TECHNICAL REPORT
https://doi.org/10.1038/s41588-018-0145-5

nature genetics

Accurate genotyping across variant classes and lengths using variant graphs

Jonas Andreas Sibbesen[1,3], Lasse Maretty[1,3], The Danish Pan-Genome Consortium[2] and Anders Krogh[1]*

Genotype estimates from short-read sequencing data are typically based on the alignment of reads to a linear reference, but reads originating from more complex variants (for example, structural variants) often align poorly, resulting in biased genotype estimates. This bias can be mitigated by first collecting a set of candidate variants across discovery methods, individuals and databases, and then realigning the reads to the variants and reference simultaneously. However, this realignment problem has proved computationally difficult. Here, we present a new method (BayesTyper) that uses exact alignment of read k-mers to a graph representation of the reference and variants to efficiently perform unbiased, probabilistic genotyping across the variation spectrum. We demonstrate that BayesTyper generally provides superior variant sensitivity and genotyping accuracy relative to existing methods when used to integrate variants across discovery approaches and individuals. Finally, we demonstrate that including a 'variation-prior' database containing already known variants significantly improves sensitivity.

Single-nucleotide variants (SNVs) and short indels can be accurately genotyped from second-generation genome sequencing data by first mapping the reads to a reference genome (using, for example, BWA-MEM[1]) followed by genotyping (using, for example, HaplotypeCaller[2]). Yet, genotyping larger variants or regions with dense variation remains difficult because we cannot reliably align reads containing such variants to the reference genome using standard approaches.

This alignment problem can be partially mitigated by allowing for larger edit distances (for example, by combining multiple local alignments of a read as used in the split-read approach (such as in BWA-MEM)), yet some reads will remain only partially aligned when one of the sequence anchors flanking the variant is too short to align uniquely. This effect is even more pronounced for insertions as the chance of having an unalignable anchor increases with the insertion length until the point at which the entire read is unalignable. Alternatively, one can first de novo assemble contigs and then call variants by realigning these back to the reference genome. However, assembly requires high coverage and is noisy, and it is computationally difficult to propagate all information about variant support contained in the raw data. Hence, even when split-read alignment or aligned de novo assemblies detect the variant in question, the genotype estimates will rely only on partial information. More importantly, due to the stochastic nature of read sampling, a variant may be detected only in some (or none) of the individuals in a study cohort, even when the raw data do in fact contain information supporting the variant. Together, these findings suggest that genotyping needs to be a two-stage procedure in which candidate variants are first discovered and combined across methods and individuals along with databases containing known variation, and then genotyped on the basis of an unbiased realignment of the reads to a representation containing both the candidate variants and the reference sequence.

Standard genotyping methods such as HaplotypeCaller and Platypus[3] do implicitly apply this two-stage approach to share information across individuals and multiple discovery approaches (for example, local assembly in Platypus) and also allow for adding databases containing known variation. Large-scale projects such as 1000 Genomes[4,5] (1000G) and Genome of the Netherlands[6,7] (GoNL) have extended this by further combining several tools in the discovery phase to increase sensitivity. However, rather than then re-interrogating the entire set of reads for information about all putative variants when performing the genotyping step, standard methods such as HaplotypeCaller, Platypus, Freebayes[8] and GenomeStrip[9] reduce the problem to merely improving the alignment of reads already anchored to a particular region. This causes loss of information, especially for larger variants or when multiple variants occur within the same read. Importantly, even expanding the realignment to all reads that did not align in the first attempt would still ignore important information as reads that did align initially may turn out to align better to a variant. Optimally, we thus need to align the entire collection of reads to the candidate variants and reference in an unbiased way.

This problem corresponds to aligning reads to a graph representation of the reference and variants in which nodes represent sequences (that is, reference or alternative allele sequences) and edges represent junctions between them. Indeed, graph alignment has recently received significant attention[10,11]. However, the methods presented so far are very computationally demanding[12–14], target only specific, complex regions[15] or are restricted to simple variation[16].

BayesTyper is a new general method that uses exact alignment of read k-mers to a graph representation of variants to efficiently obtain unbiased information about the read support for any type of variant (SNVs, large structural variants and so on). It then uses this as a basis for estimating the genotypes of a population based on a probabilistic model. Through experiments on multiple simulated and real data sets, we demonstrate that BayesTyper generally provides higher variant sensitivity and genotype accuracy across variant types than standard mapping-based methods. We further show that by enabling researchers to combine multiple variant discovery approaches with a database containing known variation, BayesTyper

[1]The Bioinformatics Centre, Department of Biology, University of Copenhagen, Copenhagen, Denmark. [2]A complete list of consortium members is provided in the Supplementary Note. [3]These authors contributed equally: Jonas Andreas Sibbesen, Lasse Maretty. *e-mail: krogh@binf.ku.dk

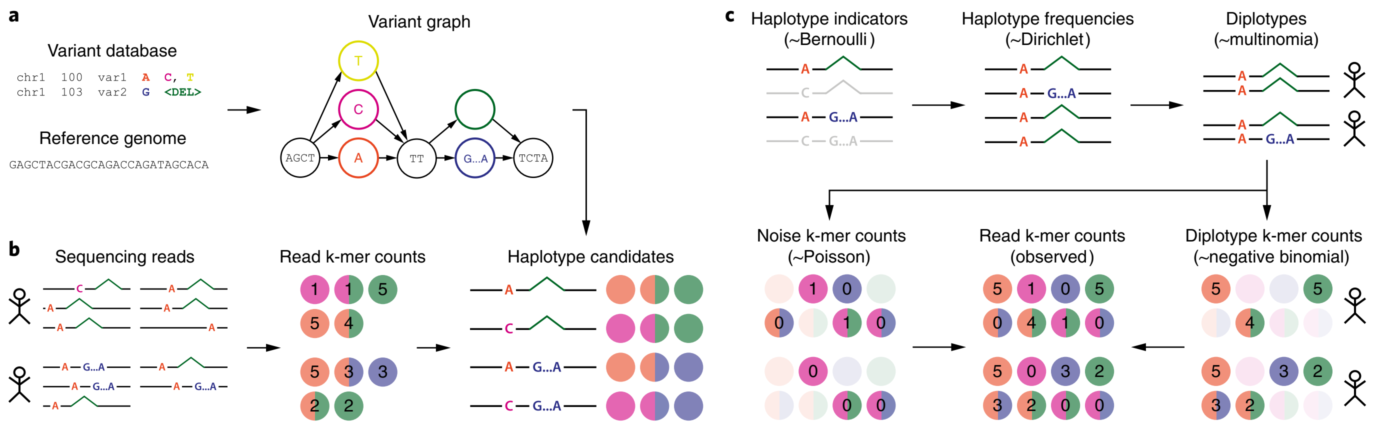
a
Variant graph
Variant database
chrl 100 var1 A C, T
chrl 103 var2 G <DEL>
Reference genome
GAGCTACGACGCAGACCAGATAGCACA
b
Sequencing reads
Read k-mer counts
Haplotype candidates
c
Haplotype indicators (~Bernoulli)
Haplotype frequencies (~Dirichlet)
Diplotypes (~multinomia)
Noise k-mer counts (~Poisson)
Read k-mer counts (observed)
Diplotype k-mer counts (~negative binomial)

Fig. 1 | BayesTyper. a, A variant graph is constructed from the set of input variants and the reference genome, where nodes represent sequences (reference or variant allele sequences) and edges represent possible genomic connections between the node sequences. **b**, All k-mers in the sequencing reads are tabulated for each individual using KMC3 (circles represent k-mers; only a single k-mer from each allele combination is shown). Haplotype candidates and their corresponding k-mer profiles are then generated by traversing the variant graph using an *n*-best algorithm, where the score is based on the individuals' k-mer counts. **c**, Genotype inference is based on a generative model of the sequencing process. First, frequencies for all haplotype candidates are sampled from a sparse-prior distribution. A diplotype (that is, haplotype pair) is then drawn independently for each individual conditioned on the population frequencies. Finally, conditioned on the diplotype, each individual's set of observed k-mer counts is modeled as generated by combining counts from the individual's haplotypes with counts from a noise process.

significantly improves the sensitivity for structural variation, especially on low-coverage data.

Results

We first present an overview of BayesTyper; a detailed description is provided in the Methods.

BayesTyper. The objective is to query the sequencing reads for support for a set of candidate variants in a way that is unbiased with respect to the reference sequence and then use this information as a basis for genotyping. The overall idea is to compare k-mers of a certain preset size in the reads from an individual with k-mers from variants in a database of variant candidates. The posterior distribution over possible genotypes is then estimated from the counts of variant k-mers in the reads based on a generative model.

To generate a database of variant k-mers, we first observe that variants that are less than $k-1$ nucleotides apart will share a k-mer and we thus need to co-estimate the genotypes at these positions. We therefore cluster variants that are less than $k-1$ nucleotides apart and construct a variant graph for each such cluster. Co-estimating the genotypes of all variants in such a cluster corresponds to estimating the pair of local haplotypes (that is, the 'diplotype'), from which the genotypes of the constituent variants can be obtained. Optimally, we would simply enumerate all possible haplotypes (that is, all possible paths in the cluster graph), collect the k-mer counts for these haplotypes in the sequencing reads and use this to obtain a genotype estimate. However, the number of haplotypes grows exponentially with the number of variants, and we therefore devised an iterative, heuristic algorithm that uses read k-mer information to enumerate only the most likely subset of the possible haplotypes.

First, variants are clustered using a k-mer size of 55 and graphs are constructed for each cluster (Fig. 1a). Next, the occurrences of k-mers in the sequencing reads are counted for each individual using the KMC3 program[17] (Fig. 1b). Haplotypes are then enumerated using a heuristic algorithm that uses a Bloom filter representation of the read k-mers from each individual to select the most likely haplotypes. Provided with a set of candidate haplotypes, we then enumerate the set of k-mers ($k=55$) found in at least one haplotype and generate a table containing the read k-mer count for each individual for each of the haplotype k-mers.

We model the vector of these observed k-mer counts for each individual as generated by combining counts obtained from an individual's diplotype, modeled using the negative binomial distribution, with counts originating from a noise process (Fig. 1c). An individual's diplotype is in turn modeled as drawn from a shared population of local haplotypes whose frequencies are modeled using a sparse prior; this construct allows for sharing of genotyping information across individuals. Under this model, the posterior distribution over genotypes is then inferred using collapsed Gibbs sampling of diplotypes, haplotype frequencies and noise parameters. The posterior distribution over genotypes is subsequently obtained directly from the diplotype posterior and the maximum a posteriori genotype provided as the genotype call together with the posterior probability estimates for all possible genotypes.

Comparison of genotyping methods. The input to BayesTyper is a set of reads, a reference sequence and a set of candidate variants, which can include those called from the reads by various methods in the initial discovery phase, as well as variants already discovered in other projects (dbSNP[18], 1000G[4,5] and so on). Hence, BayesTyper cannot genotype variants not provided as input. We performed a number of computational experiments to test the performance of BayesTyper relative to standard mapping-based, genotyping methods as well as to determine the impact of using different strategies to collect candidate variants. The key variables were the type of sequencing data set used and the completeness, and thus complexity, of the set of candidate variants used as a basis for genotyping.

We first assessed the performance of BayesTyper relative to four standard mapping-based methods: HaplotypeCaller[2], Platypus[3], FreeBayes[8] and SVTyper[19]. The first three methods cover both simple variation (SNPs and short indels) and, to a varying extent, more complex forms of variation, and they can perform both variant discovery and genotyping. In contrast, SVTyper is restricted to larger structural variants (>50 nucleotides (nt)) and is a dedicated genotyping tool (like BayesTyper) and thus cannot perform any variant discovery on its own. These methods were selected among the vast number of available callers as they are widely used and can be run in a 'genotyping-only' mode, where only variants provided as input are considered. All methods were tested on real data using 13 individuals (2 parents and 11 children) from the Platinum Genomes[20]

Table 1 | Variant candidate sets

Data set	Discovery sources	Prior[a] included	Number of variant alleles
Simulation, 10×	FB, HC, PP and MT	No	14,631,607
Simulation, 30×	FB, HC, PP and MT	No	13,433,852
PG	FB, HC, PP and MT	No	11,742,859
PG	FB, HC, PP and MT	Yes	61,103,311
GoNL	GoNL SNPs release 5 and GoNL SV release 6	No	21,404,232
GoNL	GoNL SNPs release 5 and GoNL SV release 6	Yes	64,410,663

FB, FreeBayes; HC, HaplotypeCaller; PP, Platypus; MT, Manta. [a]Variation prior constructed by combining different databases (Supplementary Table 1).

(PG, 50× coverage) study and 10 parent–offspring trios from the GoNL study[6,7] (13× coverage). For the PG data, genotyping accuracy was estimated both using pedigree haplotype transmission information and an SNV/short indel 'ground-truth' set from the Genome in a Bottle consortium (GIAB)[21,22]. For the GoNL data, a strict trio concordance metric was used, where a variant was labeled as correctly genotyped if no Mendelian violations occurred across all genotyped trios, excluding trios with filtered genotypes. All methods were also tested on simulated data (30× and 10× coverage) based on the 1000G project calls for 10 Yoruba individuals from Ibadan, Nigeria (YRI).

We generally tested the different methods on a set of input candidate variants constructed by merging the variants discovered using a panel of methods (HaplotypeCaller, Platypus, FreeBayes and Manta[23]) across individuals to maximize sensitivity across variant classes and lengths by spanning different discovery approaches (for example, de novo assembly by Manta) (Table 1). However, for GoNL, we instead used a combination of their SNV[6] and recent structural variant[7] calls for the entire GoNL cohort ($n=769$) as input, where the latter were similarly obtained using multiple discovery methods. All methods received the same candidate variants, although longer variants (>100 nt) had to be excluded from HaplotypeCaller and FreeBayes and very large variants (>10,000 nt) from Platypus as they were unable to process these. For SVTyper, only deletions, duplications and inversions longer than 50 nt were included in the input set as the method was primarily built for these types of variants. Finally, variants longer than 500,000 nt were excluded by BayesTyper to limit computation time and memory usage. BayesTyper used k-mer count tables for all individuals as input and was run with default parameters, whereas the other methods were provided with BWA-MEM read alignments post-processed according to the different methods' documentation and run in re-genotyping mode.

On the PG data set, BayesTyper was the only method to attain both high genotyping accuracy and the maximum number of called variant alleles (used as a proxy for variant sensitivity as the number of true variants is unknown) observed across methods for SNVs (Fig. 2a). For short- and medium-sized insertions and deletions (≤500 nt), BayesTyper generally exhibited both higher sensitivity and genotyping accuracy than any other method, except for medium-sized deletions (>50 nt, ≤500 nt) for which SVTyper had higher sensitivity at the cost of lower accuracy. The picture was less clear for the relatively few large variants (>500 nt) and inversions, where the tested methods balanced sensitivity and accuracy differently. BayesTyper performed uniformly better on complex variants (that is, variants that do not fall into any of the other classes).

When viewing short- and medium-sized unbalanced variants (that is, insertions, deletions and complex variants) at higher resolution (Fig. 2b), BayesTyper exhibited high and stable absolute genotyping accuracy across lengths. In contrast, the other methods generally exhibited markedly lower accuracies that tended to decline with longer variant length. BayesTyper generally exhibited markedly better sensitivity than the other methods, with the exception of parts of the deletion spectrum in which SVTyper genotyped more variants at the cost of markedly lower accuracy than BayesTyper.

Even for short variants (≤50 nt), BayesTyper significantly outperformed the other methods by achieving near-perfect genotyping accuracy across variant lengths. Importantly, the high accuracy was achieved without sacrificing variant sensitivity as BayesTyper called more variants compared to HaplotypeCaller, the most sensitive alternative method. For instance, for deletions, BayesTyper called 70,566 more variant alleles, while attaining an 8.1% increase in genotyping accuracy compared to HaplotypeCaller. We also observed some improvements for single-nucleotide indels in longer homopolymer tracts that are otherwise known to be difficult to genotype (Supplementary Fig. 1). We further assessed the callers' performance for SNVs and short indels (≤50) based on the GIAB ground-truth set for NA12878[21,22] (Supplementary Fig. 2). These results were generally in line with the results based on the inheritance vectors, although BayesTyper had lower precision for SNVs than the other methods. Interestingly, looking at the marginal contribution of each discovery variant caller to the final PG call set estimated by BayesTyper confirms that multiple variant discovery methods are currently necessary to ensure sensitivity across the variant length spectrum (Supplementary Fig. 3).

On the high-coverage simulation data, where 30× data were simulated for ten YRI individuals based on their 1000G genotype estimates, we observed significant differences between the callers (Supplementary Figs. 4 and 5). BayesTyper generally attained higher sensitivity without sacrificing precision (that is, 1 – false positive rate) as compared to all other callers except for SNVs, where BayesTyper had lower sensitivity than FreeBayes and Platypus, but was superior in terms of precision. Importantly, this was despite few simulated variants in parts of the deletion and most of the insertion spectrum due to poor sensitivity in 1000G in these regions as shown by us[24] and others[6]. In contrast to the PG analysis, we observed a significant drop in accuracy for all methods able to genotype longer variants (BayesTyper, Platypus and SVTyper) for deletions between 100 and 200 nt—a range in which few variants were called in these individuals in 1000G. Receiver operating characteristic curves computed on the simulated data corroborated these findings and confirmed that our results are robust to the choice of genotype quality threshold used (Supplementary Fig. 6).

Finally, we measured the performance of BayesTyper at lower coverage using the GoNL (13×) and simulated (10×) data. The results obtained on the GoNL data generally mirrored their high-coverage equivalents, although BayesTyper here traded an increase in SNV accuracy for a decrease in sensitivity relative to the other methods, which were more sensitive but less accurate (Supplementary Fig. 7). Performance on homopolymers was comparable to the other methods (Supplementary Fig. 8). Essentially the same patterns were observed on the low-coverage simulations, although BayesTyper exhibited worse performance on SNVs (Supplementary Figs. 9–11).

Genotyping using a variation prior. We next sought to investigate whether augmenting the candidate variants discovered using reads from the study individuals with a database of variants detected in previous studies could further improve sensitivity. To do this, we first constructed such a 'variation-prior' database by combining SNVs from dbSNP[18] (common only) with all non-SNVs from dbSNP, 1000G[4,5], GoNL[6,7], the Genotype-Tissue Expression project[25] and GenomeDenmark[24], yielding a database containing a total of 56.2 million and 55.6 million variant alleles for GRCh37 and GRCh38, respectively (Supplementary Table 1). The variation-prior databases are available at https://github.com/bioinformatics-centre/BayesTyper. BayesTyper was then run both on the candidate

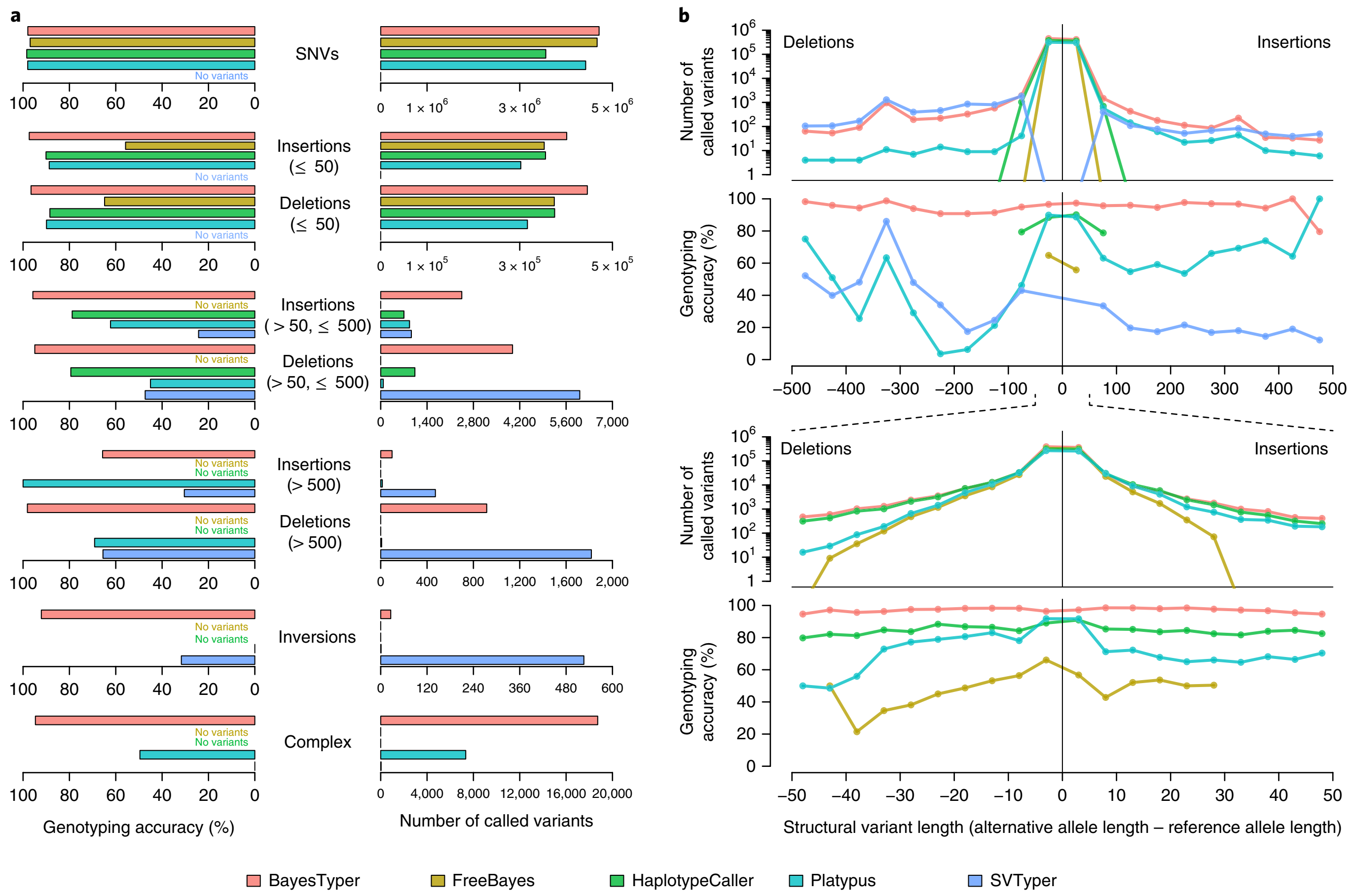

Fig. 2 | Comparison of genotyping methods on PG data (50×). BayesTyper, SVTyper, HaplotypeCaller, Platypus and FreeBayes were run on data from 13 individuals in the PG pedigree using variants discovered by merging calls from four different methods as variant candidate input (Table 1). The number of called variant alleles was used as a measure of sensitivity, whereas the genotyping accuracy was estimated by validating genotypes using pedigree inheritance information. **a**, Sensitivity (right panel) and accuracy (left panel) across variant classes; variants not classified as SNVs, insertions, deletions or inversions were labeled as complex. **b**, Sensitivity (top panel, log scale) and accuracy (bottom panel) for structural variants as a function of the net change in sequence length relative to the reference (50-nt and 5-nt bins for the ±500 and ±50 nt scales, respectively). Variant alleles that do not entail a net change in sequence length (for example, SNVs) were omitted.

set containing only variants discovered from the reads and on the union of these variants and the variation prior (Table 1).

On the PG data, we observed a significant gain in sensitivity for insertions and deletions longer than 50 nt as well as complex variants at the cost of a minor drop in accuracy (Fig. 3a and Supplementary Fig. 12a). Surprisingly, using the prior resulted in a considerable decrease in accuracy for inversions with no gain in sensitivity; however, this affected only 26 variants. On the GoNL data, the variation prior dramatically improved sensitivity for SNVs, insertions and deletions still at the cost of only a minor drop in genotyping accuracy (Fig. 3b and Supplementary Fig. 12b). For instance, the number of short (≤50 nt) insertions and deletions almost doubled from 749,541 called variant alleles without the prior to 1,391,165 when using the prior, at the cost of a less than 1% drop in accuracy.

Discussion

We here present a new method, BayesTyper, for estimating genotypes of a population of individuals based on a database containing candidate variants and k-mer counts from sequencing reads. The method uses a graph-based representation that treats reference and alternative alleles equally and thus mitigates any bias towards the reference sequence. This in turn enables BayesTyper to accurately genotype variants of any class (SNVs, insertions, deletions and so on) and length as demonstrated on both the PG and GoNL data sets with further support from simulations. More specifically, we tested BayesTyper against several state-of-the-art methods on multiple real and simulated data sets and demonstrated that BayesTyper exhibits high absolute sensitivity and accuracy across the variation spectrum. Furthermore, BayesTyper generally performed better than the other methods on non-SNV variants, while performing similarly on SNVs. However, both absolute and relative performance differed across data sets. For instance, the genotyping accuracies for structural variants were generally markedly higher on the GoNL data than on the PG data, and minor discrepancies were also observed between the two different validation strategies employed on the PG calls.

Regarding the PG and GoNL discrepancies, a number of different issues may be involved. First, the PG input variants were obtained by running our panel of discovery methods, whereas the GoNL candidates were simply the SNV, indel and structural variant calls made by the GoNL consortium on the full cohort ($n=769$). Yet, the latter yielded only about twice as many candidates as our PG strategy despite two orders of magnitude more individuals. Hence, the input to the GoNL analyses likely contained mainly high-confidence variants and hence provided less power to detect any differences between the methods. Second, despite using a conservative trio

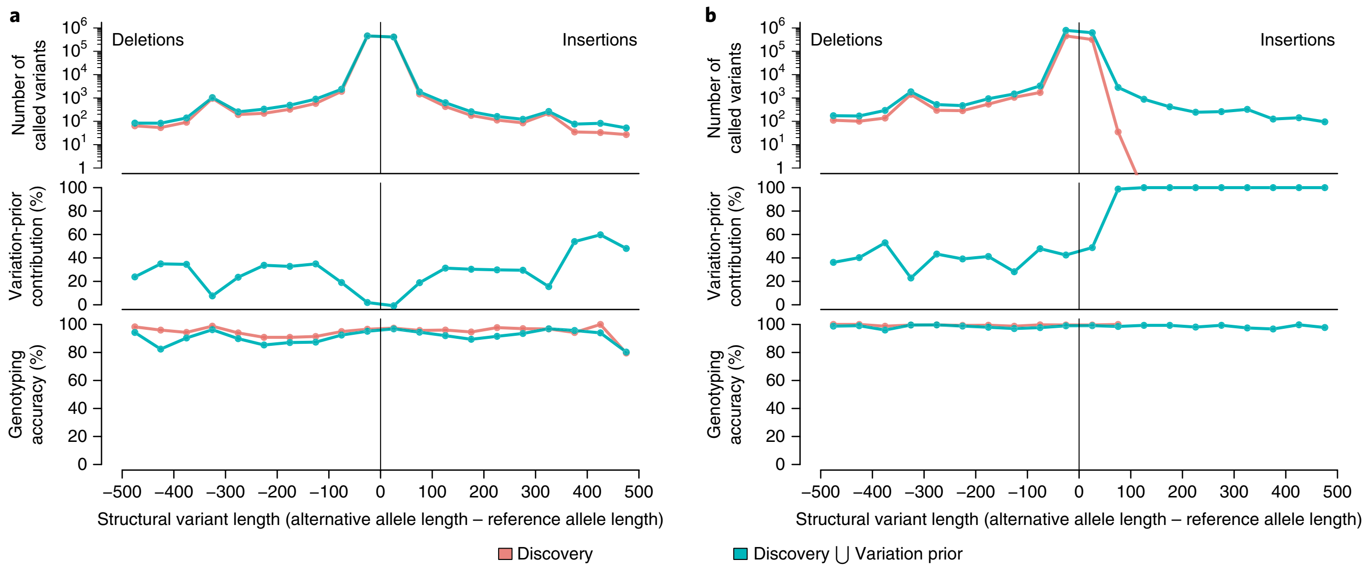

Fig. 3 | Effect of using BayesTyper with a variation prior on structural variation calling performance on the PG (50×) and GoNL (13×) data sets. A 'variation-prior' database was constructed by combining SNVs and structural variants from different databases and studies (Supplementary Table 1). BayesTyper was then run on variant candidates obtained by merging the variation prior with variants discovered using four different methods (Table 1). **a**, The number of called variant alleles when running BayesTyper on the PG cohort (50×) with and without the variation prior (top panel), the fraction of variant alleles contributed by the prior (middle panel) and the genotyping accuracy with and without the prior (bottom panel) as a function of the net change in sequence length relative to the reference (50-nt bins). Genotyping accuracy was estimated by validating genotypes using pedigree inheritance information. **b**, The same analyses as in **a** when running BayesTyper on the GoNL data (13×), where genotyping accuracy was estimated as the fraction of variants with no Mendelian errors across the ten trios. Variant alleles that do not entail a net change in sequence length (for example, SNVs) were omitted.

concordance metric on the GoNL calls, this metric remains susceptible to systematic biases (for example, overcalling of heterozygotes). In contrast, the inheritance vector metric used on the PG data is much more robust to such errors and may end up being quite conservative, as just a single incorrect genotype call results in the entire variant being classified as incorrect. Finally, the discrepancies may also reflect that the methods strike a different balance between sensitivity and accuracy at lower coverage due to increased filtering.

Within the PG analysis, the accuracies seemed to increase when using the GIAB ground-truth set to validate variants relative to the inheritance vector method, especially for FreeBayes. For instance, using the inheritance vectors, BayesTyper and FreeBayes both performed well on SNVs, albeit with BayesTyper exhibiting a higher accuracy. In contrast, the GIAB-based validation suggested FreeBayes to be more accurate, while retaining a higher sensitivity. We speculate that the increased accuracy may be partially explained by the GIAB analysis being restricted to 'high-confidence' regions (~2.5 billion nt) that may be easier to genotype, in contrast to the inheritance vectors, which spanned an additional 20% of the reference sequence. Finally, some of the difference may also be explained by the inheritance vector approach remaining susceptible to a specific type of systematic error (calling all individuals as homozygotes for the same allele will always yield a correct call).

We have generally adhered to 'best practice' recommendations when available for the different methods, but we cannot rule out that their performance might have improved if optimized further. For instance, while the performance of FreeBayes on indels has previously been reported to be inferior to that of HaplotypeCaller, the difference seemed markedly larger in our analyses[26]. Moreover, SVTyper had very high sensitivity, but very low accuracy, in the PG analysis. We note that for HaplotypeCaller, we deviated from the 'best practice' recommendations by performing a full joint genotyping based on the candidate variants and raw read alignments rather than using the GVCF approach. We expect any disadvantage resulting from this choice to be more than outweighed by giving HaplotypeCaller access to both additional variants and the full read alignment files when performing the joint genotyping. Furthermore, we have not investigated the variant resolution (breakpoint and variant length accuracy) of the different methods in detail. However, we required exact breakpoint matches for variants in the simulation and PG/GIAB studies to be labeled as correct and hence have indirect evidence that BayesTyper exhibits high breakpoint accuracy. Finally, we note that both the absolute and relative performance of BayesTyper will be sensitive to the sequencing error rate due to the k-mer approach. For instance, it is currently not clear whether BayesTyper will be applicable to long-read sequencing data (for example, PacBio and Oxford Nanopore) as it will ultimately depend on the error rate, which at least for the Oxford Nanopore system has been steadily improving[27].

The current version of BayesTyper has a large memory footprint, especially when running on high-coverage data using the variation prior, as it currently keeps both k-mer counts and variant graphs in the memory (Supplementary Table 2). However, it is possible to reduce both memory and run-time by decreasing the maximum length of variants genotyped with only minor effects on performance for variants below the threshold (as shown in Supplementary Fig. 13).

BayesTyper does not perform any variant discovery on its own, as the optimal candidate discovery strategy will depend on the study design. For instance, on high-coverage data, we directly demonstrate how combining multiple candidate callers (including de novo assembly) with BayesTyper enables sensitive and accurate genotyping across classes and varying lengths of structural variants. As a new approach, we propose to use a precompiled database of known variants, the 'variation prior', in combination with variants discovered using the reads to improve power. We demonstrate that this approach markedly increases sensitivity without significant costs on accuracy, especially on low-coverage data. Finally, we envision that BayesTyper may foster a new generation of pooled candidate discovery approaches. For instance, one could naively pool low-coverage data from multiple individuals for de novo assembly or use more elegant approaches such as colored de Bruijn graphs[28] to generate candidate variants. These can then be used as input to BayesTyper and effectively provide access to the full variation spectrum, even on low-coverage data.

In summary, we present an accurate and unbiased probabilistic method for genotyping variants of any class and length from high-throughput sequencing data that we expect will enable complex variant analyses to become a standard procedure when analyzing genomic data of any coverage.

Methods

Methods, including statements of data availability and any associated accession codes and references, are available at https://doi.org/10.1038/s41588-018-0145-5.

Received: 15 June 2016; Accepted: 20 April 2018;
Published online: 18 June 2018

References

1. Li, H. Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM. Preprint at https://arxiv.org/abs/1303.3997 (2013).
2. DePristo, M. A. et al. A framework for variation discovery and genotyping using next-generation DNA sequencing data. *Nat. Genet.* **43**, 491–498 (2011).
3. Rimmer, A. et al. Integrating mapping-, assembly- and haplotype-based approaches for calling variants in clinical sequencing applications. *Nat. Genet.* **46**, 912–918 (2014).
4. 1000 Genomes Project Consortium. A global reference for human genetic variation. *Nature* **526**, 68–74 (2015).
5. Sudmant, P. H. et al. An integrated map of structural variation in 2,504 human genomes. *Nature* **526**, 75–81 (2015).
6. Genome of the Netherlands Consortium. Whole-genome sequence variation, population structure and demographic history of the Dutch population. *Nat. Genet.* **46**, 818–825 (2014).
7. Hehir-Kwa, J. Y. et al. A high-quality human reference panel reveals the complexity and distribution of genomic structural variants. *Nat. Commun.* **7**, 12989 (2016).
8. Garrison, E. & Marth, G. Haplotype-based variant detection from short-read sequencing. Preprint at https://arxiv.org/abs/1207.3907 (2012).
9. Handsaker, R. E. et al. Large multiallelic copy number variations in humans. *Nat. Genet.* **47**, 296–303 (2015).
10. Sirén, J. Indexing variation graphs. Preprint at https://arxiv.org/abs/1604.06605 (2016).
11. Paten, B., Novak, A. M., Eizenga, J. M. & Garrison, E. Genome graphs and the evolution of genome inference. *Genome Res.* **27**, 665–676 (2017).
12. Schneeberger, K. et al. Simultaneous alignment of short reads against multiple genomes. *Genome Biol.* **10**, R98 (2009).
13. Huang, L., Popic, V. & Batzoglou, S. Short read alignment with populations of genomes. *Bioinformatics* **29**, i361–i370 (2013).
14. Sirén, J., Välimäki, N. & Mäkinen, V. Indexing graphs for path queries with applications in genome research. *IEEE/ACM Trans. Comput. Biol. Bioinform.* **11**, 375–388 (2014).
15. Dilthey, A., Cox, C., Iqbal, Z., Nelson, M. R. & McVean, G. Improved genome inference in the MHC using a population reference graph. *Nat. Genet.* **47**, 682–688 (2015).
16. Eggertsson, H. P. et al. Graphtyper enables population-scale genotyping using pangenome graphs. *Nat. Genet.* **49**, 1654–1660 (2017).
17. Deorowicz, S., Kokot, M., Grabowski, S. & Debudaj-Grabysz, A. KMC 2: fast and resource-frugal k-mer counting. *Bioinformatics* **31**, 1569–1576 (2015).
18. Sherry, S. T. et al. dbSNP: the NCBI database of genetic variation. *Nucleic Acids Res.* **29**, 308–311 (2001).
19. Chiang, C. et al. SpeedSeq: ultra-fast personal genome analysis and interpretation. *Nat. Methods* **12**, 6–10 (2015).
20. 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.* **27**, 157–164 (2017).
21. Zook, J. M. et al. Integrating human sequence data sets provides a resource of benchmark SNP and indel genotype calls. *Nat. Biotechnol.* **32**, 246–251 (2014).
22. Zook, J. M. et al. Extensive sequencing of seven human genomes to characterize benchmark reference materials. *Sci. Data* **3**, 1–26 (2016).
23. Chen, X. et al. Manta: rapid detection of structural variants and indels for germline and cancer sequencing applications. *Bioinformatics* **32**, 1220–1222 (2016).
24. Maretty, L. et al. Sequencing and de novo assembly of 150 genomes from Denmark as a population reference. *Nature* **548**, 87–91 (2017).
25. Chiang, C. et al. The impact of structural variation on human gene expression. *Nat. Genet.* **49**, 692–699 (2017).
26. Hwang, S., Kim, E., Lee, I. & Marcotte, E. M. Systematic comparison of variant calling pipelines using gold standard personal exome variants. *Sci. Rep.* **5**, 17875 (2015).
27. Jain, M., Olsen, H. E., Paten, B. & Akeson, M. The Oxford Nanopore MinION: delivery of nanopore sequencing to the genomics community. *Genome Biol.* **17**, 239 (2016).
28. Iqbal, Z., Caccamo, M., Turner, I., Flicek, P. & McVean, G. De novo assembly and genotyping of variants using colored de Bruijn graphs. *Nat. Genet.* **44**, 226–232 (2012).

Acknowledgements

This work was supported by grants from the Novo Nordisk Foundation (grant number NNF10SA1016550) to A.K. and Innovation Fund Denmark (grant number 019-2011-2). We thank the Platinum Genomes project and the Genome of the Netherlands consortium for providing access to their data.

Author contributions

J.A.S. designed and implemented the algorithm, performed the analyses and wrote the manuscript. L.M. designed and implemented the algorithm, performed the analyses and wrote the manuscript. A.K. designed the algorithm and wrote the manuscript.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information is available for this paper at https://doi.org/10.1038/s41588-018-0145-5.

Reprints and permissions information is available at www.nature.com/reprints.

Correspondence and requests for materials should be addressed to A.K.

Publisher's note: Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Methods

Variant graph construction and alignment. The input variants and reference sequence are represented as a graph. To query the reads for variant information, we use exact matching of k-mers between variant alleles in the graph and the sequencing reads and we therefore need to obtain k-mer profiles of both variants and reads.

For the sequencing reads, we simply count k-mers for each individual using the KMC3 (v3.0.0)[17] program with default parameters except for *k*, which was set to 55, and the minimum number of k-mer occurrences (-ci), which was set to 1.
We define the k-mer profile of an allele to be the multiset of all k-mers overlapping that allele (for example, for a biallelic SNV, the reference and alternative allele profiles would each contain 55 k-mers). The k-mer profiles of variants less than $k-1$ nucleotides apart are dependent and they therefore need to be genotyped together. More specifically, rather than performing genotyping in the space of single alleles, we instead cluster variants where the breakpoints are less than $k-1$ nucleotides apart and work in the space of combinations of their constituent alleles (that is, haplotypes). Furthermore, variant clusters within the deleted or copied sequence of an upstream large structural variation are also dependent. We therefore define these clusters as belonging to the same variant cluster group and model their dependency structure as a tree as shown in Supplementary Fig. 14. We define a copied reference sequence as the longest downstream region for which at least 50% of the k-mers in the region are identical to the k-mers in one of the upstream variant alleles (reference and alternative alleles). Variants inside a structural variant that remove or substitute larger parts of the reference sequence (for example, deletions and inversions) are defined as nested and nested variants can contain other nested variants (for example, due to a deletion within a deletion).

The objective is then for each cluster in each group to estimate the individual's diplotype (that is, haplotype pair) from which the genotypes for each variant can be directly obtained. Optimally, we would consider all possible haplotypes corresponding to all paths in the variant graph within a cluster, but this is not possible for larger clusters as the number of possible haplotypes grows exponentially with the number of variants. Instead, we use a heuristic based on k-mer occurrences in the reads to select a set of candidate haplotypes. However, as we also want to use information from the variant graph to limit the number of read k-mers considered, we use an iterative approach.

We first construct a set of variant graphs (one for each cluster) from the variant and reference input (Fig. 1a), where nodes represent sequences (reference or variant allele sequences) and edges represent possible haplotypic connections between the node sequences. The variation graph is directly specified by the cluster variants and reference sequence, and variants within inserted sequences are therefore not collapsed and the representation will thus contain some sequence redundancy.

A set of haplotype candidates is then generated in each variant graph using a heuristic *n*-best algorithm (Fig. 1b). In this algorithm, we first construct a Bloom filter (false positive rate 0.001) of read k-mers for each individual from the KMC3 output using the ntHash[29] library to enable memory-efficient queries of sample k-mer support. Independently for each individual, we then enumerate the *n*-best paths (that is, haplotypes) in the graph using a two-rank scoring system based on the information in the Bloom filters. More specifically, the paths are first ranked by the fraction of k-mers covered that are observed in the individual's read k-mer Bloom filter. Second, a minimum set cover approach is used, ranking paths by the number of observed (covered by at least two k-mers present in the filter) vertices in the path that are not observed in a path with a higher ranking. Using this ranking, a two-step filtering approach is used to select the best scoring paths. First, the top 16 paths with an observed vertex score above zero are kept. Second, if fewer than 16 paths have an observed vertex score above zero, the remaining top ranking paths are added until 16 is reached or all paths have been added. The paths are ranked and filtered at each vertex during graph traversal. After traversal, the final set of paths for an individual is selected using the same two-step approach; however, paths that cover unobserved vertices are excluded in the second step of this iteration. This final filter is introduced to decrease computation time by limiting the number of poor haplotypes that are evaluated in the subsequent genotype inference step. Finally, the best paths are combined across individuals to create the final set of haplotype candidates. Only paths with a unique haplotype sequence are kept during graph traversal and the final merging step; the path covering the highest number of vertices is selected when collapsing redundant paths.

To limit the number of k-mers that are read into memory, we then construct a Bloom filter (false positive rate 0.001) from the k-mer profiles of the haplotype candidates and use this to select which k-mer counts from the KMC3 output are read into memory. To estimate the genomic rate distribution, the counts of an additional 10 million, randomly selected k-mers from non-variable regions of the genome (that is, k-mers not in the Bloom filter) are also read into memory.

Finally, k-mers that satisfy one of the following criteria are removed: is observed in one of the supplied decoy sequences (that is, sequences with unknown ploidy, such as the mitochondrial genome); is observed more than 127 times in the non-variable regions and all variant graphs combined; is observed in more than one variant cluster group. The output from this part of the algorithm step is then a set of candidate haplotypes (represented as k-mer profiles) for each cluster and a table with k-mer counts for every k-mer observed in both a haplotype candidate and at least one sequencing read across individuals.

Genotyping model. The objective is to determine the most likely local diplotypes (that is, pairs of haplotypes) for the population given the read k-mer counts. To solve this, we created a model for how k-mer counts are generated from an individual's diplotype. Note that, even though the model is formulated for estimating diplotypes, it easily generalizes to clusters with other ploidy levels, such as haploid clusters (for example, the Y chromosome or clusters nested inside a heterozygous deletion). Currently BayesTyper supports both haploid and diploid chromosomes.

A schematic representation of the model is illustrated in Fig. 1c. To allow for sharing of information between individuals, we let each haplotype *h* in the set of haplotype candidates *H* be associated with a random variable *f* that models its population frequency[30]. As the input variant set will contain alleles that are not present in the population, and as we expect variants within a cluster to be in linkage disequilibrium, we expect only a subset of the haplotypes to actually be present in the population. We therefore need to model sparsity in the haplotype frequencies (that is, that only some of the frequencies will be non-zero) and introduce a sparse-Dirichlet distribution with sparsity parameter π as a prior on these frequencies[31]. We then model the generation of each individual's diplotype $d \in D = \{(a,b) | a \in H \wedge b \in H \wedge a \leq b\}$ as a draw from the multinomial distribution specified by the frequencies *F* and the ploidy. We note that this construct effectively assumes that the population is in Hardy–Weinberg equilibrium.

For each individual, the vector of observed read k-mer counts *C* in the cluster is then modeled as generated from a combination of the sampled diplotype *d*, the non-variable genomic regions (that is, between variant clusters) and a noise source. More specifically, each k-mer count *c* is generated from either the individual's genome $P(c_d|m,p,r)$ or sequencing noise $P(c_n|\lambda)$ dependent on whether it is observed in the genome (diplotype or non-variable genomic region) or not:

$$\begin{aligned} P(c|m,p,r,\lambda) &= 1(m>0) \times P(c_d|m,p,r) \\ &\quad + 1(m=0) \times P(c_n|\lambda) \end{aligned}$$

where the multiplicity *m* is defined as:

$$m = m_d + m_g$$

Here, m_d specifies the total number of occurrences of the k-mer in the diplotype *d* and m_g denotes the number of occurrences in the non-variable regions of the genome.

The k-mer counts from the genome ($m>0$) are generated by drawing from the negative binomial distribution:

$$P(c_d|m,p,r) = \binom{c_d + mr - 1}{c_d} p^{mr} (1-p)^{c_d}$$

where the parameters *p* and *r* are specific to each individual, but shared across clusters. The negative binomial distribution is used, as sequencing read (and thus k-mer) counts are overdispersed relative to the Poisson distribution. Counts from k-mers with multiplicity (number of occurrences) higher than one are modeled by scaling the size parameter *r* with the multiplicity *m*. Finally, we truncate the negative binomial distribution at 255 to handle that k-mer counts from KMC are capped at this value.

The noise k-mer counts ($m=0$) are expected to predominantly originate from sequencing errors and are modeled using the Poisson distribution with parameter λ, which is specific to each individual but shared across clusters. A gamma distribution, which is shared between individuals, is used as a prior distribution over the parameter λ.

Genotype inference. The inference objective is to estimate the posterior distribution over diplotypes, haplotype frequencies and count distribution parameters given a vector of k-mer counts for each individual in a population. Inference is conducted using collapsed Gibbs sampling as described in the Supplementary Note.

Filtering. Post-filtering of the BayesTyper call set was conducted to handle errors arising from data properties not completely accounted for by the model using four 'hard' filters implemented in BayesTyperTools filter: one variant-level filter (remove variants with only heterozygote genotypes) and three genotype-level filters (remove genotypes containing alleles with fewer than one sampled k-mer (NAK); remove genotypes containing alleles with a fraction of observed (non-zero coverage) k-mers (FAK) below $t_{\mathrm{FAK}} = 1 - e^{-0.275 \times \hat{\mu}}$, where $\hat{\mu}$ is the sample-specific mean of the negative binomial distribution used to model the k-mer coverage; remove genotypes with a genotype posterior probability below 0.99).

The default filter parameters were chosen on the basis of simulations of three individuals from the Utah Residents with Northern and Western European Ancestry (CEU) population in the 1000G project[4,5] and data from five GenomeDenmark parent–offspring trios[24].

Variation prior. A database of already known variation was created by combining SNVs from dbSNP (common only) with all non-SNVs from dbSNP and four large sequencing projects using BayesTyperTools (Supplementary Table 1). The database was created for both the GRCh37 and GRCh38 reference genomes by lifting over variants that were called on only one of the reference builds using Crossmap (v0.2.6)[32]. Reference and alternative alleles containing ambiguous nucleotides were removed.

Simulated data. Reads were simulated from ten randomly chosen female individuals from the YRI population in the 1000G project[4,5]. More specifically, symbolic alleles in the 1000G phase 3 call set were first converted to sequences (excluding mitochondrial mobile insertions) using BayesTyperTools and all variants were left-aligned and normalized using bcftools (v1.4)[33]. Next, haplotype sequences from the 10 individuals were constructed using the phased genotype calls and two sets (10× and 30× coverage) of paired-end 100-base-pair reads simulated from each haplotype using pIRS (v2.0.0)[34] with default settings (insert size mean = 180 and s.d. = 18).

To construct a set of candidate variants, each read set was first mapped to the reference genome (GRCh37) and sorted using BWA-MEM (v0.7.15)[1] and SAMtools (v1.4)[35], respectively, followed by duplication marking using Picard Tools (v2.6.0) (https://github.com/broadinstitute/picard) and base quality score recalibration according to the GATK best practices workflow (v3.6)[2]. Candidate variants were predicted independently for the high- and low-coverage data sets using four different methods, each capable of discovering deletions and insertions with breakpoint accuracy. Default settings were used for all methods except that local assembly was enabled for Platypus (Supplementary Table 3). HaplotypeCaller was given the recalibrated mapped reads, whereas FreeBayes, Platypus and Manta were given the reads after duplication marking. Variant candidates were called independently for each individual. Symbolic alleles in the Manta call set were converted to sequence using BayesTyperTools and all sets were left-aligned and normalized using bcftools (v1.4). Finally, the four predicted call sets were combined to create a single candidate set using BayesTyperTools (Table 1).

The genotyping accuracies of BayesTyper, FreeBayes, Platypus, HaplotypeCaller and SVTyper were then assessed on these candidates. FreeBayes, Platypus and SVTyper were given the mapped reads after duplication marking, whereas HaplotypeCaller was given mapped, duplicate-marked reads with recalibrated base quality scores. For the HaplotypeCaller and FreeBayes, all variants larger than 100 nt were first removed as the methods could not run on these, whereas for Platypus all variants larger than 10,000 nt were removed. Deletions, duplications and inversions longer than 50 nt were converted to precise symbolic alleles (that is, without breakpoint uncertainty) and provided as input to SVTyper. Variants longer than 500,000 nt were excluded by BayesTyper to reduce memory and computation time. Default settings for all methods were used besides specifying that only the input variants should be genotyped and all variants should be written to output regardless of quality (Supplementary Table 4). Instead of the mapped reads, BayesTyper was given a database of the read k-mers ($k = 55$) created using KMC3 (v3.0.0).

The output call set for each of the four methods was post-processed using the recommended pipeline by each developer (Supplementary Table 4). The same filtering pipeline was used for both the 10× and the 30× data. The BayesTyper calls were filtered using BayesTyperTools with default parameters (see above), whereas vcflib (https://github.com/vcflib/vcflib) was used to filter the FreeBayes calls using recommended parameters. The Platypus calls were filtered using the method's default parameter settings. Variant Quality Score Recalibration was applied to the HaplotypeCaller calls in concordance with their best practice workflow (albeit without using InbreedingCoeff as an annotation due to the low number of samples) before filtration. No filtering besides variant and genotype quality was used on the SVTyper calls, as no filter recommendations were available. For all call sets, only variants on autosomal chromosomes with a variant quality (QUAL) of at least 20 were kept and low-confidence genotypes (GQ < 20) were removed (corresponding to a genotype posterior probability below 0.99 for BayesTyper). The genotype quality filter was omitted when calculating the receiver operating characteristic curves. All filtering was performed using bcftools (v1.4) if not specified otherwise.

To evaluate the genotyping performance of the four methods, the filtered call sets were compared to the YRI call set used to generate the simulated sequencing reads (ground truth), using vcfeval from RTG Tools (v3.7.1)[36]. This tool was chosen as it handles potential differences in variant encoding between call sets by performing the comparison at the haplotype level. Since vcfeval does not handle missing alleles, used to indicate nested variation in BayesTyper, all of these were converted to the reference alleles before performing the comparison.
The options –ref-overlap and –all-records were given to vcfeval to allow for evaluation of nested variation and to ensure that all variants were evaluated after removing filtered variants and genotypes, respectively. For variants containing an allele larger than 1,000 nt, which are currently not supported by vcfeval, genotyped alleles on the same position were compared directly instead. This simpler approach was chosen on the basis of the observation that most of these larger variants are primitive deletion, duplication or inversions for which a haplotype level approach is not needed. The evaluation was performed independently for each individual. Sensitivity was defined as the fraction of alleles in the ground-truth genotypes, where both alleles in the genotype were found in the estimated call set (that is, homozygote alleles are counted twice). Precision was defined as the fraction of alleles in the estimated genotypes where both alleles in the genotype were found in the ground-truth call set. Sensitivity and precision were calculated across all called positions for each individual in the ground truth and estimated call set, respectively, with the exception of variants that were in regions vcfeval deemed too 'hard' to evaluate. Finally, both metrics were aggregated across individuals to provide a single sensitivity and accuracy measurement across all ten individuals. Variants were classified as either SNVs, insertions, deletions or inversions, where inversions were defined as alternative alleles of equal length to the reference that were at least 10 nt long and matched the reverse complement of the reference allele with no more than 5% mismatches. Variants not falling into any of these categories were labeled as complex.

PG data. Paired-end, 50× PCR-free sequencing data from the 2 parents and 11 children of the CEPH pedigree 1463 generated by the PG project[20] were downloaded from dbGaP (accession phs001224). Candidate variants were predicted from these 13 individuals using the same pipeline as described for simulated data (Supplementary Table 3), although GRCh38 was used as the reference genome instead of GRCh37. The predicted variants, together with the variation prior (Supplementary Table 1), were combined into two variant candidate input sets: one containing only variants discovered from the reads and one containing the union of these variants and the variation prior (Table 1). Genotyping and filtering was performed using the same pipelines as described for simulated data (Supplementary Table 4). The genotype estimates were evaluated using haplotype transmission information essentially as described in an earlier study[20]. In brief, haplotype inheritance vectors and the pedigree members' haplotype configurations were obtained from dbGaP (accession phs001224). These were then used to validate variant genotypes by defining a variant to be correct if a single variant haplotype assignment configuration that explains the estimated genotypes across all pedigree members existed using custom software implemented in BayesTyperTools. Variants not contained within an inheritance vector interval were excluded. Accuracy was defined as the fraction of alleles with correct pedigree haplotype transmission. To account for the fact that variants with many filtered samples are more likely to be labeled as correct due to chance, each allele was weighted by the number of unfiltered genotypes when computing the accuracy. Single-nucleotide indels (as compared against all other alleles at the same variant position) were classified as homopolymers if the first nucleotide after the variant position was identical to the deleted or inserted nucleotide. Homopolymer lengths were calculated in both directions from the first downstream variant position on the reference genome. Variant types were classified as described for simulations. SNVs and indel genotyping performance was further tested against the GIAB 'high-confidence calls' for NA12878 (v3.3.2, GRCh38)[21,22] with sensitivity and precision estimated using vcfeval as for simulated data. Variants outside the 'high-confidence' regions were excluded by vcfeval.

GoNL data. Paired-end, 13× sequencing data from ten parent–offspring trios sequenced as part of the GoNL project[6] were downloaded from the GoNL consortium servers after access was granted by the consortium. The sequencing data of the 30 individuals were mapped to the GRCh37 reference genome using the same pipeline as described for the simulated data. Two distinct variant candidate input sets were created: one containing all variants called across all 769 individuals in the GoNL project (GoNL SNPs release 5 and GoNL SV release 6) and one containing the union of these variants and the variation prior (Table 1). Genotyping and filtering was performed using the same pipelines as described for simulations (Supplementary Table 4). For BayesTyper, the genotyping was performed in three batches of ten individuals (mothers, fathers and children in separate batches), whereas all 30 individuals were genotyped together for the other methods. Trio concordance (1 − Mendelian error) was used to assess the accuracy of the predicted variants. Only trios where all three genotypes were unfiltered were included. Accuracy was defined as the fraction of alleles for which all genotype calls were concordant. As for the haplotype transmission metric, each allele was weighted by the number of unfiltered trios when computing the accuracy. Homopolymer classification was carried out as for the PG data and variant types were classified as described for simulations.

Reporting Summary. Further information on experimental design is available in the Nature Research Reporting Summary linked to this article.

Code availability. All custom software including BayesTyper and BayesTyperTools used to perform genotyping as well as pre- and postprocessing of data for genotyping is freely available as both source code and static builds at https://github.com/bioinformatics-centre/BayesTyper under the MIT license. BayesTyper v1.2 was used for the analyses in this paper. A number of custom R scripts were used to generate the figures and tables presented in this paper; these are available upon request.

Data availability. All raw real data used in this study were produced by the PG project (dbGaP accession phs001224) and the GoNL consortium (available by

request to the consortium). Data derived from these data will be available upon request subject to the constraint that access has been granted by the original study authors. Simulations were based on genotypes publically available through the 1000G project, and both raw and derived data are available upon request.
The 'variation-prior' databases (GRCh37 and GRCh38 builds) are available at https://github.com/bioinformatics-centre/BayesTyper.

References

29. Mohamadi, H., Chu, J., Vandervalk, B. P. & Birol, I. NtHash: Recursive nucleotide hashing. *Bioinformatics* **32**, 3492–3494 (2016).
30. Albers, C. A. et al. Dindel: accurate indel calls from short-read data. *Genome Res.* **21**, 961–973 (2011).
31. Maretty, L., Sibbesen, J. A. & Krogh, A. Bayesian transcriptome assembly. *Genome Biol.* **15**, 501 (2014).
32. Zhao, H. et al. CrossMap: a versatile tool for coordinate conversion between genome assemblies. *Bioinformatics* **30**, 1006–1007 (2014).
33. Li, H. A statistical framework for SNP calling, mutation discovery, association mapping and population genetical parameter estimation from sequencing data. *Bioinformatics* **27**, 2987–2993 (2011).
34. Hu, X. et al. pIRS: Profile-based Illumina pair-end reads simulator. *Bioinformatics* **28**, 1533–1535 (2012).
35. Li, H. et al. The Sequence Alignment/Map format and SAMtools. *Bioinformatics* **25**, 2078–2079 (2009).
36. Cleary, J. G. et al. Comparing variant call files for performance benchmarking of next-generation sequencing variant calling pipelines. Preprint at https://www.biorxiv.org/content/early/2015/08/03/023754 (2015).

natureresearch

Corresponding author(s): Anders Krogh

Reporting Summary

Nature Research wishes to improve the reproducibility of the work that we publish. This form provides structure for consistency and transparency in reporting. For further information on Nature Research policies, see Authors & Referees and the Editorial Policy Checklist.

Statistical parameters

When statistical analyses are reported, confirm that the following items are present in the relevant location (e.g. figure legend, table legend, main text, or Methods section).

n/a	Confirmed	
☒	☐	The exact sample size (*n*) for each experimental group/condition, given as a discrete number and unit of measurement
☒	☐	An indication of whether measurements were taken from distinct samples or whether the same sample was measured repeatedly
☒	☐	The statistical test(s) used AND whether they are one- or two-sided *Only common tests should be described solely by name; describe more complex techniques in the Methods section.*
☒	☐	A description of all covariates tested
☒	☐	A description of any assumptions or corrections, such as tests of normality and adjustment for multiple comparisons
☒	☐	A full description of the statistics including central tendency (e.g. means) or other basic estimates (e.g. regression coefficient) AND variation (e.g. standard deviation) or associated estimates of uncertainty (e.g. confidence intervals)
☒	☐	For null hypothesis testing, the test statistic (e.g. *F*, *t*, *r*) with confidence intervals, effect sizes, degrees of freedom and *P* value noted *Give P values as exact values whenever suitable.*
☐	☒	For Bayesian analysis, information on the choice of priors and Markov chain Monte Carlo settings
☒	☐	For hierarchical and complex designs, identification of the appropriate level for tests and full reporting of outcomes
☒	☐	Estimates of effect sizes (e.g. Cohen's *d*, Pearson's *r*), indicating how they were calculated
☒	☐	Clearly defined error bars *State explicitly what error bars represent (e.g. SD, SE, CI)*

Our web collection on statistics for biologists may be useful.

Software and code

Policy information about availability of computer code

Data collection	No software was used to collect data.
Data analysis	All custom software used to perform genotyping as well as pre- and postprocessing of data for genotyping are available at github.com/bioinformatics-centre/BayesTyper (v1.2). A number of custom R scripts were used to generate the figures and tables presented in the manuscript; these are available upon request. Finally, a number of freely available software packages were used: KMC3 (v3.0.0, github.com/refresh-bio/KMC), GATK/HaplotypeCaller (v3.6 and v3.7, software.broadinstitute.org/gatk), Platypus (v0.8.1, www.well.ox.ac.uk/platypus), FreeBayes (v1.1.0, github.com/ekg/freebayes), SVTyper (v0.1.4, github.com/hall-lab/svtyper), manta (v1.0.3, github.com/Illumina/manta), Crossmap (v0.2.6, crossmap.sourceforge.net), bcftools/samtools (v1.4, www.htslib.org), pIRS (v2.0.0, github.com/galaxy001/pirs), bwa-mem (v0.7.15, github.com/lh3/bwa), Picard Tools (v2.6.0, github.com/broadinstitute/picard), vcflib (github.com/vcflib/vcflib), vcfeval (v3.7.1, github.com/RealTimeGenomics/rtg-tools)

For manuscripts utilizing custom algorithms or software that are central to the research but not yet described in published literature, software must be made available to editors/reviewers upon request. We strongly encourage code deposition in a community repository (e.g. GitHub). See the Nature Research guidelines for submitting code & software for further information.

Data

Policy information about availability of data

All manuscripts must include a data availability statement. This statement should provide the following information, where applicable:
- Accession codes, unique identifiers, or web links for publicly available datasets
- A list of figures that have associated raw data
- A description of any restrictions on data availability

All raw real data used in this study was produced by the Platinum Genomes project (available in dbGaP, accession phs001224) and the Genome of the Netherlands Consortium (available by request to the consortium). Simulations were based on genotypes publically available through the 1000 Genomes Project. Data derived from these raw data will be available upon request subject to the constraint that access has been granted by the original study authors (for the Platinum Genomes and GoNL data). The "variation-prior" databases (GRCh37 and GRCh38 builds) are available at https://github.com/bioinformatics-centre/BayesTyper.

Field-specific reporting

Please select the best fit for your research. If you are not sure, read the appropriate sections before making your selection.

☒ Life sciences ☐ Behavioural & social sciences

For a reference copy of the document with all sections, see nature.com/authors/policies/ReportingSummary-flat.pdf

Life sciences

Study design

All studies must disclose on these points even when the disclosure is negative.

Sample size	No statistical methods were used to determine the sample size. The main objective of the analyses presented in this paper is to determine the absolute and relative genotyping performance for which a single individual generally provides a sufficient number of variants (for the variant classes and lengths presented in this paper) to detect even small differences in accuracy. Genotyping was performed on a population of individuals to both test the methods' ability to use population information as well as to enable population-based validation metrics. All experiments were limited to a maximum of 30 individuals to limit the computational resources used.
Data exclusions	All methods generally received the same input although variants longer than 100 nts were excluded from HaplotypeCaller and FreeBayes and variants longer than 10,000 nts from Platypus as they were unable to process these. Variants longer than 500,000 nts were excluded from BayesTyper to limit the computational resources used.
Replication	No attempts to replicate the findings on multiple, similar datasets were made due to a scarcity of data, where validation can be performed. Instead genotyping performance on real data was evaluated on two datasets (PG and GoNL) with different characteristics in terms of sequencing depth and genotype validation strategy and hence it is not possible to perform a direct comparison between the two analyses. However, both analyses supported the overall conclusions of the paper with some minor discrepancies discussed at length in the discussion section of the paper.
Randomization	For simulations, ten individuals genotyped in the 1000 Genomes project were selected randomly among Yoruba females from Ibadan, Nigeria (YRI). For GoNL, ten parent-offspring were selected randomly among the entire GoNL cohort.
Blinding	No group allocation was used.

Materials & experimental systems

Policy information about availability of materials

n/a	Involved in the study
☒	☐ Unique materials
☒	☐ Antibodies
☒	☐ Eukaryotic cell lines
☒	☐ Research animals
☒	☐ Human research participants

Method-specific reporting

n/a	Involved in the study
☒	☐ ChIP-seq
☒	☐ Flow cytometry
☒	☐ Magnetic resonance imaging