nature genetics

TECHNICAL REPORT

https://doi.org/10.1038/s41588-018-0316-4

Fast and accurate genomic analyses using genome graphs

Goran Rakocevic[1,2,3], Vladimir Semenyuk[1,2,3], Wan-Ping Lee[1], James Spencer[1,2], John Browning[1,2], Ivan J. Johnson[1,2], Vladan Arsenijevic[1,2], Jelena Nadj[1,2], Kaushik Ghose[1,2], Maria C. Suciu[1,2], Sun-Gou Ji[1,2], Gülfem Demir[1,2], Lizao Li[1,2], Berke Ç. Toptaş[1,2], Alexey Dolgoborodov[1], Björn Pollex[1,2], Iosif Spulber[1], Irina Glotova[1,2], Péter Kómár[1,2], Andrew L. Stachyra[1,2], Yilong Li[1,2], Milos Popovic[1,2], Morten Källberg[1], Amit Jain[1,2] and Deniz Kural[1,2*]

The human reference genome serves as the foundation for genomics by providing a scaffold for alignment of sequencing reads, but currently only reflects a single consensus haplotype, thus impairing analysis accuracy. Here we present a graph reference genome implementation that enables read alignment across 2,800 diploid genomes encompassing 12.6 million SNPs and 4.0 million insertions and deletions (indels). The pipeline processes one whole-genome sequencing sample in 6.5 h using a system with 36 CPU cores. We show that using a graph genome reference improves read mapping sensitivity and produces a 0.5% increase in variant calling recall, with unaffected specificity. Structural variations incorporated into a graph genome can be genotyped accurately under a unified framework. Finally, we show that iterative augmentation of graph genomes yields incremental gains in variant calling accuracy. Our implementation is an important advance toward fulfilling the promise of graph genomes to radically enhance the scalability and accuracy of genomic analyses.

Completion of the first draft of the human reference genome[1,2] was a landmark achievement in human genetics, establishing a standardized coordinate system for annotating genomic elements and comparing individual human genomes. The reference genome serves as a scaffold for mapping and assembling short DNA sequencing reads into longer consensus contigs, thus underpinning the quality of all ensuing analyses and ultimately the ability to draw conclusions of clinical significance from DNA sequencing.

The current human reference genome is represented as a linear haploid DNA sequence[3]. This structure poses practical limitations due to the prevalence of genetic diversity in human populations: any given human genome has, on average, 3.5–4.0 million SNPs or indels and ~2,500 large structural variations (SVs) compared with the reference genome[4,5]. This genetic divergence may cause sequencing reads to map incorrectly or to fail to map altogether[6,7], particularly when they span SV breakpoints. Read-mapping accuracy thus varies significantly across genomic regions in a given sample and across genetically diverged samples. Misplaced reads may in turn result in both missed true variants (false negatives) and incorrectly reported false variants (false positives), as well as hamper other applications that rely on accurate read mapping. Identifying SVs is particularly challenging: despite the large number of SVs already characterized[5], most methods for genotyping SVs still rely on detecting complex combinations of abnormal read alignment patterns to detect SVs[8,9], although more recent algorithms such as BayesTyper[10] can take into account known SVs.

Recent large-scale resequencing efforts have comprehensively catalogued common genetic variants[4,11,12], prompting suggestions to make use of this information through multigenome references[13,14], which have been suggested to alleviate reference bias by facilitating read mapping[15,16]. Despite these promising observations, currently available implementations of multigenome graph references are either orders of magnitude slower than conventional linear reference genome-based methods on human whole-genome sequencing (WGS) data (one example is BWBBLE[17]) or are intended for use with small genomes[13] and small regions within large genomes[15,16,18,19]. Graphtyper[19] is a recently published tool that performs local realignment of reads initially aligned by a linear aligner. Although whole-genome workflows using graph genomes are under active development[20,21], the full genome-wide impact of using multigenome references for human genomic analyses has only recently begun to be assessed[20,22].

Here we present a graph genome pipeline for building, augmenting, storing, querying and variant calling from graph genomes composed of a population of genome sequences. We show that graph genomes improve the mapping accuracy of next-generation sequencing reads on the genome-wide level. Our next-generation sequencing read alignment and variant calling pipeline, Graph Genome Pipeline, leverages our graph genome data structure and outperforms the state of the art linear reference-genome pipeline[23] comprising BWA-MEM and GATK HaplotypeCaller[23], as measured by multiple complementary benchmarks. By including breakpoint-resolved SV polymorphisms into the graph genome, we demonstrate that SVs can be genotyped rapidly and accurately in a unified fashion. As novel genetic variation data are accumulated in graph genomes, incremental improvements in read mapping and variant calling accuracy can be achieved. This will allow our approach to scale and improve with expanding genetic variation catalogs.

Results

A computationally efficient graph genome implementation. We implemented a graph genome data structure that represents

[1]Seven Bridges Genomics, Inc, Cambridge, MA, USA. [2]Totient, Inc, Cambridge, MA, USA. [3]These authors contributed equally: Goran Rakocevic, Vladimir Semenyuk. *e-mail: deniz.kural@sbgenomics.com

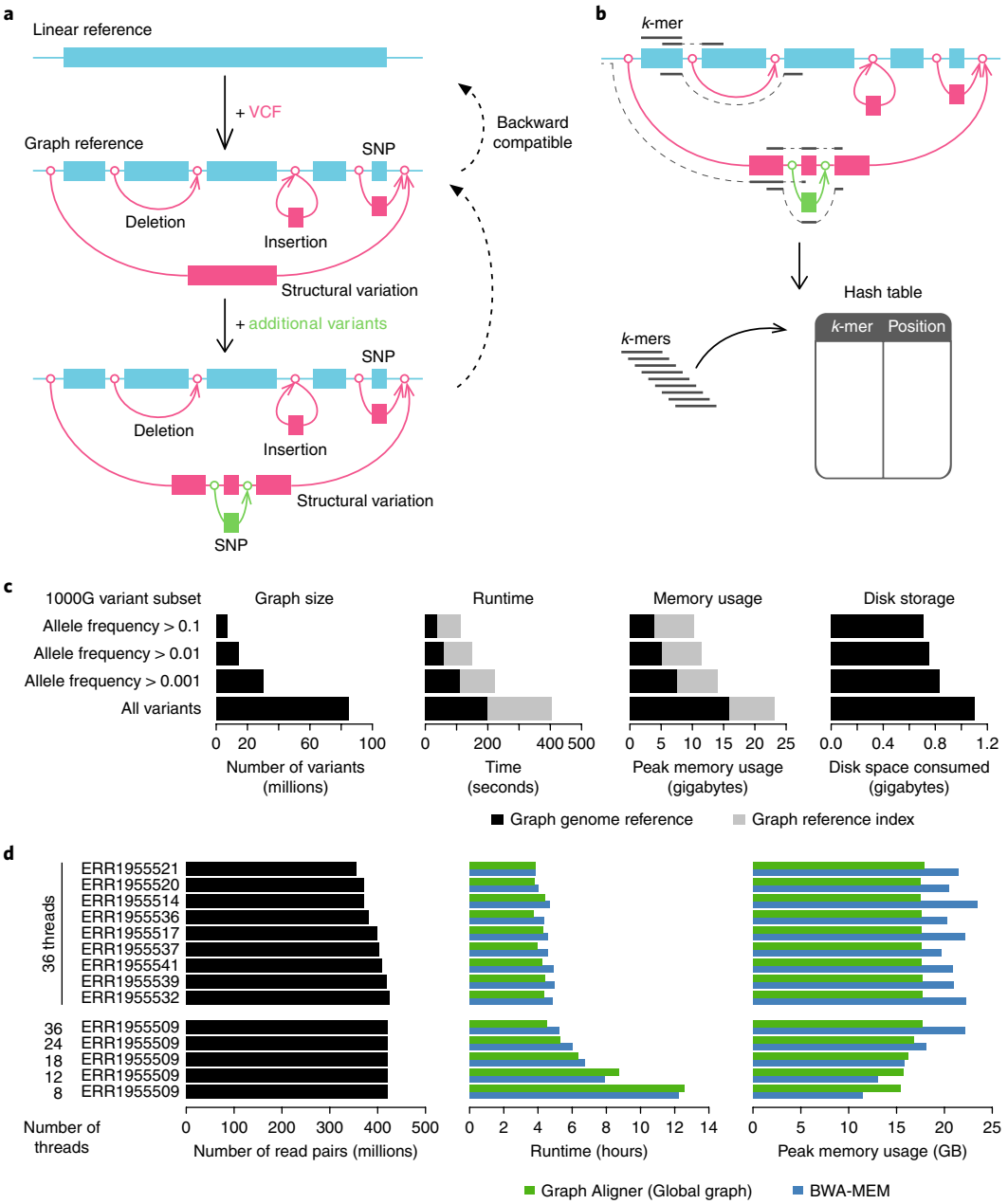

Fig. 1 | The graph genome architecture and computational resource requirements. a, A graph genome is constructed from a standard linear reference genome FASTA file augmented by a set of genetic variants provided in VCF format. A graph genome can further be augmented with additional genetic variants in a second VCF file or, in the case of variants within variants, using Graph Genome Pipeline. The coordinate system of each constructed graph genome is backward compatible with that of the linear reference genome. Each segment between vertices corresponds to an edge in the graph; inserting a variant to the graph can therefore add two (for insertions) or three (other variants) edges to the graph as the original edge is split into separate edges (as needed) at the start and end vertices of the new edge as well as the edge corresponding to the additional variant. The three graphs shown contain 1, 9 and 13 edges, respectively. A rendering of these graphs constructed using Graph Genome Pipeline is shown in Supplementary Figure 1. **b**, A graph genome is indexed by creating a hash table with *k*-mers along all possible paths of the graph as keys and their corresponding graph genome positions as values. These *k*-mer positions can then be used as seeds for aligning sequencing reads against the graph. **c**, Computational resource requirements of building, indexing and storing graph genomes on one high-coverage WGS sample. All tests were performed using a single thread on the Amazon AWS instance type c4.8xlarge. **d**, Runtime and memory usage for BWA and Graph Aligner using the global graph for ten randomly selected samples from the Coriell cohort. Both BWA-MEM and Graph Aligner were executed using 36 threads on the Amazon AWS cloud instance type c4.8xlarge.

genomic sequences on the edges of the graph (Fig. 1a and Supplementary Note). A graph genome is constructed from a population of genome sequences, such that each haploid genome in this population is represented by a sequence path through the graph. To facilitate the use of widely available datasets, we implemented a process to build a graph genome using VCF files indicating genetic variants with respect to a standard linear reference genome, which is provided using a FASTA file. For representational purposes, the linear reference genome path is labeled as the initial edge, and all coordinates of genetic variants are reported with respect to it. This method ensures backward compatibility of graph coordinates to linear reference genome coordinates. In practice, a graph genome is

built by iteratively adding edges corresponding to a non-reference allele, terminating at nodes corresponding to genomic loci on the initial edge. Insertions are represented as cyclic edges starting and terminating at the same node, but our mapping algorithm enforces acyclic traversal of the graph (Fig. 1a) by requiring that a path traverses a cycle at most once and traverses at most one cycle from each vertex. Genomic features such as tandem repeat expansions and inversions are represented as insertions or sequence replacements in the graph. Our graph genome infrastructure supports building and aligning reads against general graph topologies such as hierarchical variation (Fig. 1a and Supplementary Fig. 1). However, unlike VG and HISAT2, we do not support bidirectionality or cycles, because such structures impose unnecessary computational complexity; our directed acyclic graph data structure is able to fully describe all genetic variation encapsulated in the sequential representation of nucleotides comprising chromosomes. For querying a graph genome, we use a hash table that associates short sequences of length k (k-mers) along all valid paths in the graph with their graph coordinates (Fig. 1b). Uninformative k-mers that occur exceptionally frequently are omitted (Supplementary Note).

We used the 1000 Genomes (1000G) Phase 3, Simons Genome Diversity, and other variant datasets to construct a graph genome reference that we refer to as the global graph (Supplementary Note, Supplementary Table 1 and Supplementary Figs. 2 and 3). The global graph can be built and indexed in less than 10 min in total (Fig. 1c). Such a graph reference can be stored in less than 30 gigabytes (GB) of memory or just over 1 GB of disk space (Fig. 1c and Supplementary Note). Loading a stored graph reference into memory takes less than 2 min. Over a tenfold range in the number of variants included (the reference genome remains constant), time, memory and disk storage consumption only grew around fourfold, twofold and 50%, respectively (Fig. 1c).

Improved read mapping accuracy using graph genomes. To support genomic analyses on our graph genome implementation, we developed a graph aligner for short reads that uses the k-mer index for seeding (Fig. 1b and Supplementary Fig. 1) and then local read alignment against the graph (Supplementary Note). The read alignments against a path in the graph are projected to the standard reference genome and output to a standard BAM file, with the alignment path along the graph reported using custom annotation tags. Thus, the output format of our graph aligner maintains full compatibility with existing genomics data processing tools. When an unambiguous projection is not possible (for example, for reads fully mapped within a long insertion variant), the reads are placed to the closest reference position, so that downstream analysis tools can access these reads conveniently.

We measured the graph aligner runtimes on ten randomly selected high-coverage whole-genome sequencing datasets (Supplementary Table 2). Read alignment against the global graph containing around 16 million variants (Supplementary Table 1) required around 4.5 h per sample when using 36 threads, which was on average 9% shorter than that of BWA-MEM[24] (Fig. 1d). This trend is reversed when only 8 or 12 threads are used, but the runtimes remain comparable (Supplementary Table 3). The peak RAM memory usage of the Graph Aligner was approximately 17.5 GB compared with an average of 21 GB of BWA-MEM (Supplementary Table 3). Overall mapping coverage between Graph Aligner and BWA-MEM are similar, with small differences in low- and high-coverage regions (Supplementary Fig. 4).

In order to test the read mapping accuracy of the graph aligner, we simulated sequencing reads from individual samples drawn from VCFs from the 1000G[4] and the GiaB[25] projects (Supplementary Note and Supplementary Fig. 5). While reads without any variants are mapped equally accurately to the reference genome by the Graph Aligner and BWA-MEM, the Graph Aligner maintains a high mapping rate and accuracy even in reads containing long indels relative to the standard linear reference genome (Fig. 2 and Supplementary Figs. 6–8). Less than 1% of the reads with >10-base pair (bp) insertions and deletions are mismapped using the 1000G graph, whereas this number is two and three times as high with BWA-MEM, respectively (Fig. 2). Even against a linear reference without variants, our graph aligner is able to align more reads containing indels than BWA-MEM (Fig. 2).

Graph Genome Pipeline improves recall in variant detection. Graph Genome Pipeline (Supplementary Fig. 9) calls variants, including SVs, using a reassembly variant caller and variant call filters, as suggested previously[23] (Supplementary Note). Generating variant calls from raw FASTQs in the Coriell cohort (29–42× coverage) took on average 6 h 19 min ($\sigma = 25$ min) using Graph Genome Pipeline on 36 CPU cores and 20 GB of memory. In comparison, the best practices GATK pipeline using GATK HaplotypeCaller (https://software.broadinstitute.org/gatk/best-practices/, hereafter referred to as BWA-GATK) executed on the same hardware required 50 GB of memory and an average of 11 h 30 min ($\sigma = 3$ h 16 min) of runtime.

We devised four independent and complementary experiments to compare the variant calling accuracy of our Graph Genome Pipeline against that of two commonly used linear pipelines (BWA-GATK and BWA-Freebayes) as well as a recently published graph-based approach (Graphtyper). Furthermore, to separate the impact of the graph aligner and from the variant caller, we also benchmarked GATK HaplotypeCaller results derived from Graph Aligner BAMs (Fig. 3b and Supplementary Table 4, which also presents results from BayesTyper, though we note that BayesTyper is not designed to detect variants not present in the graph, placing a limit on the recall with the graphs we use in this paper). The first benchmarking experiment is based on sequencing data simulation, which provides a known ground truth for all variants throughout the genome, but likely incompletely reproduces the error modalities of real sequencing data. The second benchmarking experiment uses truth data established by the Genome in a Bottle Consortium (GiaB)[25] for five high-coverage whole-genome sequenced samples (50× coverage). These truth data cover only about 70% of the genome considered as 'high-confidence' regions by GiaB, likely excluding the ~30% of the genome that is hardest to align and call variants against. The third variant-calling benchmark is based on measuring Mendelian consistency in family trios (Supplementary Figs. 10–12), which is an indirect proxy for variant calling accuracy, but can be conducted on real data throughout the genome. To support this approach, we developed computational methods to resolve variant representation differences in trio comparison and estimate the precision and recall rates of a variant caller using variant calls derived independently from each member of a family trio (Supplementary Note). Finally, we compared the variant calling results to SNP genotyping results from two commonly used SNP array platforms (Supplementary Note, Supplementary Fig. 13, and Supplementary Table 5).

In addition to the standard approach of filtering the false positive variants that we mainly use in this paper, we have also developed a machine learning–based approach (Supplementary Note). Although this method yields a significant improvement in the results and outperforms all other pipelines in the PrecisionFDA Truth contest (Supplementary Figs. 6 and 7 and Supplementary Table 6), we do not use it as the main indicator of the Graph Genome Pipeline's performance, as it relies on training using one of the GiaB samples, which are extensively used in most of our benchmarks.

A consistent pattern emerges from the benchmarking experiments. In both SNP and indel calling, Graph Genome Pipeline either has an equally good precision with better recall (Fig. 3a,b and Supplementary Tables 4 and 7) or better precision with the same recall (Fig. 3c and Supplementary Table 8) compared with other

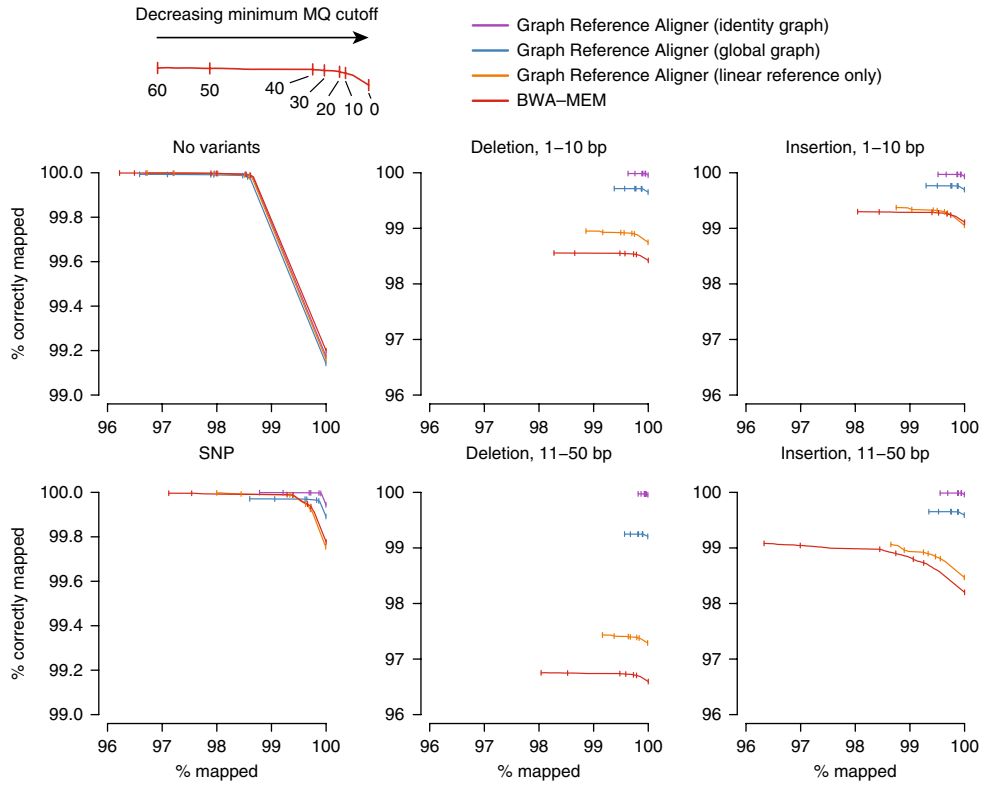

Fig. 2 | Read mapping accuracy using BWA-MEM and graph genomes. Simulated reads were divided into six categories based on the type of simulated variants they contain relative to the linear reference genome. The percentage of reads mapped correctly was plotted against the percentage of reads mapped for a range of mapping quality (MQ) cutoffs. 'Identity graph' refers to a graph genome containing only the genetic variants present in the respective target sample.

pipelines. Graph Genome Pipeline has the lowest rate of Mendelian violations and calls the second highest number of variants after Graphtyper (Fig. 3d,e, Supplementary Fig. 16 and Supplementary Tables 9 and 10), but Graphtyper has lower precision overall (Fig. 3). The gain in SNP calling accuracy is driven by graph alignment, as SNPs called by GATK-HC from Graph Aligner BAMs have a recall similar to those by BWA-GATK (Fig. 3). Interestingly, Graph Genome Pipeline's good performance in indel recall is driven primarily by our graph alignment–aware reassembling variant caller (Fig. 3). Overall, Graph Genome Pipeline has a similar precision to BWA-GATK but improves recall by around 0.5%, corresponding to around 20,000 additional true variants being detected when extrapolated genome-wide.

The GiaB variant call sets provide an estimate for the practical upper limit in achievable accuracy using the standard linear reference genome, because they are carefully curated from an extensive amount of high-quality data generated from a combination of several different sequencing platforms and meta-analyzed across a suite of state-of-the-art bioinformatics tools[25]. Interestingly, among the variants detected by Graph Genome Pipeline but asserted as homozygous reference by GiaB, a substantial proportion (26–42% across four samples, Supplementary Table 11) exhibited strong support from alternative sequencing technologies and in terms of Mendelian concordance (Supplementary Fig. 17). These variants are often located in variant-dense regions, half (52%) of them are part of the global graph, and most of the remaining variants (46%) are phased with one or multiple nearby variants present in the graph (Supplementary Fig. 17 and Supplementary Table 12). Contrary to the linear reference genome, Graph Aligner is by design able to map reads across known variations without reference bias, which allows it to mitigate the impact of reference bias in repetitive or variant-dense genomic regions. We therefore hypothesized that these variants could be real but missed by all other linear reference genome–based pipelines used by GiaB due to reference bias. We successfully carried out Sanger sequencing at 351 and 598 of these "false FP" variants in two GiaB samples, HG001 and HG002, validating 63.6% and 60% of these variants as real variants missed by GiaB, respectively (Supplementary Tables 11 and 13). Although these numbers constitute only 8.7% and 15.5% of the total number of FP calls made by our pipeline, they demonstrate that our graph genome implementation is able to overcome some practical accuracy limitations of linear reference approaches.

A unified framework for SV calling using Graph Genome Pipeline. Sequence information of known SVs can be incorporated into a graph genome, allowing reads to be mapped across them. Graph Genome Pipeline is able to align reads across SVs, whereas BWA-MEM fails to do so with both short Illumina reads (Fig. 4a and Supplementary Figs. 17 and 18) and long PacBio reads, even when PacBio reads are aligned using parameters tuned for PacBio data (Supplementary Fig. 18).

To demonstrate that reads spanning SV breakpoints can be used to directly genotype SVs, we manually curated a dataset of 230 high-quality, breakpoint-resolved deletion-type SVs (Supplementary Table 14 and Supplementary Note) and genotyped them across 49 individuals from the Coriell cohort, for which the true SV genotypes are available from the 1000 Genomes Project[5]. Although our SV set does not include any events composed purely of inserted sequence, many of them involve novel sequence insertions at their breakpoints (Supplementary Fig. 3). The fractions of reads spanning SV breakpoints segregates cleanly into three clusters based on SV genotype (Fig. 4b), suggesting that graph genome-based SV genotyping could be accomplished, even with a simple read counting–based method (Supplementary Note and Supplementary Table 15). On the basis of

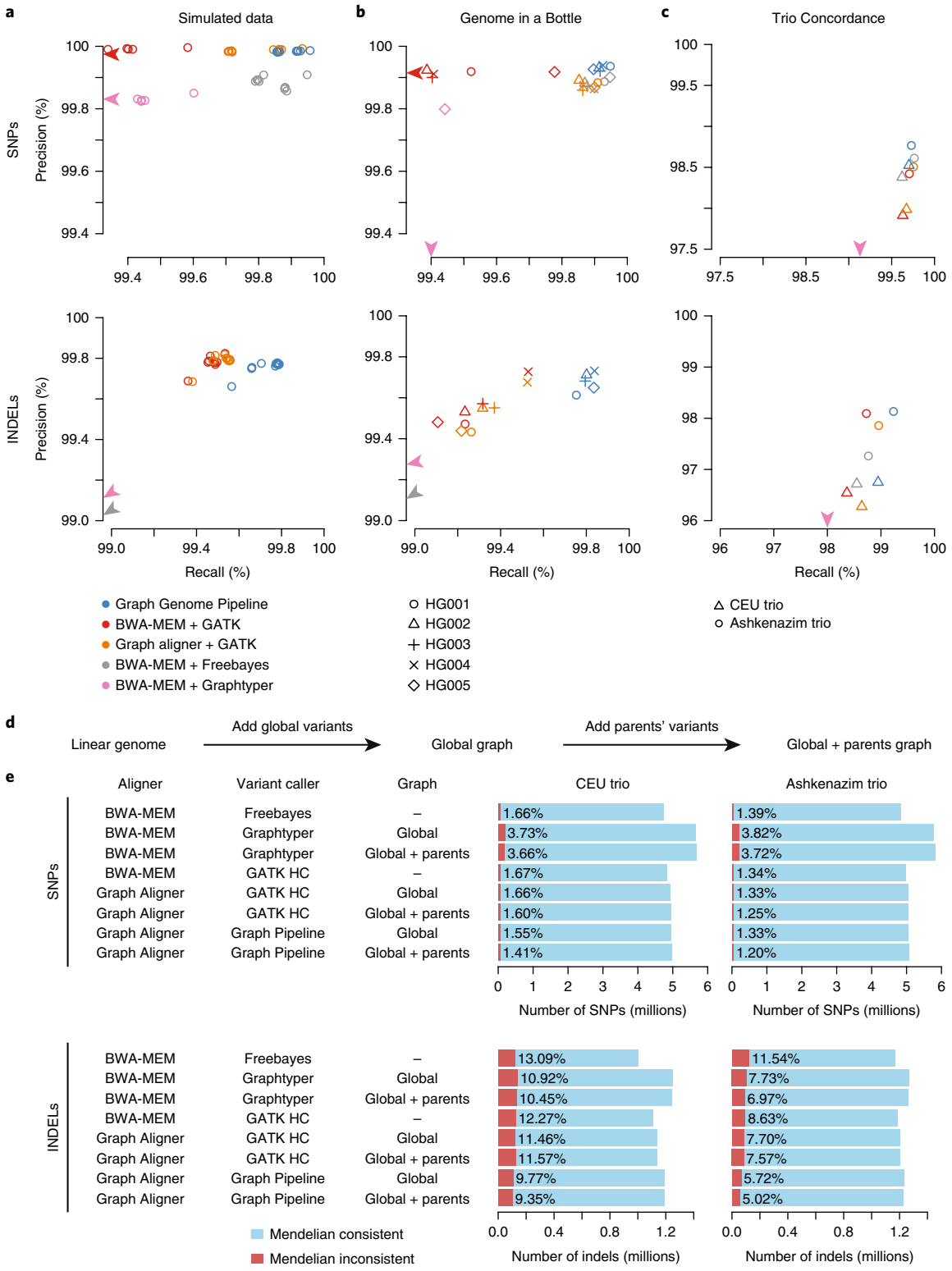

Fig. 3 | Variant calling benchmarking between Graph Genome Pipeline and BWA-GATK. a, Precision and recall in simulated sequencing data. Every point corresponds to a different simulated individual. **b**, Precision and recall benchmarking based on five samples for which the Genome in a Bottle truth sets are available. We show BWA-GATK results with Hard Filtering, as that approach performed better out of the two recommended options. Supplementary Table 10 shows BWA-GATK results with VQSR, as well as results from BayesTyper. **c**, Statistically estimated precision and recall from the Mendelian consistency data derived from two family trios with independently called variants for each family member. **d**, Schematic representation of the trio concordance analysis. **e**, Mendelian concordance of two trios analyzed using BWA-GATK, Graph Genome Pipeline with the global graph and Graph Genome Pipeline using a global graph augmented by the variants detected in either parent of each family. The numbers shown are computed after resolving variant representation differences (Supplementary Note). Mendelian concordance rates without variant representation resolution are shown in Supplementary Figure 16 and Supplementary Table 13.

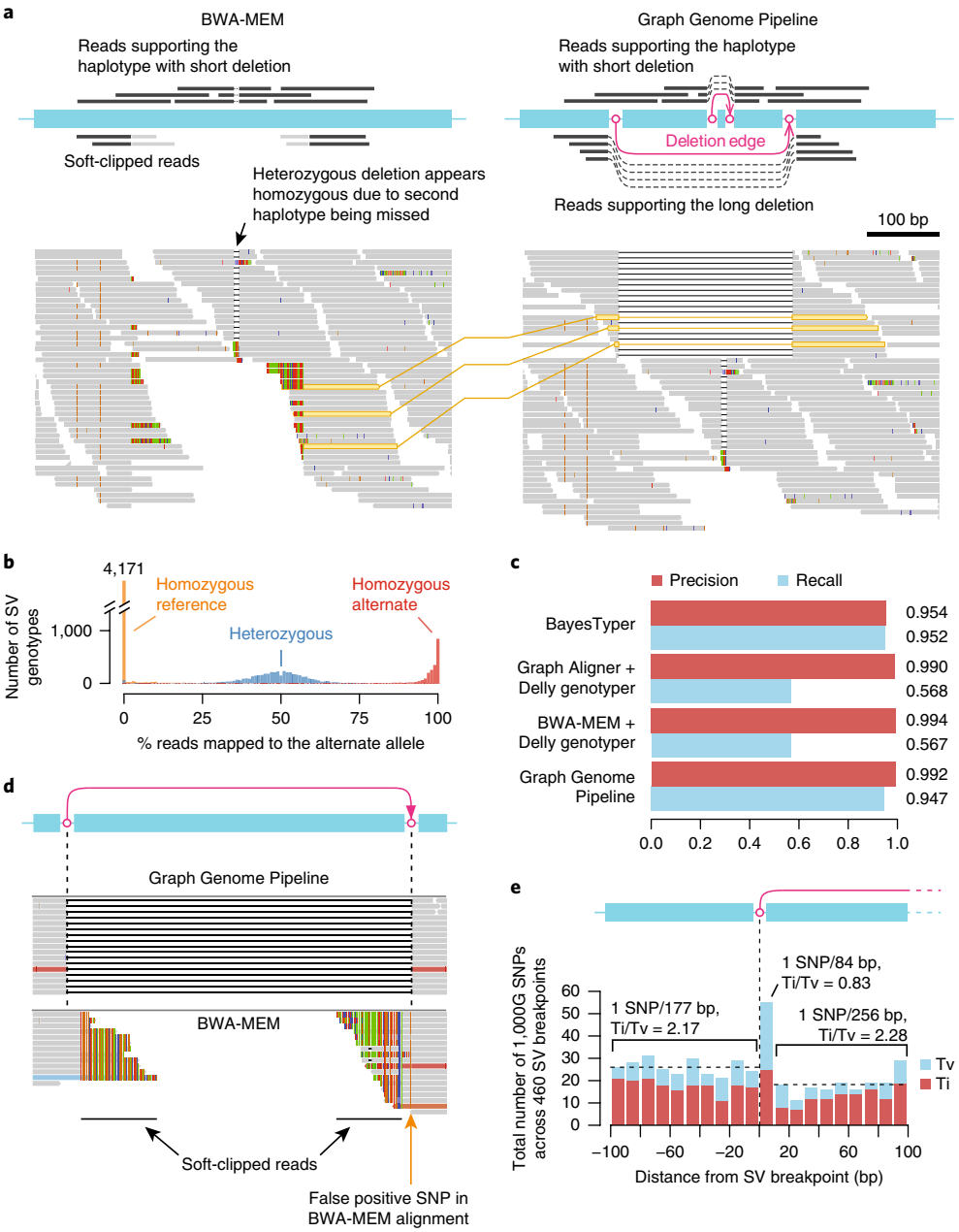

Fig. 4 | SV genotyping using graph genomes. a, An example Integrative Genomics Viewer (IGV) screenshot of BWA-MEM and Graph Aligner alignments over an SV at chromosome 4 position 132,524,277-132,524,548 (GRCh38, Database of Genomic Variants ID: esv3602264), which, in this sample (HG01628), overlaps with a small deletion variant. BWA-MEM is able to align reads across the small deletion, but not across the SV. Instead, reads on the SV haplotype become soft clipped (colored vertical bars at the tips of the reads facing the SV). As a consequence of the SV not being called, the small intervening deletion appears homozygous in the read alignment. A schematic of the read alignment pattern is shown above the screenshot. Because this SV is included in the graph genome, Graph Aligner correctly aligns reads across it, revealing the SV haplotype. **b**, The fraction of reads aligning to an SV branch at an SV breakpoint colored by the true genotype of each SV in each sample, summarized across 230 SVs and 49 individuals. Orange, homozygous reference; blue, heterozygous; red, homozygous alternate. **c**, Benchmarking the graph genome pipeline against three other SV genotyping tools (BayesTyper, Delly2) across 230 SVs and 49 individuals. BayesTyper and Graph Genome Pipeline use raw FASTQ as input, whereas Delly2 requires aligned reads. Within the curated 230 SVs, BayesTyper and Graph Genome Pipeline outperform Delly2 on recall (Supplementary Note). All tools perform equally well in precision. **d**, Example of an alignment that causes a false positive SNP due to misalignment against the linear reference genome. This sample has a homozygous deletion in this region, and Graph Aligner (top) aligns reads successfully across it. BWA-MEM (bottom) fails to align reads across the SV, but because the SV has 20 bp of imperfect microhomology at the breakpoint, BWA-MEM aligns the reads on the right, resulting in a spurious single-nucleotide mismatch. **e**, The total number of transition and transversion SNPs in 1000G aggregated over the 460 breakpoints of 230 SVs and grouped by distance to an SV breakpoint in 10-bp bins. Positive distances (>0 bp) are within an SV and negative distances (<0 bp) are outside an SV. For example, positions +1 to +10 bp correspond to the 10 bp closest to an SV breakpoint within a deletion, and positions −1 to −10 bp correspond to the 10 bp closest to an SV breakpoint outside a deletion.

these reads, the Graph Genome Pipeline reassembles SVs (alongside SNPs and indels) with an SV genotyping accuracy comparable to those of current SV callers (Fig. 4c and Supplementary Table 15). Its SV genotyping performance within the 230 SVs is on par with BayesTyper, and although both Delly and Graphtyper have high precision, they suffer from low recall (Fig. 4c and

Supplementary Table 15). However, we acknowledge that the SV set presented here lacks more complex variants such as mobile elements and inversions, and Graph Genome Pipeline's performance with complex SVs is yet to be fully tested.

To compare the SV genotyping performance of the graph aligner to competing technologies, we focused on the GiaB sample HG002, for which both Illumina read and PacBio long read data are publicly available. We manually examined the alignment results from each technology in the 230 SVs in this sample by the graph aligner. BWA-MEM is unable to align Illumina reads across any of these SVs (Fig. 4a and Supplementary Figs. 18 and 19). Similarly, PacBio long read alignment fails in all of these SVs, even when aligned with BWA-MEM using parameters tuned for PacBio data (Supplementary Fig. 19). Thus, graph genomes could potentially improve SV genotyping for short-read and long-read sequencing technologies alike.

Two events among the 230 curated SVs, esv3642033 and esv3638126, are in strong linkage disequilibrium with SNPs significantly associated ($P < 5 \times 10^{-8}$) with breast cancer (rs1436904) and obesity class II risk (rs11639988), respectively[26,27] (Supplementary Fig. 19). We were able to correctly genotype the presence of these two SVs in 48/49 and 49/49 samples, respectively. Thus, graph genome technology may enable integrated methods for genotyping of common SVs, including those of clinical and biological relevance.

Graph genomes prevent erroneous variant calls around SVs. Structural variations mediated by certain DNA repair mechanisms can exhibit microhomology, including imperfect microhomology, around their breakpoints[28]. If an aligner is not aware of an SV, sequencing reads spanning the SV could become erroneously aligned over a region of imperfect microhomology instead, causing mismatches over the region to be spuriously reported as SNPs and indels (Fig. 4d). To quantify this effect in 1000G, we compared the rate of 1000G SNPs around the SV breakpoints with the background rate of SNPs in 1000G (Fig. 4e). These metrics are computed in aggregate over all 230 SVs. Within the deleted portions of the 230 curated deletion SVs combined, the aggregate rate of 1000G SNPs is 1.8 per bp, and these SNPs have a transition/transversion ratio (Ti/Tv) of 2.28, which is expected from real biological SNPs[29] (Fig. 4e). In contrast, in the first 10 bp immediately after an SV breakpoint, the aggregate 1000G SNP rate is increased threefold to 5.5/bp, suggesting that 67% of 1000G SNPs called within 10 bp of an SV breakpoint are false (Fig. 4d,e). The Ti/Tv of these SNPs is 0.83, deviating considerably from the expected ratio of 2.1. Assuming that spurious SNPs have an expected Ti/Tv of 0.5, the FP SNP rate over this region estimated using Ti/Tv is 79%, reaching a similar value to that estimated using total SNP counts (Supplementary Note). In addition to FP variant calls, we also encountered examples where variants overlapping with an SV erroneously appear homozygous, because BWA-MEM fails to align reads across the SV and thus fails to detect the corresponding SV haplotype (Fig. 4a). Thus, using population variation information in graph genomes can mitigate variant calling and genotyping errors around SVs.

Incremental improvement in variant calling recall through iterative graph augmentation. As common genetic variants continue to be catalogued across populations, newly discovered variants can be incrementally added to existing graph genomes to increase the comprehensiveness of the graph while maintaining backward compatibility to samples analyzed using earlier versions of the graph (Fig. 1a). To test whether incremental graph augmentation would improve variant calling, we augmented the global graph with variants detected in ten samples from three super-populations of the Coriell cohort as well as a Qatar genome project cohort[30] (Fig. 5a,b and Supplementary Table 16) and compared the variant calls obtained using the global graph and the four augmented global graphs. Augmenting the global graph genome increases the number of known variants (present in dbSNP) discovered slightly (5% and 10% median increase in known SNPs and indels discovered, respectively; Fig. 5c,d and Supplementary Table 16). However, the augmented graphs result in almost twice the number of novel (those not present in dbSNP) SNPs and indels being called (Fig. 5c,d and Supplementary Table 16).

We measured the quality of the detected variants indirectly using Ti/Tv and heterozygous-to-homozygous alternate allele ratio (het/hom) for SNPs and indels, respectively[29]. For each tested pipeline, known SNPs and indels have a Ti/Tv ratio of 2.04–2.06 and het/hom ratio of 1.3–2.0, respectively (except for indels called by the graph pipeline in the African population). These values fall within the expected range for common variants[29]. In contrast, these metrics fall outside the expected range for novel SNPs and indels. The Ti/Tv ratios of novel SNPs are closer to the expected Ti/Tv when called by Graph Genome Pipeline compared with BWA-GATK, whereas the ordering is reversed for the het/hom ratios of novel indels (Fig. 5c,d). Importantly, although graph augmentation results in more known and novel variants being called, their Ti/Tv and het/hom ratios remain unaffected.

Graph augmentation had a similar impact in the other read alignment and variant calling experiments. Read alignment recall reaches almost 100% if a target sample is aligned against a graph genome that contains all of its actual variants (Fig. 2). Likewise, trio concordance and variant calling recall is further improved if variant calling in a child is performed using a graph genome augmented with variants detected in the respective parents (Fig. 3c–e and Supplementary Tables 8–10). Thus, incremental augmentation of graph genome references yields cumulative improvements in variant calling recall without an accompanying decrease in precision.

Discussion

Our benchmarking experiments demonstrate that using a graph genome reference improves read mapping and variant calling recall, including that of SVs, without a concomitant loss in precision (Figs. 2–5). Our graph aligner is able to readily align reads across breakpoint-resolved SVs included in the graph, unlike linear reference genome–based methods even with long reads (Fig. 4a and Supplementary Fig. 18 and 19). Direct genotyping of SVs using these reads is possible in some cases (Supplementary Table 15). In contrast, existing methods for identifying and genotyping SVs require specifically designed multistep algorithms[8]. Graph Genome Pipeline allows the identification and genotyping of SVs and small variants in a single unified process, raising the prospect for population-scale SV genetics and association studies.

Currently, Graph Genome Pipeline analyzes samples individually, and a version that performs joint variant calling[23,31] is under development. Further improvements to variant calling could be achieved by representing haplotypes of small variants and SVs as paths in the graph genome. Information such as allele frequencies of each variant and linkage disequilibrium between them could be incorporated into the graph, providing additional statistical information for read alignment and variant calling. This approach could provide a computationally efficient alternative to joint variant calling in leveraging previously accumulated population genetics information when analyzing a newly sequenced sample. Future efforts from us and others[32] will be required to assess the benefits of using graph genomes with encoded allele frequency and linkage disequilibrium information.

The potential benefits of an unbiased multigenome graph reference are not limited to variant calling, but cover the full range of genomics research. Graph genomes provide an unbiased, representative scaffold for read alignment, which is critical to sequencing alignment quantification applications such as RNA-sequencing analysis, ChIP-seq analysis and CNV calling. Our graph genome implementation can also be used to encode information other than

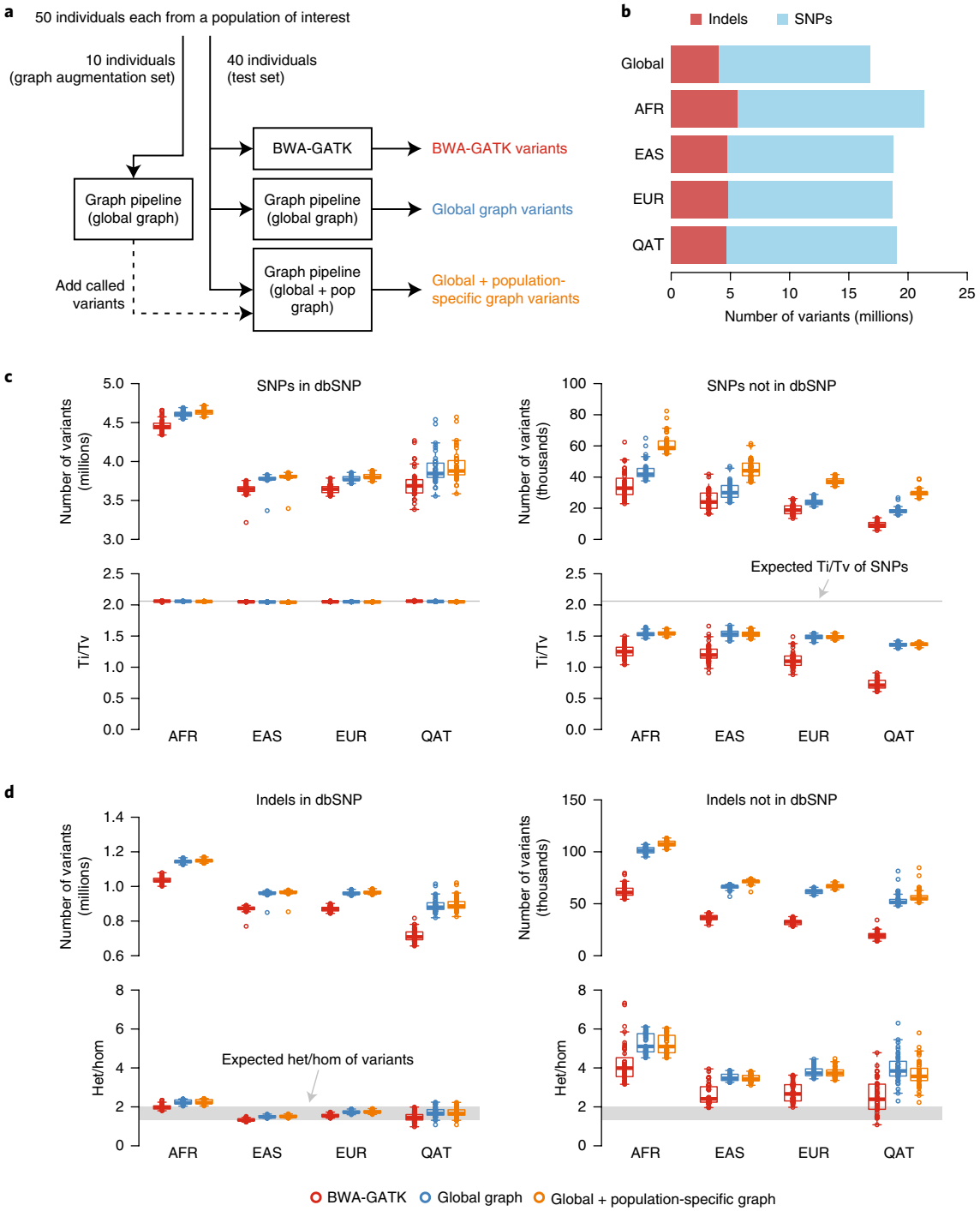

Fig. 5 | The effect of iteratively augmented graph genomes on variant calling. a, Schematic representation of the graphs generated in the graph augmentation experiment. **b**, Numbers of SNPs, indels and other variant types in the global and augmented graphs. **c**, Counts and Ti/Tv of known and novel SNPs called through the BWA-GATK, global graph and augmented graph pipelines ($n = 40$ Coriell WGS samples for each pipeline as described in **a**). Gray horizontal lines indicate the expected Ti/Tv genome-wide ratio. In box plots, center line, box edges and whiskers indicate the median, upper and lower quartiles and 1.5× interquartile range, respectively. **d**, Counts and het/hom ratios of known and novel indels called through the BWA-GATK, global graph and population-augmented graph pipelines ($n = 40$ Coriell WGS samples for each pipeline as described in **a**). Horizontal bars indicate the average value of each group. Gray rectangles indicate the expected range of population averages for het/hom alternate ratios. Box plot elements are defined as in **c**.

whole human genomes. Individual gene families or related microbial strains could be compressed and efficiently searched using our graph genome algorithms. Similarly, the transcriptome could be represented as genomic deletions, allowing RNA-seq reads to be directly aligned across exon–exon junctions[21]. A personalized graph genome could be constructed from a sequenced germline genome, in order to provide an optimized scaffold for somatic variant detection in matched cancer genomes.

As more genetic variants are accumulated on a reference graph genome, cumulative accuracy gains in genomics analysis can be achieved. This is consistent with recent efforts to establish population-specific reference panels, which have been shown to contribute

to increased accuracy in imputation[33–35] and genetic risk prediction[36]. The current wave of national sequencing projects will extend the catalogs of population-specific genetic variants, which will incrementally improve the prospects for graph genome reference approaches[16,20,21], including ours. Completion of the first draft of the human reference genome marked the beginning of human genomics. Our computationally efficient and flexible graph genome implementation supports the community for a gradual transition toward a graph-based reference system.

Online content

Any methods, additional references, Nature Research reporting summaries, source data, statements of data availability and associated accession codes are available at https://doi.org/10.1038/s41588-018-0316-4.

Received: 2 October 2017; Accepted: 14 November 2018;

References

1. Lander, E. S. et al. Initial sequencing and analysis of the human genome. *Nature* **409**, 860–921 (2001).
2. Venter, J. C. et al. The sequence of the human genome. *Science* **291**, 1304–1351 (2001).
3. 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).
4. 1000 Genomes Project Consortium. et al. 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. Degner, J. F. et al. Effect of read-mapping biases on detecting allele-specific expression from RNA-sequencing data. *Bioinformatics* **25**, 3207–3212 (2009).
7. Brandt, D. Y. C. et al. Mapping bias overestimates reference allele frequencies at the HLA genes in the 1000 Genomes Project Phase I data. *G3* **5**, 931–941 (2015).
8. Alkan, C., Coe, B. P. & Eichler, E. E. Genome structural variation discovery and genotyping. *Nat. Rev. Genet.* **12**, 363–376 (2011).
9. Antaki, D., Brandler, W. M. & Sebat, J. SV2: accurate structural variation genotyping and de novo mutation detection. *Bioinformatics* **34**, 1774–1777 (2018).
10. Maretty, L. et al. Sequencing and de novo assembly of 150 genomes from Denmark as a population reference. *Nature* **548**, 87–91 (2017).
11. Mallick, S. et al. The Simons Genome Diversity Project: 300 genomes from 142 diverse populations. *Nature* **538**, 201–206 (2016).
12. Lek, M. et al. Analysis of protein-coding genetic variation in 60,706 humans. *Nature* **536**, 285–291 (2016).
13. Schneeberger, K. et al. Simultaneous alignment of short reads against multiple genomes. *Genome Biol.* **10**, R98 (2009).
14. Paten, B., Novak, A. M., Eizenga, J. M. & Garrison, E. Genome graphs and the evolution of genome inference. *Genome Res.* **27**, 665–676 (2017).
15. Paten, B., Novak, A. & Haussler, D. Mapping to a reference genome structure. *arXiv [q-bio.GN]* 1404.5010 (2014).
16. Novak, A. M. et al. Genome graphs. *bioRxiv* https://doi.org/10.1101/101378(2017).
17. Huang, L., Popic, V. & Batzoglou, S. Short read alignment with populations of genomes. *Bioinformatics* **29**, i361–i370 (2013).
18. 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).
19. Eggertsson, H. P. et al. Graphtyper enables population-scale genotyping using pangenome graphs. *Nat. Genet.* **49**, 1654–1660 (2017).
20. Garrison, E. et al. Variation graph toolkit improves read mapping by representing genetic variation in the reference. *Nat. Biotechnol.* **36**, 875–879 (2018).
21. Kim, D., Langmead, B. & Salzberg, S. L. HISAT: a fast spliced aligner with low memory requirements. *Nat. Methods* **12**, 357–360 (2015).
22. Sirén, J., Garrison, E., Novak, A. M., Paten, B. & Durbin, R. Haplotype-aware graph indexes. *arXiv [cs.DS]* 1805.03834 (2018).
23. DePristo, M. A. et al. A framework for variation discovery and genotyping using next-generation DNA sequencing data. *Nat. Genet.* **43**, 491–498 (2011).
24. Li, H. Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM. *arXiv [q-bio.GN]* 1303.3997v2 (2013).
25. Zook, J. M. et al. Extensive sequencing of seven human genomes to characterize benchmark reference materials. *Sci. Data* **3**, 160025 (2016).
26. Michailidou, K. et al. Large-scale genotyping identifies 41 new loci associated with breast cancer risk. *Nat. Genet.* **45**, 353–361 (2013).
27. Berndt, S. I. et al. Genome-wide meta-analysis identifies 11 new loci for anthropometric traits and provides insights into genetic architecture. *Nat. Genet.* **45**, 501–512 (2013).
28. McVey, M. & Lee, S. E. MMEJ repair of double-strand breaks (director's cut): deleted sequences and alternative endings. *Trends Genet.* **24**, 529–538 (2008).
29. Wang, J., Raskin, L., Samuels, D. C., Shyr, Y. & Guo, Y. Genome measures used for quality control are dependent on gene function and ancestry. *Bioinformatics* **31**, 318–323 (2015).
30. Fakhro, K. A. et al. The Qatar genome: a population-specific tool for precision medicine in the Middle East. *Hum. Genome Var.* **3**, 16016 (2016).
31. Nho, K. et al. Comparison of multi-sample variant calling methods for whole genome sequencing. *IEEE Int. Conf. Systems Biol.* **2014**, 59–62 (2014).
32. Novak, A. M., Garrison, E. & Paten, B. A graph extension of the positional Burrows-Wheeler transform and its applications. *Algorithms Mol. Biol.* **12**, 18 (2017).
33. Huang, J. et al. Improved imputation of low-frequency and rare variants using the UK10K haplotype reference panel. *Nat. Commun.* **6**, 8111 (2015).
34. van Leeuwen, E. M. et al. Genome of The Netherlands population-specific imputations identify an ABCA6 variant associated with cholesterol levels. *Nat. Commun.* **6**, 6065 (2015).
35. Nagasaki, M. et al. Rare variant discovery by deep whole-genome sequencing of 1,070 Japanese individuals. *Nat. Commun.* **6**, 8018 (2015).
36. Martin, A. R. et al. Human demographic history impacts genetic risk prediction across diverse populations. *Am. J. Hum. Genet.* **100**, 635–649 (2017).

Acknowledgements

We are grateful for the members of the GA4GH Data Workgroup, Benchmarking, and Reference variation initiatives, in particular J. Zook, for insightful discussions and ideas. M. Huvet helped refine the treatment and presentation of ideas behind trio-based benchmarking. Research reported in this publication was supported in part by the UK Department of Health grant SBRI Genomics Competition: Enabling Technologies for Genomic Sequence Data Analysis and Interpretation administered by Genomics England.

Author contributions

G.R., V.S., W.-P.L., J.S., A.D., B.P., A.J., and I.S. developed the algorithms and implemented the tools for graph genome alignment. J.B. and I.J.J. developed the algorithms and implemented the tools for variant calling. K.G. implemented the simulation experiments and carried out the benchmarks based on simulated data. V.A., J.N., A.J., and G.R. devised and carried out the experiments with population-specific genome graphs. P.K. and B.C.T. developed the computational tools used for benchmarks based on related genomes, and A.J. and M.C.S. carried out the experiments. S.-G.J., G.D., L.L., and P.K. created the genome graph containing the structural variants, designed, and carried out all of related experiments. M.P. created the machine learning–based variant filters and carried out the related experiments. I.G. and M.K. aided in interpreting the results and worked on the manuscript. Y.L., G.R., and D.K. prepared the manuscript with input from all other authors. D.K. conceived and oversaw the project with assistance from A.J., A.L.S., and M.K.

Competing interests

G.R., J.S., V.A., J.N., M.C.S., G.D., L.L., B.C.T., B.P., I.S., I.G., P.K., A.L.S., Y.L., M.P., W.-P.L., M.K., and D.K. were employed by Seven Bridges Genomics Inc. during the development of the described tools. V.S., J.B., I.J.J., K.G., S.-G.J., A.D., and A.J. are current employees of Seven Bridges Genomics Inc. G.R., V.S., J.S., J.B., I.J.J., V.A., K.G., S.-G.J., L.L., I.S., P.K., A.L.S., Y.L., A.J., M.P. and D.K. hold shares, stock options or restricted stock units in Seven Bridges Genomics Inc. D.K. is co-inventor on 12 patents (issued: 14/016,833; 14/811,057; 15/196,345; 14/041,850 14/157,759; 14/157,979; published: 14/517,406; 14/517,419; 14/517,513; 14/517,451; 14/744,536; 14/798,686). V.S. is inventor on four patents (pending: 15/061,235; 14/885,192; 15/598,404; 15/597,464). W.-P.L. is co-inventor on three patents (published: 14/994,385, pending: 15/353,105; 15/007874). B.P., I.S. and A.J. are co-inventors on one patent (pending: 15/452,963). I.J.J is inventor on one patent (62/630,347). Applicant for patents is Seven Bridges Genomics Inc.

Additional information

Supplementary information is available for this paper at https://doi.org/10.1038/s41588-018-0316-4.

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

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

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

Methods

Genome graphs. All analyses are based on the human reference genome version GRCh37 (ref. [37]). Variants for the Global Graph Reference (Figs. 1–4) were obtained by combining common SNPs and indels from 1000 Genomes Phase 3 (ref. [4]), Simons Genome Diversity Project[11], the Mills indels database[38], 1000 Genomes Phase 3 indels[39], and a curated set of 704 long deletions from 1000 Genomes Phase 3. Variants were normalized using Bcftools norm (https://samtools.github.io/bcftools/bcftools.html) prior to being merged into a single VCF used to build the Global Graph Reference.

Pedigree Graph References (Fig. 3d,e) were generated by adding parents' variants to the Global Graph Reference. Each Pedigree Graph Reference's parents' variants were generated using the respective variant calling pipeline; Graphtyper was used to generate parents' variants for the Pedigree Graph Reference used in the Graphtyper experiments.

Population-augmented graph references for each population (Fig. 5) were generated by combining the Global Graph Reference variants with variants called from a randomly selected set of ten individuals from the same population. For these individuals, initial variant calling was performed using the Graph Genome Pipeline and the Global Graph Reference. Individuals used for augmenting the Global Graph Reference were distinct from the individuals used to produce benchmarking metrics in Fig. 5 (Supplementary Table 16).

Graph structure. We use adjacency lists to store the set of connected vertices for each vertex in the graph. Each vertex stores references to the set of incoming and outgoing edges. The sequence data are stored separately in a buffer; edges need only contain an offset address in the buffer and length, thus inserting a new edge (which involves splitting an existing edge) is a fast operation.

The graph is efficiently serialized by handling the sequence data and graph structure separately. The sequence data are compressed in blocks using an n-bit encoding, where n is determined according to the alphabet size of the data in the block; $n=2$ for a sequence containing {A,C,T,G}. The structure of the graph is serialized by storing a reference to the sequence represented by the i^{th} edge along with the start and end loci at which the edge was inserted into the graph containing $i-1$ edges; in other words, the set of graph manipulations are stored in the same order they were used to generate the graph. The index is not serialized.

Given these approaches, the graph containing the complete 1000G variant set occupies 23.8 GB of RAM, comprising of 13.3 GB for the graph structure, 3 GB for the sequence data, 7.5 GB for the index and (when serialized) 1.1 GB of disk space, of which 63% is used for sequence data and the remainder for the graph structure.

Read alignment. A hash-based search index is used for efficiently placing reads into a graph reference. A hash value is calculated for nucleotide sequences of length k (k-mer) in the graph reference. Each search index entry contains a list of start positions of k-mers (k-mer loci) that have the same hash value. The k-mers are determined by sequentially traversing the graph and indexing sequences of length k starting at every s^{th} position, where s is the indexing step and $1 \le s \le k$ (Supplementary Fig. 1). Variant edges are similarly indexed at every s^{th} position until the k-mer would start on the parent edge. For the purposes of indexing, the position on an alternate branch is counted starting from the beginning of the graph and taking the reference path in all branching points preceding the variant branch(es) of interest (Supplementary Fig. 1). A k-mer is limited to follow up to 16 edges in the graph, where the limit of 16 follows from the observation that such regions of the graph act as k-mer attractors (sites where excessively large numbers of k-mers match). Entries in the index that contain more than a certain number of loci (hash list size threshold) are removed to prioritize seeding based on informative k-mers (k-mers containing less common nucleotide sequences), thus speeding up the search and reducing memory requirement. The results presented in this study were calculated for $k=21$ and $s=7$. This combination of settings was found to deliver a good trade-off between index size and performance.

The search procedure is implemented as follows. For each k consecutive symbols in a read, a hash index is calculated by applying the hash function used during the search index construction; a list of k-mer loci corresponding to the calculated hash index is determined. We use a sliding search window approach to locate substantial spatial clusters of loci belonging to an aggregate of k-mer lists determined for the read. These clusters represent candidate match regions. We allow for gaps between loci. A search window size is selected based on the read length and the upper limit for novel indels. Each located cluster is assigned a score, which is calculated by analyzing matching k-mers, their positions in the graph and corresponding positions in the read. The higher the score, the larger the probability of match with the read. Clusters with scores exceeding a threshold are treated as seeds for local alignment. Each seed thus represents a region in the graph against which the read might align. Paired-end reads are treated as single search patterns with gaps. These reads are processed in the same way as single-end reads, which significantly reduces the computational complexity of paired-read search.

The majority of reads do not contain novel insertions or deletions. We take advantage of this fact by employing a hierarchical approach to local alignment: for each candidate seed, we first attempt a fast gapless alignment algorithm and then, if required, a slower alignment algorithm that permits novel insertions and deletions. In both cases, the graph region is extended on either side of the seed to accommodate cases where k-mers at each end of the read are not contained in the graph index or the read contains novel variations or comes from a repetitive region. We use a graph-aware version of the bit-parallel approximate (BPA) string-matching algorithm (also known as the bitap algorithm) for the first stage. Reads in that presence of novel indels and structural variations not contained in the graph are mapped using a custom SIMD-optimized implementation of the Smith–Waterman algorithm against the graph reference[40] in cases where BPA fails to find an alignment with fewer than a given number of mismatches for a seed (default: 4). For BAM output, each read aligned to the graph genome is projected to and given a CIGAR string against the linear reference genome. Reads fully aligned within an insertion are placed to the linear reference nucleotide just before the insertion. Mapping qualities for single-end reads are computed by closely following the approach taken in MAQ[41]. Mapping qualities for paired-end reads are computed by summing the individual single-end mapping qualities for each mate. Further details are presented in the Supplementary Note (section 3.1).

Aligner benchmarking was run on Amazon AWS c4.8xlarge instance with 36 CPU cores and 60 GB of RAM.

Variant calling. The Reassembling Variant Caller of the Graph Genome Pipeline takes elements from Samtools[41], FreeBayes[42] and HaplotypeCaller[43]. An important novel aspect of our approach is the use of the graph reference to help guide the assembly and genotyping phases (Supplementary Note, section 3.2.5), reducing the effects of reference bias. We combine regions with candidate variants based on read CIGARs into reassembly windows of roughly 300 bp. For each reassembly window, overlapping reads are compressed into a De Bruijn-like graph[43]. Non-unique k-mers are flagged and treated separately to avoid loops in the De Bruijn-like graph (Supplementary Note, section 3.2). Candidate haplotypes are derived from the De Bruijn-like graph using depth-first search by prioritizing variants with the highest degree of read support and scored using the pair Hidden Markov Model[43,44]. SNPs, small indels and large SVs were all assembled and called using the same variant calling algorithm. The graph reference genome factors into variant calling in that: (1) k-mers present in the graph are prioritized during a k-mer filtering preprocessing step; (2) variants in the graph are given a higher prior probability.

For the benchmarking experiments in the main text, we used the standard Graph Genome Pipeline that employs a set of previously proposed hard variant filters[45] to filter raw variant calls. For the PrecisionFDA Truth Challenge benchmarking experiment only (Supplementary Fig. 14), we used a logistic regression model to filter variants based on the GATK best practices[45] and other features extracted from the raw variant calls (Supplementary Note, section 3.4).

Benchmarking experiments. We simulated reads for ten genomes using the open-source Mitty program (https://github.com/sbg/Mitty). As truth VCFs, we used five samples from the GiaB project (HG001, HG002, HG003, HG004, HG005) and five samples from the 1000G Phase 3 release (HG00096, HG00551, HG03585, NA12878 and NA18488). We generated 50× coverage reads from the whole genome, excluding regions with masked reference sequence (i.e., those with 'N's).

Benchmarking on real data (Fig. 3b) was done on all five GiaB[25,46] samples: HG001 (NA12878), HG002 (son of the Ashkenazim Jewish Trio), HG003 (father of the Ashkenazim Jewish Trio), HG004 (mother of the Ashkenazim Jewish Trio), and HG005 (son of the Chinese Trio). HG001–HG004 were sequenced to 50× coverage using PCR-free library preparation protocol and 2×150 sequencing reads, and HG005 was sequenced using 2×250 bp reads. All files were downloaded from the GiaB FTP site (ftp://ftp-trace.ncbi.nih.gov/giab/ftp/).

We used two sets of trios in the experiments based on the related genomes: the CEU Trio (NA12878 (daughter), NA12891 (father), and NA12892 (mother)), and the AJ Trio (HG002 (son), HG003 (father), and HG004 (mother)). Data for the CEU trio are 40× coverage of 100-bp paired-end reads (available from 1000 Genomes FTP, http://ftp.1000genomes.ebi.ac.uk/), and the AJ Trio is based on 50× coverage of 150-bp paired-end read data from the GiaB FTP site (ftp://ftp-trace.ncbi.nih.gov/giab/ftp/).

Variant sets called by each pipeline were evaluated against their respective truth sets using vcfeval[47] and hap.py (https://github.com/Illumina/hap.py). For trio benchmarking, we developed a variant comparison tool (https://github.com/sbg/VBT-TrioAnalysis) that generalizes the idea of vcfeval to family trios (Supplementary Note, section 4.6.1). This tool computes the numbers of different types of Mendelian compliant or inconsistent variants in the entire trio. From the computed Mendelian compliant and inconsistent variant counts, we developed an expectation maximization approach to estimate the respective underlying precision and recall metrics (Supplementary Note, section 4.6.2). Like small variants, SVs can have multiple representations in a VCF, but a brute force–based variant comparison approach like vcfeval[47] is computationally prohibitive for large SVs. We therefore developed an SV comparison tool tolerant to different VCF-compatible SV representations (Supplementary Note, section 6.2).

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

Code availability

Graph Genome Pipeline is freely available to academic users for non-commercial use. Compiled standalone tools and the License of Use can be accessed at https://www.sevenbridges.com/graph-genome-academic-release/. The source code of the Graph Genome Pipeline tools is not publicly available.

Data availability

Raw sequencing data for the 150 Coriell WGS samples (Figs. 1, 4 and 5) can be accessed from the European Nucleotide Archive under accession PRJEB20654. Raw sequencing data for the Qatari samples (Fig. 5) used can be found under NCBI SRA accessions SRP060765, SRP061943 and SRP061463. Genome in a Bottle data (Fig. 3) are available from the NCBI FTP site (ftp://ftp-trace.ncbi.nlm.nih.gov/giab/ftp/data). The Sanger sequencing traces have been deposited in the European Nucleotide Archive under accession PRJEB26700.

References

37. Church, D. M. et al. Modernizing Reference Genome Assemblies. *PLoS Biol.* **9**, e1001091 (2011).
38. Mills, R. E. et al. An initial map of insertion and deletion (INDEL) variation in the human genome. *Genome Res.* **16**, 1182–1190 (2006).
39. 1000 Genomes Project Consortium. et al. An integrated map of genetic variation from 1,092 human genomes. *Nature* **491**, 56–65 (2012).
40. Kural, D. *Methods for Inter- and Intra-species Genomics for the Detection of Variation and Function*. (Boston College Graduate School of Arts and Sciences, Boston, 2014).
41. Li, H. & Durbin, R. Fast and accurate short read alignment with Burrows-Wheeler transform. *Bioinformatics* **25**, 1754–1760 (2009).
42. Garrison, E. & Marth, G. Haplotype-based variant detection from short-read sequencing. *arXiv [q-bio.GN]* 1207.3907 (2012).
43. Poplin, R. et al. Scaling accurate genetic variant discovery to tens of thousands of samples. *bioRxiv* https://doi.org/10.1101/201178 (2017).
44. Durbin, R., Eddy, S. R., Krogh, A. & Mitchison, G. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. (Cambridge University Press, 1998).
45. Van der Auwera, G. A. et al. From FastQ data to high confidence variant calls: the Genome Analysis Toolkit best practices pipeline. *Curr. Protoc. Bioinformatics.* **43**, 11.10.1–33 (2013).
46. 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).
47. Cleary, J. G. et al. Comparing variant call files for performance benchmarking of next-generation sequencing variant calling pipelines. *bioRxiv* https://doi.org/10.1101/023754 (2015).

nature research

Corresponding author(s): Deniz Kural

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 code or software was used in data collection.
Data analysis	Graph genome toolkit is available for use on the Seven Bridges Cloud Platform (https://www.sevenbridges.com). Tool versions used are: SBG Graph aligner 0.9.11, SBG Reassembly Variant Caller 0.5.20 We also used the following software tools: GATK 3.7 (including HaplotypeCaller) https://github.com/broadgsa/gatk vcfeval 3.7.0 https://github.com/RealTimeGenomics/rtg-tools hap.py 0.3.5 https://github.com/Illumina/hap.py bcftools 1.3 https://samtools.github.io/bcftools/ Freebayes 1.10 https://github.com/ekg/freebayes Graphtyper 1.3 https://github.com/DecodeGenetics/graphtyper Bayestyper 1.1 https://github.com/bioinformatics-centre/BayesTyper Delly2 0.7.7 https://github.com/dellytools/delly

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

Data availability: Raw sequencing data for the 150 Coriell WGS samples (Figs. 1, 4 and 5) can be accessed European Nucleotide Archive, Study Accession PRJEB20654, (https://www.ebi.ac.uk/ena/data/view/PRJEB20654). Raw sequencing data for the Qatari samples (Fig. 5) used can be found in the NCBI SRA accessions SRP060765, SRP061943 and SRP061463 (https://www.ncbi.nlm.nih.gov/Traces/study/?acc=SRP060765%2CSRP061943%2CSRP061463&go=go). Genome in a Bottle data (Fig. 3) is available from the NCBI FTP site: ftp://ftp-trace.ncbi.nlm.nih.gov/giab/ftp/data. The Sanger sequencing traces will be deposited to European Nucleotide Archive (Accession ID PRJEB26700).

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 ☐ Ecological, evolutionary & environmental 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 sample size selection was done.
Data exclusions	No data were excluded.
Replication	We ran three different classes f benchmarks using truth sets, simulations and population genetics measures, all of which support the conclusions. The described computational methods are deterministic and the results are reproducible.
Randomization	We did not take part in selecting participants for any of the cohorts used.
Blinding	We did not take part in selecting participants for any of the cohorts used.

Reporting for specific materials, systems and methods

Materials & experimental systems

n/a	Involved in the study
☒	☐ Unique biological materials
☒	☐ Antibodies
☐	☒ Eukaryotic cell lines
☒	☐ Palaeontology
☒	☐ Animals and other organisms
☒	☐ Human research participants

Methods

n/a	Involved in the study
☒	☐ ChIP-seq
☒	☐ Flow cytometry
☒	☐ MRI-based neuroimaging

Eukaryotic cell lines

Policy information about cell lines

Cell line source(s)	NA12878 and HG002 sample DNA used for validation of calls was obtained from NIST: https://www-s.nist.gov/srmors/view_detail.cfm?srm=8398 and https://www-s.nist.gov/srmors/view_detail.cfm?srm=8391
Authentication	DNA was authenticated by sequencing of 3 private SNPs identified by comparing the GiaB High Confidence Truth Set data to set of calls from 1000 Genomes phase 3 project.
Mycoplasma contamination	We did not test for mycoplasma contamination.

Commonly misidentified lines (See ICLAC register)	We did not use any commonly misidentified cell lines