ratio for true positive variants
QUERY.FP.TiTv_ratio	Transition / Transversion ratio for false positive variants
QUERY.FP.het_hom_ratio	Het/Hom ratio for false-positive variants
QUERY.TOTAL.TiTv_ratio	Transition / Transversion ratio for all query variants
QUERY.TOTAL.het_hom_ratio	Het/Hom ratio for all query variants
QUERY.TP.TiTv_ratio	Transition / Transversion ratio for true positive variants (query representation)
QUERY.TP.het_hom_ratio	Het/Hom ratio for true positive variants (query representation)
QUERY.UNK.TiTv_ratio	Transition / Transversion ratio for unknown variants
QUERY.UNK.het_hom_ratio	Het/Hom ratio for unknown variants
Subset.Size	When using stratification regions, the number of nucleotides in the region
Subset.IS_CONF.Size	This gives the number of confident bases (-f regions) in the region

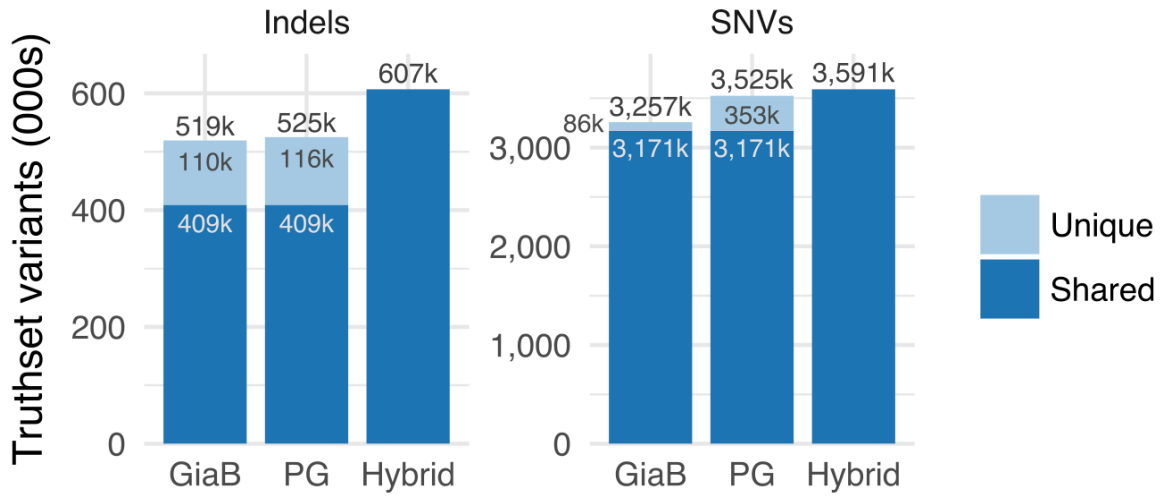
D Merging Truth Calls for NA12878

Platinum Genomes v2016.1 (PG) and NIST Genome in a Bottle v3.3.2 (GIAB) calls were combined through a k-mer validation process adapted from that used in [PG 2017]. The steps in this procedure are as follows:

1. Use hap.py to compare both NA12878 truth set VCFs and identify those records exclusive to GIAB (i.e. 'false positives' in the GIAB query)
2. Merge these GIAB-exclusive calls with the PG set to produce a combined VCF of candidate records for validation
3. For fully-phased loci:
    a. Induce k-mers containing local haplotype sequence of 51 bp centered on each record of the merged VCF
    b. Count exact k-mer matches in aligned reads in the region (+/- 400 bp) of the VCF record for each haplotype sequence
    c. For each haplotype, sum the counts over expected inherited haplotypes in the lower pedigree and divide by the number of expected inheritances to get a normalised k-mer score (KM)
    d. Take the minimum KM of both NA12878 haplotypes in the phased record and filter those where the minimum KM is less than 1
4. For loci spanning one or more unphased records:
    a. Generate all possible k-mer sequences considering any phased or unphased records in the merged VCF within the 51 bp window
    b. Count exact matches in aligned sequence reads for all k-mers, again in the aligned reads surrounding the site (+/- 400 bp). Those k-mers with >=2 counts in NA12878 are taken forward as candidate haplotypes
    c. Assign a k-mer to a PG haplotype label (relative to NA12878) based on which assignment maximises the normalised k-mer score (KM, see above) and subsequently filter any haplotype with KM < 1
    d. If a single pair of haplotypes are able to form a complementary diplotype, this is a validated and phased record. If there are more than one valid haplotype pairs (as may happen at short tandem repeats) the record cannot be validated and is discarded
5. Newly-validated calls are added to the PG confidence tracks

Steps 1 and 2 in the above procedure sidestep the problem of conflicting sites between the two call sets. An improved approach would be to consolidate and arbitrate conflicts, else filtering ambiguous sites where the true state cannot be determined.

The k-mer validation method could also be applied so as to graft PG-exclusive calls onto the GIAB truth set, however many of these PG exclusive calls will fall outside GIAB confident regions and as such would benefit from a more sophisticated method capable of integrating confident regions.



Supplementary Figure 2: Hybrid Genome in a Bottle and Platinum Genomes truthset. The hybrid truth set combines variants from Genome in a Bottle and Platinum Genomes into a single, more comprehensive gold standard. Intersection counts are shown for Genome in a Bottle (GiaB) v3.3.2 GRCh37 compared with Platinum Genomes (PG) v2016.1 as reported by hap.py v0.3.7. The union of both callsets was then re-validated using k-mer testing of inherited haplotypes in the CEPH 1463 pedigree, with all passing calls added to the hybrid truth set (Supp. Methods).

E Comparison tools

The matching methods have been implemented in different variant comparison engines which can be used as part of the GA4GH benchmarking workflow. These methods are:

- *scmp-distancebased*: this will match variants by location only. Any variant in the query that has a truth variant nearby within a specified distance.
- *scmp-somatic*: this mode is intended to be used with Tumor/Normal VCF files. Variants in the query will be matched to truth variants if the same normalized alleles are found nearby. Genotypes are ignored, and multi-sample VCF files will be collapsed into a single column with a pre-specified genotype. Alleles are split out into one allele per VCF line.
- *RTG Tools vcfeval* (see below for more detailed instructions): this is a comparison that is similar to xcmp, but which uses an optimization method to determine TP/FP status on a per-variant level (rather than xcmp's per-superlocus level). This method does not use phasing information when it is present in the input VCF file and produces genotype matches. Vcfeval can also perform allele matching when using the --squash-ploidy option, dealing appropriately with variant representation issues, which is useful for somatic callset benchmarking.
- *xcmp* (hap.py's default comparison engine): this method will assume that both input samples are diploid / human samples. Matching is performed on a haplotype level: xcmp enumerates all possible haplotypes that may be described by truth and query within a small superlocus. If matching pairs of haplotype sequences are found, xcmp will convert all variants within the surrounding superlocus into TPs. Xcmp also recognizes direct genotype or allele matches / mismatches. Global phasing information is used to restrict haplotype enumeration for phased genotype matching, but PS phasing is not supported.