Author version of *Denti, L., Khorsand, P., Bonizzoni, P. et al. SVDSS: structural variation discovery in hard-to-call genomic regions using sample-specific strings from accurate long reads. Nature Methods (2022)*
DOI: 10.1038/s41592-022-01674-1
Article URL: https://www.nature.com/articles/s41592-022-01674-1

SVDSS: Structural Variation Discovery in hard-to-call genomic regions using Sample-specific Strings from accurate long-reads

Luca Denti[1,*], Parsoa Khorsand[2,*], Paola Bonizzoni[3,†,✉], Fereydoun Hormozdiari[2,4,5,†,✉], and Rayan Chikhi[1,†,✉]

[1]Sequence Bioinformatics, Department of Computational Biology, Institut Pasteur, F-75015 Paris, France
[2]Genome Center, UC Davis, Davis, CA, USA
[3]Department of Informatics, Systems and Communication, University of Milano-Bicocca, Viale Sarca 336, 20126, Milan, Italy
[4]UC Davis MIND Institute, Sacramento, CA, USA
[5]Department of Biochemistry and Molecular Medicine, Sacramento, UC Davis, Sacramento, CA, USA
[*]These authors contributed equally
[†]These authors jointly supervised this work
[✉]Corresponding authors: paola.bonizzoni@unimib.it, fhormozd@ucdavis.edu, rayan.chikhi@pasteur.fr

Abstract

Structural variants (SVs) account for a large amount of sequence variability across genomes and play an important role in human genomics and precision medicine. Despite intense efforts over the years, the discovery of SVs in individuals remains challenging due to the diploid and highly repetitive structure of the human genome, and by the presence of SVs that vastly exceed sequencing read lengths. However, the recent introduction of low-error long-read sequencing technologies such as PacBio HiFi may finally enable to overcome these barriers. Here we present `SVDSS`, a method for discovery of SVs from long-read sequencing technologies (e.g., PacBio HiFi) that combines and effectively leverages mapping-free, mapping-based and assembly-based methodologies for overall superior SV discovery performance. Our experiments on several human samples show that `SVDSS` outperforms state-of-the-art mapping-based methods for discovery of insertion and deletion SVs in PacBio HiFi reads and achieves significant improvements in calling SVs in repetitive regions of the genome.

1 Introduction

Structural variants (SVs) are defined as medium to large-size genomic rearrangements [1, 2]. SVs can range from tens of base-pairs to over megabases of sequence. The different types of SVs include balanced SVs, such as inversions and translocations, and unbalanced SVs, such as insertions and deletions [3]. The study and characterization of SVs has been driven by constant improvements in the available technologies to assay variants. Although structural variants are not the most ubiquitous type of genetic variants, the total volume of base-pairs impacted by SVs is far more than any other type of genetic variant, including Single Nucleotide Variants (SNVs) [4, 5]. Furthermore, recent studies of structural variants using orthogonal technologies has shown that SVs are the least well-characterized type of genetic variants with many basic questions still not

completely resolved, such as the average number of SVs per sample or sequence biases contributing to their formation [6, 7, 8, 9]. In addition, the homology-driven mechanisms behind SV formation (e.g., non-allelic homologous recombination) has contributed to the complexity of their systematic study [10]. It is believed that a large fraction of polymorphic SVs are still not fully characterized [11, 12].

As our current understanding of SVs evolves, it is becoming clear that SVs are a major contributing factor to human diseases [13, 14, 15], population genomics [5, 16] and evolution [17]. The comparative study of SVs in multiple closely-related species (e.g., great apes) has shown significant contribution of SVs to evolution (e.g., through gene duplication or deletion [18, 19]). Furthermore, study of rare and *de novo* SVs in disease such as autism and epilepsy has proven the significant contribution of these variants in such diseases [20, 21, 22, 23, 15]. It is also known that somatic SVs are one of the major causative variants in different types of cancer [24, 25, 26, 27].

With the advent of short-read and long-read high-throughput sequencing technologies in the past decade, significant progress has been made in our understanding of the abundance, complexity, and importance of SVs [28, 29, 30, 31]. There are many methods developed for prediction of SVs using whole-genome sequencing (WGS) data produced from different sequencing technologies [32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42]. Majority of these methods try to predict variants by detecting certain SV signatures (i.e., read-depth, read-pair, or split-read) in mappings of the reads to the reference genome [43, 44, 6] and are hence known as "mapping-based" methods. Mapping-based methods have contributed significantly to our understanding of the abundance of SVs in the general population and their role in diseases [11, 45, 46]. Mapping-free methods are a more recent group of approaches that try to predict SVs without mapping the reads to the reference genome and instead by comparing sequence data between different genomes [47, 34]. Finally, assembly-based approaches first assemble the sequenced reads into longer contigs and use the assembled contigs to predict variants [48, 49, 50]. Assembly-based methods have recently been shown to provide superior performance to mapping-based tools [51].

There are limiting factors for predicting SVs using each of these frameworks. Since a majority of SV prediction tools use mappings of the reads to the reference genome for making SV calls, predicting SVs in highly repeated regions of the genome (e.g., segmental duplications) where mappings can be inaccurate would be prone to false discovery. Reference genome gaps and misassemblies further complicate the prediction of SVs in these regions and result in decreased accuracy and increased variability across tools [8]. The mapping-free approaches on the other hand suffer from not being able to provide the loci of the event. Furthermore, fixed-length (k-mer) sequence comparisons performed in mapping-free tools can result in collapse of repeats and lower sensitivity/accuracy. Finally, assembly-based approaches are very computationally resource intensive and often require integration of data from multiple different technologies (i.e., long-reads, short-reads, and Hi-C) [52, 51], higher sequencing depths (35x was reported in [51]), and extensive polishing and post-processing to yield a high-quality *de novo* assembly suitable for variant prediction, thus making them impractical for SV discovery across large populations.

Here, we propose a method called `SVDSS` that combines advantages of all three mapping-based, mapping-free, and assembly-based approaches for predicting SVs. Our method utilizes mapping-free sample-specific signatures [53] along with mapping information to cluster reads potentially including SVs and then performs local assembly and alignment of the clusters for SV prediction. With the combination of different analysis methods, our algorithm is able to improve SV calling performance particularly in repetitive areas of the genome compared to other contemporary approaches.

2 Results

Overview of `SVDSS`. We present `SVDSS` (Structural Variant Discovery with Sample-specific Strings), a method for the discovery of structural variants from accurate long reads (e.g., PacBio HiFi). `SVDSS` takes as input a reference genome and a mapped BAM file and produces SV calls in VCF format along with assembled contigs for SV sites in SAM format. We use the concept of sample-specific strings (SFS) which we previously introduced as all the shortest substrings unique to one string set with regards to another string set [53]. We employ SFS here to pinpoint differences between reads and a reference genome [53]. Our method

computes SFS for coarse-grained identification of potential SV sites. It assembles clusters of SFS from such sites to produce contigs that are then locally aligned to the reference genome to detect SVs. The main advantage of using SFS is that they are not limited to fixed length seeds (unlike k-mers) and the algorithm can dynamically find the shortest string for covering the breakpoints of each variant, thus, making SFS ideal for anchoring potential SV breakpoints.

`SVDSS` has three main steps as depicted in Figure 1, sketched here and explained in more details in the Online Methods:

1. **Read smoothing:** reads are modified to remove sequencing errors, SNPs and small indels (< 20bp) that may interfere with SV calling (Figure 1 step 1, and Supplementary Figure S3). Smoothing significantly reduces the number of extracted SFS while increasing their specificity for the purpose of SV calling.

2. **SFS superstring construction:** SFS are computed from the smoothed reads using the optimal Ping-Pong algorithm [53] (Figure 1, 2A) and then assembled into superstrings to reduce redundancy (2B).

3. **SV prediction using SFS superstrings:** SFS superstrings are clustered based on position and extended to include unique anchoring sequences from the reference genome (Figure 1, 3A), further subclustered by length then assembled based on POA approach to generate haplotype candidates (3B). Finally SVs are called by aligning the resulting POA consensus(es) (3C).

In the following sections, using experimental analysis on multiple WGS samples, we demonstrate that `SVDSS` accurately predicts SVs and outperforms state-of-the-art approaches. We further show that the main contribution of our proposed approach is the ability to more accurately predict SVs falling in repeated regions of the genome compared to other methods.

Benchmark and evaluation callsets. One complexity in comparing different tools for calling SVs is the imperfectness of available callsets. Missing variants and potentially false predictions affect almost all published callsets, and even the most high-quality callsets have been reported to have a $\sim 5\%$ false discovery rate and a much higher false negative rate [6]. Furthermore, many callsets are constructed using state-of-the-art but imperfect SV prediction tools and are thus biased towards these methods [54]. For these reasons, we have opted out of using pre-existing callsets such as the 2020 Genome In A Bottle (GIAB) v0.6 callset [45] in our experimental benchmarking. Instead we constructed our ground truth SV callsets from scratch using high-quality haplotype-resolved *de novo* assemblies generated by utilizing many technologies (T2T CHM13 v1.1, HG002, and HG007, described in the next paragraph). A similar ground truth construction strategy was employed in a 2022 GIAB benchmark [51], although focusing on a subset of medically relevant genes. We applied the assembly-to-assembly SV calling tool `dipcall` [54] to each assembly versus the entire GRCh38 reference genome (see Supplementary Section A for more details). The three VCFs built using `dipcall` and used as ground truth in our experimental evaluation are available at `https://github.com/ldenti/SVDSS-experiments`. For a detailed comparison of the HG002 callset built with `dipcall` and the v0.6 callset provided by the GIAB project, we refer the reader to Supplementary Section B.

Comprehensive detection of insertions and deletions. We experimentally validated the accuracy of the `SVDSS` pipeline in calling SVs from three whole-genome sequenced samples sequenced using PacBio HiFi technology: the homozygous CHM13 sample from the telomore-to-telomere (T2T) project [52] and the HG002 and HG007 samples corrected using `DeepConsensus` [55]. These samples were chosen because of the availability of high-quality and effectively complete assemblies for them. Furthermore, the DeepConsensus corrected HG002 and HG007 samples show higher accuracy than standard HiFi samples corrected using only `pbccs` [55]. The use of both homozygous (CHM13) and heterozygous (HG002 and HG007) samples allows for more comprehensive analysis and comparison of SV calling methods.

We mapped each sample against the reference genome using `pbmm2` and then we called SVs on each sample using the `SVDSS` pipeline. We compared our approach to five state-of-the-art mapping-based SV callers: `pbsv`,

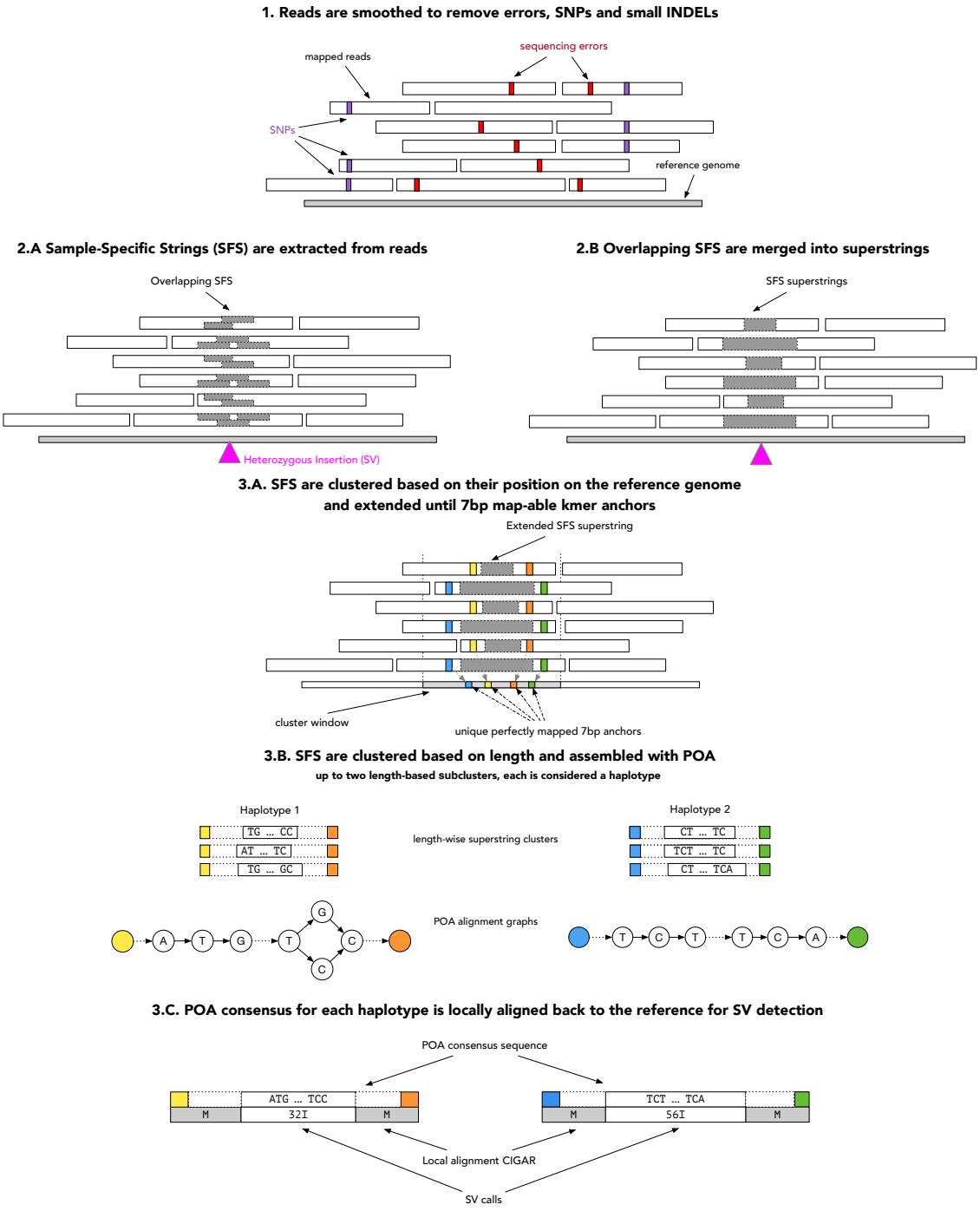


Figure 1: **Overview of the SVDSS SV prediction pipeline.** (1) Reads are smoothed to remove SNPs and sequencing errors. (2.A) SFS are extracted from reads and (2.B) assembled into superstrings. (3.A) Superstrings (grey) are clustered based on their placements on the reference genome and extended until uniquely mappable 7bp anchors on each side (colored). (3.B) Each cluster is further clustered into up to two subclusters based on length of the superstring. Each subcluster signifies a potential haplotype. The subclusters are assembled with POA to generate a consensus sequence. (3.C) The POA consensus for each cluster is locally aligned to the reference genome and SVs are called from the mapping information.

Region	Tool	HG002			HG007			CHM13		
		P	R	F1	P	R	F1	P	R	F1
Full Genome	SVDSS	88.4	**78.2**	**83.0**	**90.1**	**76.5**	**82.7**	87.3	**84.6**	**86.0**
	cuteSV	86.0	68.6	76.3	88.3	68.1	76.9	87.1	79.7	83.2
	pbsv	86.9	68.8	76.8	84.9	68.6	75.9	84.6	82.7	83.6
	sniffles	82.0	67.3	73.9	86.7	64.1	73.7	86.4	81.4	83.8
	SVIM	83.5	65.1	73.2	84.9	64.7	73.4	**90.1**	79.9	84.7
	debreak	**88.6**	67.5	76.6	**90.1**	64.2	75.0	83.7	79.6	81.6
Tier 1	SVDSS	95.2	**85.5**	**90.1**	95.2	**82.7**	**88.5**	95.3	93.4	**94.5**
	cuteSV	90.9	82.9	86.7	93.0	79.9	86.0	94.8	93.1	93.9
	pbsv	95.7	83.1	89.0	89.7	80.5	84.9	94.0	93.7	93.9
	sniffles	87.7	81.1	84.3	92.3	75.9	83.3	87.2	93.6	90.3
	SVIM	90.1	81.1	85.4	91.5	77.9	84.2	**96.6**	92.5	**94.5**
	debreak	**96.8**	82.5	89.1	**96.2**	76.4	85.2	93.7	93.0	93.3
Extended Tier 2	SVDSS	**82.7**	**72.3**	**77.2**	**84.6**	**70.2**	**76.7**	80.3	**77.4**	**78.8**
	cuteSV	80.9	57.0	66.9	82.3	56.0	66.6	79.9	68.1	73.6
	pbsv	78.4	57.2	66.1	78.8	56.4	65.7	76.0	73.3	74.6
	sniffles	77.8	56.1	65.2	80.3	52.1	63.2	72.7	73.2	72.9
	SVIM	76.4	52.0	61.9	76.2	51.2	61.2	**83.4**	69.3	75.7
	debreak	80.4	55.3	65.5	82.3	51.9	63.7	74.4	68.1	71.1

Table 1: **Comparison of performance of SVDSS and other methods on calling SVs.** Results are shown in terms of Precision (P), Recall (R), and F-measure (F1) with bold faced numbers indicating the best performance. Results are further broken down by considered regions of the genome. Tier 1 accounts for nearly half of the SVs and consists of 86% of the genome. Extended Tier 2 accounts for the remaining 14% of the genome and 50% of SVs and includes repetitive regions that are more difficult to genotype. See Supplementary Figure S5 for more detail on tiers.

cuteSV [56], sniffles [41], SVIM [57], and a recent preprint on a POA-based method, debreak [58]. We ran each caller setting the minimum SV support to 4 when analyzing the 30x CHM13 sample and to 2 when analyzing the 15x HG002 and HG007 samples. We then examined their insertions and deletions calls. We validated the calls of each tool against the set of SVs constructed with dipcall using Truvari [59], a SV evaluation framework which reports precision, recall, and F1 score for each method. We ignored genotype-level accuracy, i.e., we checked only for the presence of the corrected allele (see Supplementary Section D for more information on how we ran Truvari, as well as other tools used in our analysis). From this comparison, we further exclude calls made in regions of the reference genome not covered by both haplotypes, as any such call would be classified as false positive regardless of correctness.

On HG002 and HG007 samples, SVDSS outperforms the other callers' recall by 5-10% while achieving the highest (or the second highest) precision on the full genome (Table 1, *Full Genome* rows). SVDSS has been able to report 2,342 (+10% w.r.t. second-best approach) more correct calls on HG002 and 1,631 (+8%) more calls on HG007 without introducing many false calls. SVDSS also achieves the highest recall on CHM13 and reports 782 (+2%) more true positive calls than other methods while maintaining a very high precision. While SVDSS has the highest F1 score on CHM13, we note that the whole-genome improvements achieved by SVDSS over other approaches is less significant for this sample compared to the other two samples (improvement of 2-5% in recall and 1% in F1 while achieving similar precision to other tools). This is likely due to the homozygous nature of CHM13 making SV calling relatively easier for all approaches.

Figure 2(a) reports the length distribution of the SVs called by each tool on the HG007 sample. On HG007, the number of SVs reported by each tool ranges from 34,827 to 38,659 with SVIM reporting the lowest number of SVs and SVDSS reporting the highest number. Overall, all the tools report more insertions

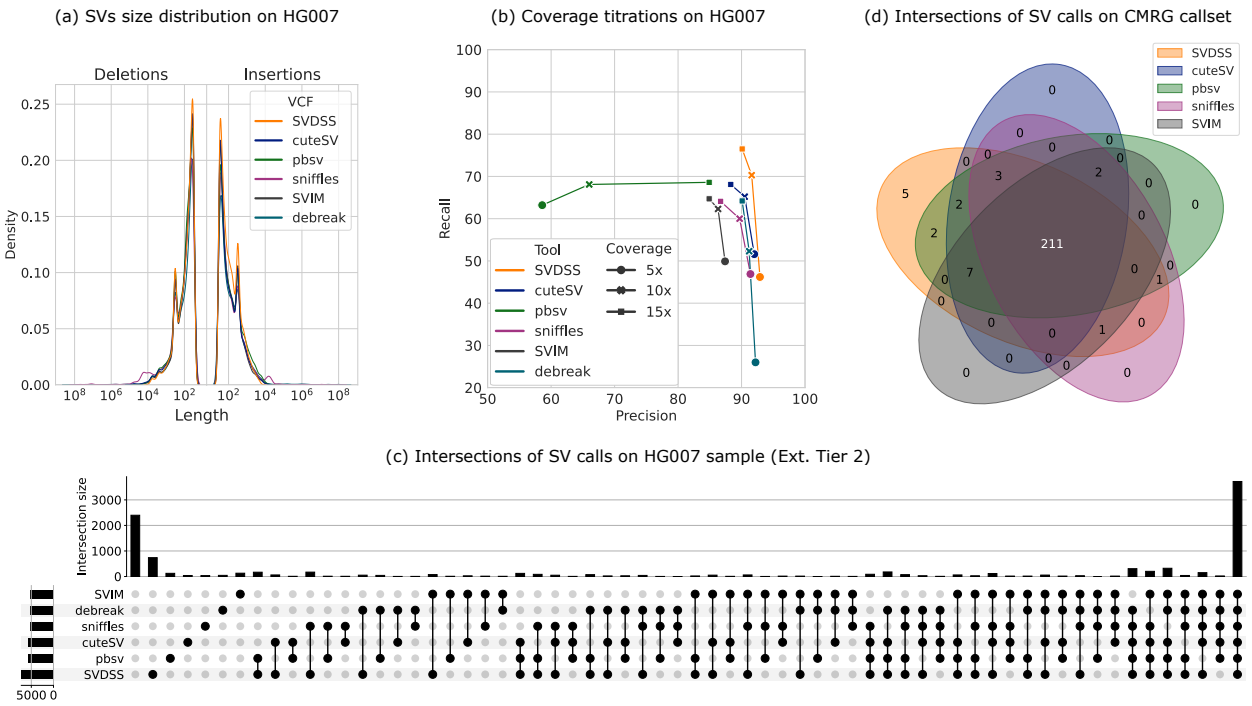


Figure 2: **Extended comparative analysis of SV calls across methods.** (a) Distribution of SVs lengths reported by different tools on HG007 (Full Genome). (b) Lineplot presenting results of the coverage titration for 5x, 10x, and 15x. (c) Analysis of shared calls (True Positives) between different tools on HG007 (Extended Tier 2). (d) Venn diagram showing shared calls (True Positives) between different tools on the 273 medically-relevant genes considered in the CMRG callset. To keep the Venn diagram cleaner, we decided to exclude `debreak` since it correctly called the least TPs. A supervenn figure including all tools is shown in Supplementary Figure S7.

than deletions with shorter SVs (of length $\leq 100bp$) being more frequent than longer ones. Moreover, all the tools show a clear peak at around 300bp reflecting *Alu* mobile elements.

We also repeated the above experiment on HG007 using different aligners to test how SV callers are influenced by how reads are aligned. We tested all 6 callers in combination with `minimap2` [60] and `ngmlr` [41] (Supplementary Table S2). We also noticed that `SVDSS` significantly improves our ability to predict SVs in comparison to state-of-the-art approaches using `minimap2` mapper, while being one of top performer tools using `ngmlr` mapper (Supplementary Table S2).

We also investigated how read coverage affects SV calling performance. To this aim, we subsampled the HG007 sample (coverage 15x) down to 5x and 10x and we ran the 5 considered approaches on these two newly-created samples. Our `SVDSS` approach was also able to outperform other approaches using 10x sequencing coverage in all the metrics of interest (precision, recall, and F1, see Figure 2(b) and Supplementary Table S3). When sample coverage is low (5x), `pbsv` achieves the highest recall (63.2%) at the expense of lower precision (58.6%) whereas other tools achieve similar high precision (ranging from 87.4% of `SVIM` to 92.9% of `SVDSS`) but low recall (ranging from 46.2% achieved by `SVDSS` to 51.6% achieved by `cuteSV`). As already pointed out in [58], `debreak` works poorly with low coverage samples. On the other hand, with higher coverages of 10x and 15x, `SVDSS` achieves the best precision and recall, outperforming other approaches.

Finally, our pipeline has the second-lowest runtime among the considered methods behind `cuteSV`. More details on runtime and performance are available in Online Methods.

Improved SVs calling in hard-to-analyze regions. For further analysis, we partitioned the genome into two sets of intervals (*tiers*) as previously done by GIAB [45]. Tier 1 accounts for nearly 86% of the genome spanning 2.51 Gbp, includes 50% or less of the total expected number of SVs, and is likely biased towards easy-to-call SVs (as stated in the README of the GIAB v0.6 callset provided at `https://ftp-trace.ncbi.nlm.nih.gov/giab/ftp/data/AshkenazimTrio/analysis/NIST_SVs_Integration_v0.6/README_SV_v0.6.txt`). Tier 2 accounts for nearly 0.8% of the genome and consists of $\sim$ 6000 difficult-to-genotype sites. The remaining 13% of the genome mostly consists of centromeres, telomeres, and microsatellite regions (e.g., Short Tandem Repeats) which are generally more difficult to genotype because of their repeat structure and due to the ambiguities of the reference genome. Because the high-quality assemblies that are the basis of our analysis include effectively complete genomes for each individual, we decided to extend Tier 2 to also include these remaining 13% regions (Extended Tier 2). This way, we are able to more thoroughly evaluate each methods accuracy across the entire human genome and we do not limit our analysis to easier-to-call regions (i.e., Tier 1). Our final partitioning consists of Tier 1 and Extended Tier 2, represented in Supplementary Figure S5.

In this analysis, we considered the callsets produced by `SVDSS`, `cuteSV`, `pbsv`, `sniffles`, `debreak` and `SVIM` starting from `pbmm2` alignments. Table 1 reports the results of this analysis. Results on both tiers follow the same trend as with the full genome, with `SVDSS` managing to call more correct SVs without introducing many false calls. As expected, all tools achieve higher accuracy on Tier 1 regions, that are easier to analyze. Furthermore, we observed that the improvement between performance of `SVDSS` and other tools widens in the Extended Tier 2 regions of the genome (Table 1). Remarkably, on difficult-to-analyze regions (i.e., extended Tier 2), `SVDSS` achieves the highest recall, outperforming other callers by 15%, 14% and 4% on the HG002, HG007 and CHM13 samples, respectively.

To further provide evidence of correctness for true positive calls in these hard regions, we analyzed how these calls are shared among the tested callers using an upset plot [61]. Upset plots are an alternative to Venn diagrams that represents more conveniently the intersections of multiple sets. Figure 2(c) shows that out of the 10,333 total SVs in the truth set for HG007 (i.e., the `dipcall` callset), 3,720 (36%) of these SVs are correctly called by all the tested approaches whereas 2,399 (23%) are not detected by any tool. Remarkably, 739 SVs (7%) are detected only by our pipeline, partially explaining the higher recall it is able to achieve. `SVIM` has the second-highest number of specific calls at 130. Supplementary Figure S14 shows the distribution of `SVDSS`-specific vs `SVIM`-specific calls on chr1, chr2 and chr3 of the HG007 sample. `SVDSS` also detects the highest number of SVs that would have been exclusive to other tools, i.e., 172 (1.6%) calls are shared by `SVDSS` and `sniffles`, and 169 (1.6%) are shared between `SVDSS` and `pbsv`.

We manually investigated some of the SVs that are exclusively called by `SVDSS`. Some of such calls are SVs that exhibit two different alleles on the two haplotypes. These SVs account for heterozygous SVs with two non-reference alleles (as defined in [62]), i.e., SVs genotyped 1/2 (see two examples in Supplementary Figures S10 and S11) as well as pairs of close SVs whose alleles come from different haplotypes (see an example in Supplementary Figure S12). We observed that a total of 343 SVs called exclusively by `SVDSS` and matching `dipcall` predictions on HG007 genome were located at exactly the same position as another called SV and are heterozygous SVs with two non-reference alleles, while a total of 227 SVs are close ($\leq$ 100bp) to another predicted SV (Supplementary Figure S6)".

Hard-to-analyze regions harbor SVs with clinical importance. To perform a more thorough analysis of the HG002 individual, we considered the CMRG (Challenging Medically Relevant Genes) callset provided in [51] and we evaluated callers' accuracy against it. The CMRG callset consists of 250 SVs falling in 126 challenging and medically relevant genes that were excluded from the previously published GIAB benchmark [45] due to their complexity: compound heterozygous insertions, complex variants in segmental duplications, and long tandem repeats. The CMRG callset was created by diploid assembly of the haplotypes using hifiasm and then `dipcall`, proving one more time the effectiveness of assembly-based methods for detecting hard-to-analyze SVs, when well-curated assemblies are available.

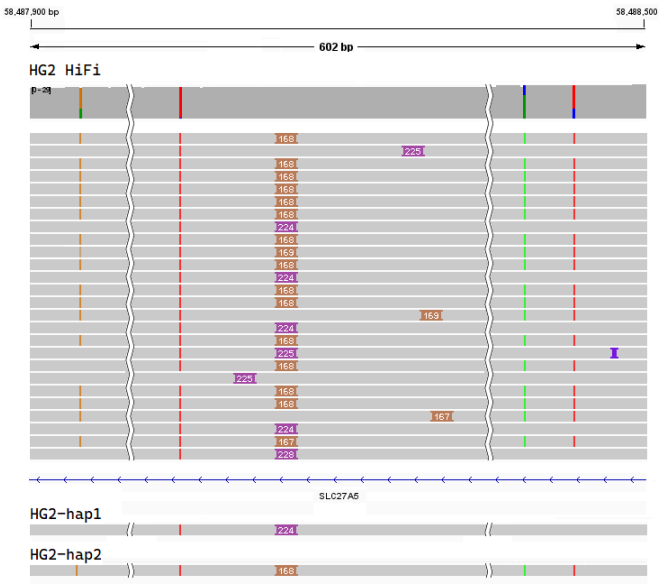
As done previously, we computed the accuracy of `SVDSS` and the other 5 SV callers using `Truvari`. Out of the 250 SVs contained in the CMRG callset, `SVDSS` correctly called 232 SVs followed by `pbsv` (228) and `cuteSV` (225), `SVIM` (221), `debreak` (220), and `sniffles` (218). As shown in Figure 2(d) (and Supplementary Figure S7, where all tools are considered), 5 SVs are exclusive to `SVDSS`, while 2 are missed exclusively by `SVDSS`: one was reported but with a length just under the evaluation threshold of `Truvari`, the other was missed due to being only detectable in clipped reads, which `SVDSS` does not consider by default. We then manually investigated the SVs that were exclusively called by `SVDSS`, discovering that all them exhibited two alleles, one per haplotype (i.e., heterozygous SVs with two non-reference alleles). This result confirms previous findings [51] that heterozygous insertions in tandem repeats are among the most challenging classes of SVs to discover with current methods.

Figure 3 shows one of the `SVDSS`-exclusive SVs, a double insertion inside the *SLC27A5* gene on chromosome 19. Although the two haplotypes can be easily distinguished by visual inspection of adjacent heterozygous SNPs, the tested callers disagree on which allele to call. For instance only `SVDSS` calls two alleles of length 168bp and 224bp agreeing with the CMRG callset, whereas `pbsv` and `sniffles` report only one of the two (168bp). Surprisingly, `cuteSV`, `SVIM`, and `debreak` report a single allele of length 185bp, which does not match any of the evidence from read alignment. Additionally, we considered the portion of the high-quality HG002 assembly covering that locus (`chr19:58487900-58488500`) and we checked its alignment against the reference genome (Figure 3 and Supplementary Figure S8). Although the considered locus is in a repetitive region (as also proven by the noisiness of the dotplots shown in Supplementary Figure S8), the haplotype alignment confirms the presence of two allelic insertions of different lengths.

`SVDSS` has extremely low baseline error rate. Finally, we further investigate the lower bound on baseline false discovery rate of `SVDSS` by comparing the HiFi reads from CHM13 against the high-quality T2T assembly [52] of the same sample. Given the almost perfect T2T CHM13 assembly produced using multiple orthogonal technologies, it is expected that an ideal SV caller would predict no SVs when comparing CHM13 reads against this assembly. Thus, we propose an experiment to establish a lower bound on the baseline false discovery rate of different methods by comparing how many SV calls they report on the CHM13 HiFi reads against its T2T assembly.

Ideally, the `SVDSS` pipeline should generate zero SVs calls in this scenario as no SFS should be extracted when querying smoothed CHM13 reads against the T2T assembly. However, this will not be the case in practice due to mapping ambiguities in repetitive regions of the genome. Still, we expect the method to produce very few variant calls.

As a side-objective, we will also investigate the resulting SV calls to find if our method has discovered any true SVs missing from the T2T assembly. Due to the effectively homozygous nature of the CHM13



CMRG:	chr19:58488284	SVLEN=168,224	GT	1/2
SVDSS:	chr19:58488284	SVLEN=168,224	GT	1/2
cuteSV:	chr19:58488291	SVLEN=185	GT	0/1
pbsv:	chr19:58488284	SVLEN=168	GT	0/1
svim:	chr19:58488290	SVLEN=185	GT	0/1
sniffles:	chr19:58488285	SVLEN=168	GT	0/1
debreak:	chr19:58488290	SVLEN=185	GT	1/1

Figure 3: **Example of an SV at a medically-relevant gene that has been correctly called exclusively by SVDSS.** On the right panel, IGV sketch of the 602bp region around the SV (the full region is reported in Supplementary Figure S13). The sketch reports the HiFi reads alignment along with the haplotype alignment performed using `minimap2` (as part of the `dipcall` pipeline). On the left panel, details on the SVs reported by the CMRG callset, SVDSS, and the other alignment-based callers considered in our evaluation.

genome, any true variant discovered must be homozygous. However it is possible that artifacts accumulated in the cell-line and actual heterozygosities in the genome may result in heterozygous SVs being reported.

We built the FMD index for v1.1 of the CHM13 assembly and extracted SFS from CHM13 HiFi reads smoothed against the T2T assembly using this index. We then passed the SFS through the SVDSS pipeline for SV discovery. Our pipeline discovers a total of 102 SVs. For comparison, we repeated the above experiment with the other tools `pbsv`, `cuteSV`, `SVIM`, `debreak` and `sniffles`. Table 2 includes a summary of the result. We calculate the baseline False Discovery Rate for each tool as the number of calls it makes against T2T divided by the number of calls it makes against GRCh38. SVDSS has the lowest number of calls against the T2T assembly and also has the lowest baseline error rate.

We further investigate if any of our calls are indeed true variants. The T2T project provides a list of known heterozygous sites on CHM13 [63, 52] and 13 of our SV calls intersect these regions, suggesting that they may be actual heterozygous alleles missing from the homozygous assembly. We also report the number of intersecting calls in Table 2 for every tool. SVDSS has the highest ratio of calls intersecting known heterozygous regions. We performed additional filtering of the calls using Merfin [64], a variant call polishing tool that filters VCF files based on whether the variants introduce k-mers not found in the sequencing reads. Only one of our calls passes Merfin's filtering and we verify that the call seems to be a heterozygous site (Supplementary Figure S9).

In summary, SVDSS produces only 102 calls using CHM13 HiFi reads against the T2T CHM13 assembly, some of which may be actual true heterozygous variants. Furthermore, with our earlier experiments showing an average of 33,000 SV calls per sample, this amounts to a baseline error rate of less than 0.4% showing that SVDSS is robust to false detection of variants.

Tool	GRCh38 calls	T2T calls	Baseline FDR	Het Intersections	Het Precision
SVDSS	**23777**	**102**	**0.4%**	13	**12.7%**
cuteSV	22654	667	2.94%	23	3.4%
pbsv	23707	616	2.59%	28	4.5%
sniffles	22680	314	1.38%	22	7.0%
SVIM	22176	948	4.27%	**29**	3.0%
debreak	23432	834	3.55%	24	2.8%

Table 2: **Comparison of baseline FDR rate of SVDSS with other methods.** Number of SV calls against both the reference genome and the CHM13 assembly is included. Baseline FDR is calculated as division of first two columns for each tool. The last two columns report the number of known CHM13 heterozygous (Het) sites covered by each method and the precision of the method calculated as the number of covered heterzoygous sites divided by the number of predicted calls. Bold face indicates best performance.

3 Discussion

We have introduced SVDSS, a method for SV discovery that combines advantages of different SV discovery approaches to achieve significant improvements in SV calling. A highlight of SVDSS is its much higher recall compared to other approaches in repeated regions of the genome (i.e., extended tier 2), and also its overall higher accuracy in particular in repetitive and traditionally hard-to-genotype regions of the genome. We also observed that reducing sequencing coverage impacts SVDSS less than other approaches. Thus SVDSS can accurately predict SVs in low-coverage sequenced samples. Furthermore, utilizing the recent CHM13 assembly produced by T2T consortium, we could estimate baseline error rare for each methods and observed that SVDSS further has the lowest baseline error rate followed by sniffles.

While the availability of low-error long-read data enables more extensive variant discovery on new samples, SV discovery in repetitive regions of the genome such as STRs and microsatellites remains challenging but also hard to evaluate. This is evidenced by comparisons presented in this manuscript. Despite SVDSS's significant performance improvements in repetitive regions, precision and recall in these regions are still lower than on the rest of the genome.

SVDSS currently supports the discovery of unbalanced structural variants, i.e. deletions and insertions, however as the underlying SFS signatures capture nearly all variation in the genome, a next step could be to extend the method to finding other classes of SVs such as inversions and duplications. Our current best technique for creating SV truth sets (dipcall) does not evaluate inversions and duplications, yet a recent study [28] provides one of the first gold standards.

Throughout this work we highlight the importance of accurate benchmarks of SV calling methods. We evaluated SVDSS on a recent benchmark extensively curated over the HG002 sample [51] with the specific purpose of producing SVs occurring in genes of medical relevance. These genes are considered challenging for mapping-based and assembly-based SV prediction methods even from highly accurate long reads. This benchmark revealed that other methods fail to call heterozygous indels in highly homozygous regions or erroneous indels interpreted by a consensus approach. SVDSS is the only method able to discover 5 of such SVs in medically relevant gene regions. We believe the current examples of accurate prediction of multi-allelic heterozygous events based on SVDSS indicates the merit of extending this approach for genotype prediction of SVs.

Acknowledgements This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grants agreements No. 872539 and 956229 (P.B., R.C.). This work has also been supported in part by NSF award DBI-2042518 to F.H. The funding body had no role in the design of the study and collection, analysis, and interpretation of data and in writing

the manuscript. R.C was supported by ANR Transipedia, SeqDigger, Inception and PRAIRIE grants (ANR-18-CE45-0020, ANR-19-CE45-0008, PIA/ANR16-CONV-0005, ANR-19-P3IA-0001).

Author Contributions Statement L.D. and P.K. devised and implemented the approach. L.D. and P.K. performed the experimental evaluation. P.B., F.H. and R.C. conceived the study, supervised and coordinated the work. All authors wrote, reviewed, edited and approved the manuscript.

Competing Interests Statement The authors declare no competing interests.

References

[1] Can Alkan, Bradley P Coe, and Evan E Eichler. Genome structural variation discovery and genotyping. *Nature Reviews Genetics*, 12(5):363–376, 2011.

[2] Lars Feuk, Andrew R Carson, and Stephen W Scherer. Structural variation in the human genome. *Nature Reviews Genetics*, 7(2):85–97, 2006.

[3] Steve S Ho, Alexander E Urban, and Ryan E Mills. Structural variation in the sequencing era. *Nature Reviews Genetics*, 21(3):171–189, 2020.

[4] Ryan E Mills, Klaudia Walter, Chip Stewart, Robert E Handsaker, Ken Chen, Can Alkan, Alexej Abyzov, Seungtai Chris Yoon, Kai Ye, R Keira Cheetham, et al. Mapping copy number variation by population-scale genome sequencing. *Nature*, 470(7332):59–65, 2011.

[5] Peter H Sudmant, Tobias Rausch, Eugene J Gardner, Robert E Handsaker, Alexej Abyzov, John Huddleston, Yan Zhang, Kai Ye, Goo Jun, Markus Hsi-Yang Fritz, et al. An integrated map of structural variation in 2,504 human genomes. *Nature*, 526(7571):75–81, 2015.

[6] Mark JP Chaisson, John Huddleston, Megan Y Dennis, Peter H Sudmant, Maika Malig, Fereydoun Hormozdiari, Francesca Antonacci, Urvashi Surti, Richard Sandstrom, Matthew Boitano, et al. Resolving the complexity of the human genome using single-molecule sequencing. *Nature*, 517(7536):608–611, 2015.

[7] Peter Ebert, Peter A Audano, Qihui Zhu, Bernardo Rodriguez-Martin, David Porubsky, Marc Jan Bonder, Arvis Sulovari, Jana Ebler, Weichen Zhou, Rebecca Serra Mari, et al. Haplotype-resolved diverse human genomes and integrated analysis of structural variation. *Science*, 372(6537), 2021.

[8] Michael M Khayat, Sayed Mohammad Ebrahim Sahraeian, Samantha Zarate, Andrew Carroll, Huixiao Hong, Bohu Pan, Leming Shi, Richard A Gibbs, Marghoob Mohiyuddin, Yuanting Zheng, et al. Hidden biases in germline structural variant detection. *Genome Biology*, 22(1):1–15, 2021.

[9] Shobana Sekar, Livia Tomasini, Christos Proukakis, Taejeong Bae, Logan Manlove, Yeongjun Jang, Soraya Scuderi, Bo Zhou, Maria Kalyva, Anahita Amiri, et al. Complex mosaic structural variations in human fetal brains. *Genome Research*, 30(12):1695–1704, 2020.

[10] Claudia MB Carvalho and James R Lupski. Mechanisms underlying structural variant formation in genomic disorders. *Nature Reviews Genetics*, 17(4):224–238, 2016.

[11] Peter A Audano, Arvis Sulovari, Tina A Graves-Lindsay, Stuart Cantsilieris, Melanie Sorensen, AnneMarie E Welch, Max L Dougherty, Bradley J Nelson, Ankeeta Shah, Susan K Dutcher, et al. Characterizing the major structural variant alleles of the human genome. *Cell*, 176(3):663–675, 2019.

[12] Xuefang Zhao, Ryan L Collins, Wan-Ping Lee, Alexandra M Weber, Yukyung Jun, Qihui Zhu, Ben Weisburd, Yongqing Huang, Peter A Audano, Harold Wang, et al. Expectations and blind spots for structural variation detection from long-read assemblies and short-read genome sequencing technologies. *The American Journal of Human Genetics*, 108(5):919–928, 2021.

[13] Paweł Stankiewicz and James R Lupski. Structural variation in the human genome and its role in disease. *Annual Review of Medicine*, 61:437–455, 2010.

[14] Andrew J Sharp, Ze Cheng, and Evan E Eichler. Structural variation of the human genome. *Annu. Rev. Genomics Hum. Genet.*, 7:407–442, 2006.

[15] Ryan L Collins, Harrison Brand, Konrad J Karczewski, Xuefang Zhao, Jessica Alföldi, Laurent C Francioli, Amit V Khera, Chelsea Lowther, Laura D Gauthier, Harold Wang, et al. A structural variation reference for medical and population genetics. *Nature*, 581(7809):444–451, 2020.

[16] Peter H Sudmant, Swapan Mallick, Bradley J Nelson, Fereydoun Hormozdiari, Niklas Krumm, John Huddleston, Bradley P Coe, Carl Baker, Susanne Nordenfelt, Michael Bamshad, et al. Global diversity, population stratification, and selection of human copy-number variation. *Science*, 349(6253), 2015.

[17] Peter H Sudmant, John Huddleston, Claudia R Catacchio, Maika Malig, LaDeana W Hillier, Carl Baker, Kiana Mohajeri, Ivanela Kondova, Ronald E Bontrop, Stephan Persengiev, et al. Evolution and diversity of copy number variation in the great ape lineage. *Genome Research*, 23(9):1373–1382, 2013.

[18] Andrew Fortna, Young Kim, Erik MacLaren, Kriste Marshall, Gretchen Hahn, Lynne Meltesen, Matthew Brenton, Raquel Hink, Sonya Burgers, Tina Hernandez-Boussard, et al. Lineage-specific gene duplication and loss in human and great ape evolution. *PLoS Biology*, 2(7):e207, 2004.

[19] Matthew Hurles. Gene duplication: the genomic trade in spare parts. *PLoS Biology*, 2(7):e206, 2004.

[20] Jeremiah A Wala, Pratiti Bandopadhayay, Noah F Greenwald, Ryan O'Rourke, Ted Sharpe, Chip Stewart, Steve Schumacher, Yilong Li, Joachim Weischenfeldt, Xiaotong Yao, et al. Svaba: genome-wide detection of structural variants and indels by local assembly. *Genome Research*, 28(4):581–591, 2018.

[21] Tom Walsh, Jon M McClellan, Shane E McCarthy, Anjené M Addington, Sarah B Pierce, Greg M Cooper, Alex S Nord, Mary Kusenda, Dheeraj Malhotra, Abhishek Bhandari, et al. Rare structural variants disrupt multiple genes in neurodevelopmental pathways in schizophrenia. *Science*, 320(5875):539–543, 2008.

[22] Donald F Conrad, Dalila Pinto, Richard Redon, Lars Feuk, Omer Gokcumen, Yujun Zhang, Jan Aerts, T Daniel Andrews, Chris Barnes, Peter Campbell, et al. Origins and functional impact of copy number variation in the human genome. *Nature*, 464(7289):704–712, 2010.

[23] Christian R Marshall, Abdul Noor, John B Vincent, Anath C Lionel, Lars Feuk, Jennifer Skaug, Mary Shago, Rainald Moessner, Dalila Pinto, Yan Ren, et al. Structural variation of chromosomes in autism spectrum disorder. *The American Journal of Human Genetics*, 82(2):477–488, 2008.

[24] ICGC The, TCGA Pan-Cancer Analysis of Whole, Genomes Consortium, et al. Pan-cancer analysis of whole genomes. *Nature*, 578(7793):82, 2020.

[25] Yilong Li, Nicola D Roberts, Jeremiah A Wala, Ofer Shapira, Steven E Schumacher, Kiran Kumar, Ekta Khurana, Sebastian Waszak, Jan O Korbel, James E Haber, et al. Patterns of somatic structural variation in human cancer genomes. *Nature*, 578(7793):112–121, 2020.

[26] Kai Ye, Jiayin Wang, Reyka Jayasinghe, Eric-Wubbo Lameijer, Joshua F McMichael, Jie Ning, Michael D McLellan, Mingchao Xie, Song Cao, Venkata Yellapantula, et al. Systematic discovery of complex insertions and deletions in human cancers. *Nature Medicine*, 22(1):97–104, 2016.

[27] Emma C Scott, Eugene J Gardner, Ashiq Masood, Nelson T Chuang, Paula M Vertino, and Scott E Devine. A hot l1 retrotransposon evades somatic repression and initiates human colorectal cancer. *Genome Research*, 26(6):745–755, 2016.

[28] David Porubsky, Wolfram Höps, Hufsah Ashraf, PingHsun Hsieh, Bernardo Rodriguez-Martin, Feyza Yilmaz, Jana Ebler, Pille Hallast, Flavia Angela Maria Maggiolini, William T Harvey, et al. Recurrent inversion polymorphisms in humans associate with genetic instability and genomic disorders. *Cell*, 185(11):1986–2005, 2022.

[29] David Porubsky, Ashley D Sanders, Wolfram Höps, PingHsun Hsieh, Arvis Sulovari, Ruiyang Li, Ludovica Mercuri, Melanie Sorensen, Shwetha C Murali, David Gordon, et al. Recurrent inversion toggling and great ape genome evolution. *Nature Genetics*, 52(8):849–858, 2020.

[30] Songbo Wang, Jiadong Lin, Xiaofei Yang, Zihang Li, Tun Xu, Peng Jia, Tingjie Wang, Bo Wang, Liangshuo Hu, and Kai Ye. Long read sequencing reveals sequential complex rearrangements driven by hepatitis b virus integration. *bioRxiv*, 2021.

[31] Justin M Zook, Nancy F Hansen, Nathan D Olson, Lesley Chapman, James C Mullikin, Chunlin Xiao, Stephen Sherry, Sergey Koren, Adam M Phillippy, Paul C Boutros, et al. A robust benchmark for detection of germline large deletions and insertions. *Nature Biotechnology*, 38(11):1347–1355, 2020.

[32] Alexej Abyzov, Alexander E Urban, Michael Snyder, and Mark Gerstein. Cnvnator: an approach to discover, genotype, and characterize typical and atypical cnvs from family and population genome sequencing. *Genome Research*, 21(6):974–984, 2011.

[33] Zamin Iqbal, Mario Caccamo, Isaac Turner, Paul Flicek, and Gil McVean. De novo assembly and genotyping of variants using colored de bruijn graphs. *Nature Genetics*, 44(2):226–232, 2012.

[34] Sai Chen, Peter Krusche, Egor Dolzhenko, Rachel M Sherman, Roman Petrovski, Felix Schlesinger, Melanie Kirsche, David R Bentley, Michael C Schatz, Fritz J Sedlazeck, et al. Paragraph: a graph-based structural variant genotyper for short-read sequence data. *Genome Biology*, 20(1):1–13, 2019.

[35] Jiadong Lin, Xiaofei Yang, Walter Kosters, Tun Xu, Yanyan Jia, Songbo Wang, Qihui Zhu, Mallory Ryan, Li Guo, Chengsheng Zhang, et al. Mako: A graph-based pattern growth approach to detect complex structural variants. *Genomics, Proteomics & Bioinformatics*, 2021.

[36] Eugene J Gardner, Vincent K Lam, Daniel N Harris, Nelson T Chuang, Emma C Scott, W Stephen Pittard, Ryan E Mills, Scott E Devine, 1000 Genomes Project Consortium, et al. The mobile element locator tool (melt): population-scale mobile element discovery and biology. *Genome Research*, 27(11):1916–1929, 2017.

[37] Arda Soylev, Thong Minh Le, Hajar Amini, Can Alkan, and Fereydoun Hormozdiari. Discovery of tandem and interspersed segmental duplications using high-throughput sequencing. *Bioinformatics*, 35(20):3923–3930, 2019.

[38] Jana Ebler, Alexander Schönhuth, and Tobias Marschall. Genotyping inversions and tandem duplications. *Bioinformatics*, 33(24):4015–4023, 2017.

[39] Jacob J Michaelson and Jonathan Sebat. forestsv: structural variant discovery through statistical learning. *Nature Methods*, 9(8):819–821, 2012.

[40] Ryan M Layer, Colby Chiang, Aaron R Quinlan, and Ira M Hall. Lumpy: a probabilistic framework for structural variant discovery. *Genome Biology*, 15(6):1–19, 2014.

[41] Fritz J Sedlazeck, Philipp Rescheneder, Moritz Smolka, Han Fang, Maria Nattestad, Arndt Von Haeseler, and Michael C Schatz. Accurate detection of complex structural variations using single-molecule sequencing. *Nature Methods*, 15(6):461–468, 2018.

[42] Suzanne Sindi, Elena Helman, Ali Bashir, and Benjamin J Raphael. A geometric approach for classification and comparison of structural variants. *Bioinformatics*, 25(12):i222–i230, 2009.

[43] Paul Medvedev, Monica Stanciu, and Michael Brudno. Computational methods for discovering structural variation with next-generation sequencing. *Nature Methods*, 6(11):S13–S20, 2009.

[44] Medhat Mahmoud, Nastassia Gobet, Diana Ivette Cruz-Dávalos, Ninon Mounier, Christophe Dessimoz, and Fritz J Sedlazeck. Structural variant calling: the long and the short of it. *Genome Biology*, 20(1):1–14, 2019.

[45] Justin M Zook, Nancy F Hansen, Nathan D Olson, Lesley Chapman, James C Mullikin, Chunlin Xiao, Stephen Sherry, Sergey Koren, Adam M Phillippy, Paul C Boutros, et al. A robust benchmark for detection of germline large deletions and insertions. *Nature Biotechnology*, 38(11):1347–1355, 2020.

[46] Jonathan R Belyeu, Harrison Brand, Harold Wang, Xuefang Zhao, Brent S Pedersen, Julie Feusier, Meenal Gupta, Thomas J Nicholas, Joseph Brown, Lisa Baird, et al. De novo structural mutation rates and gamete-of-origin biases revealed through genome sequencing of 2,396 families. *The American Journal of Human Genetics*, 108(4):597–607, 2021.

[47] Parsoa Khorsand and Fereydoun Hormozdiari. Nebula: Ultra-efficient mapping-free structural variant genotyper. *Nucleic Acids Research*, 49(8):e47–e47, 2021.

[48] Lasse Maretty, Jacob Malte Jensen, Bent Petersen, Jonas Andreas Sibbesen, Siyang Liu, Palle Villesen, Laurits Skov, Kirstine Belling, Christian Theil Have, Jose MG Izarzugaza, et al. Sequencing and de novo assembly of 150 genomes from denmark as a population reference. *Nature*, 548(7665):87–91, 2017.

[49] Jia-Yuan Zhang, Hannah Roberts, David SC Flores, Antony J Cutler, Andrew C Brown, Justin P Whalley, Olga Mielczarek, David Buck, Helen Lockstone, Barbara Xella, et al. Using de novo assembly to identify structural variation of eight complex immune system gene regions. *PLoS Computational Biology*, 17(8):e1009254, 2021.

[50] Lu Zhang, Xin Zhou, Ziming Weng, and Arend Sidow. De novo diploid genome assembly for genome-wide structural variant detection. *NAR Genomics and Bioinformatics*, 2(1):lqz018, 2020.

[51] Justin Wagner, Nathan D. Olson, Lindsay Harris, Jennifer McDaniel, Haoyu Cheng, Arkarachai Fungtammasan, Yih-Chii Hwang, Richa Gupta, Aaron M. Wenger, William J. Rowell, Ziad M. Khan, Jesse Farek, Yiming Zhu, Aishwarya Pisupati, Medhat Mahmoud, Chunlin Xiao, Byunggil Yoo, Sayed Mohammad Ebrahim Sahraeian, Danny E. Miller, David Jáspez, José M. Lorenzo-Salazar, Adrián Muñoz-Barrera, Luis A. Rubio-Rodríguez, Carlos Flores, Giuseppe Narzisi, Uday Shanker Evani, Wayne E. Clarke, Joyce Lee, Christopher E. Mason, Stephen E. Lincoln, Karen H. Miga, Mark T. W. Ebbert, Alaina Shumate, Heng Li, Chen-Shan Chin, Justin M. Zook, and Fritz J. Sedlazeck. Curated variation benchmarks for challenging medically relevant autosomal genes. *Nature Biotechnology*, Feb 2022.

[52] Sergey Nurk, Sergey Koren, Arang Rhie, Mikko Rautiainen, Andrey V Bzikadze, Alla Mikheenko, Mitchell R Vollger, Nicolas Altemose, Lev Uralsky, Ariel Gershman, et al. The complete sequence of a human genome. *Science*, 376(6588):44–53, 2022.

[53] Parsoa Khorsand, Luca Denti, Human Genome Structural Variant Consortium, Paola Bonizzoni, Rayan Chikhi, and Fereydoun Hormozdiari. Comparative genome analysis using sample-specific string detection in accurate long reads. *Bioinformatics Advances*, 1(1):vbab005, 2021.

[54] Heng Li, Jonathan M Bloom, Yossi Farjoun, Mark Fleharty, Laura Gauthier, Benjamin Neale, and Daniel MacArthur. A synthetic-diploid benchmark for accurate variant-calling evaluation. *Nature Methods*, 15(8):595–597, 2018.

[55] Gunjan Baid, Daniel E Cook, Kishwar Shafin, Taedong Yun, Felipe Llinares-López, Quentin Berthet, Anastasiya Belyaeva, Armin Töpfer, Aaron M Wenger, William J Rowell, et al. Deepconsensus improves the accuracy of sequences with a gap-aware sequence transformer. *Nature Biotechnology*, pages 1–7, 2022.

[56] Tao Jiang, Yongzhuang Liu, Yue Jiang, Junyi Li, Yan Gao, Zhe Cui, Yadong Liu, Bo Liu, and Yadong Wang. Long-read-based human genomic structural variation detection with cutesv. *Genome Biology*, 21(1):1–24, 2020.

[57] David Heller and Martin Vingron. Svim: structural variant identification using mapped long reads. *Bioinformatics*, 35(17):2907–2915, 2019.

[58] Yu Chen, Amy Wang, Courtney Barkley, Xinyang Zhao, Min Gao, Micky Edmonds, and Zechen Chong. Debreak: Deciphering the exact breakpoints of structural variations using long sequencing reads. *Research Square*, 2022.

[59] Adam C English, Vipin K Menon, Richard Gibbs, Ginger A Metcalf, and Fritz J Sedlazeck. Truvari: Refined structural variant comparison preserves allelic diversity. *bioRxiv*, 2022.

[60] Heng Li. Minimap2: pairwise alignment for nucleotide sequences. *Bioinformatics*, 34(18):3094–3100, 2018.

[61] Alexander Lex, Nils Gehlenborg, Hendrik Strobelt, Romain Vuillemot, and Hanspeter Pfister. UpSet: visualization of intersecting sets. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1983–1992, 2014.

[62] Luca Denti, Marco Previtali, Giulia Bernardini, Alexander Schönhuth, and Paola Bonizzoni. Malva: genotyping by mapping-free allele detection of known variants. *Iscience*, 18:20–27, 2019.

[63] Ann M Mc Cartney, Kishwar Shafin, Michael Alonge, Andrey V Bzikadze, Giulio Formenti, Arkarachai Fungtammasan, Kerstin Howe, Chirag Jain, Sergey Koren, Glennis A Logsdon, et al. Chasing perfection: validation and polishing strategies for telomere-to-telomere genome assemblies. *Nature Methods*, pages 1–9, 2022.

[64] Giulio Formenti, Arang Rhie, Brian P Walenz, Françoise Thibaud-Nissen, Kishwar Shafin, Sergey Koren, Eugene W Myers, Erich D Jarvis, and Adam M Phillippy. Merfin: improved variant filtering, assembly evaluation and polishing via k-mer validation. *Nature Methods*, pages 1–9, 2022.

4 Methods

4.1 Sample-specific string computation and assembly

Sample-specific Substring-free Strings (SFS) are defined as sequences that are specific to a "target" set of strings (a genome or sequencing sample) with respect to another "reference" set of strings (another genome or sequencing sample) [53]. The "substring-free" part means that they do not occur as substrings of each other. Note that, in the context of SV discovery, the "reference" will always be an assembled reference genome, e.g., GRCh38, and the "target" here is a set of reads. SFS can be optimally computed using the Ping-Pong algorithm, presented in [53]. Ping-Pong builds the FMD index [65] of the reference genome and queries the reads of the target sample against this index to report substrings that are not present in the index. The FMD index is a bidirectional text index with constant-time forward and backward search operations, thus allowing for efficient computation of SFS.

When SFS are computed between a reference genome and a target sample, they capture nearly all variations expressed in the sample with respect to the reference genome, as shown in [53]. Indeed, each sequencing read including a variant produces at least one SFS supporting the variant, hence a variant will be supported by at least one SFS per read covering it. SV breakpoints usually result in novel sequences that are captured as SFS. However, due to the "shortest" property of SFS, the entire SV sequence is not necessarily covered by a single SFS: a read may produce several overlapping SFS for long variations. To remove unnecessary redundancy in the information captured by overlapping SFS, we newly assemble all such overlapping SFS into longer strings called "superstrings". Assembling SFS into superstrings also reduces the number of SFS by an order of magnitude, making any downstream analysis more efficient.

As SFS on each read are naturally sorted based on their start positions, the assembly stage can be implemented as a single pass over the SFS on each read, merging each SFS with the next one if they overlap. The resulting superstring can further be merged with the next SFS if they also overlap, etc. More formally, on a read R where k consecutive SFS are overlapping such that $R[i_1, j_1]$ overlaps with $R[i_2, j_2]$ and $R[i_2, j_2]$ overlaps with $R[i_3, j_3]$ and $\dots R[i_{k-1}, j_{k-1}]$ overlaps with $R[i_k, j_k]$, we merge the strings into the single superstring $R[i_1, j_k]$.

The SFS assembly procedure effectively merges all the SFS belonging to the same variant into a single long superstring. This results in superstrings from the same variant to have similar length, sequence and position with respect to the reference genome which allows them to be easily clustered for SV prediction.

4.2 Read Smoothing

The SFS extraction step (Ping-Pong algorithm) requires reads with low error-rates for optimal performance as sequencing errors can result in millions of undesirable SFS. While most such SFS can be filtered later on, they can negatively affect the accuracy and will increase runtime by adding excess processing. Furthermore, the presence of millions of SNPs and small indels in a sample also results in tens of millions of additional SFS being extracted that are not directly useful for genotyping SVs. To solve both of the above problems, we introduce a preprocessing step called *"read smoothing"* that aims to eliminate both sequencing errors and short variants from input reads. The smoothing algorithm starts from read alignments (a BAM file) and uses information from the CIGAR strings of each alignment to remove any short mismatch between a read and the reference genome.

In more detail, for segments reported as a match between a read and the reference genome (CIGAR operation `'M'`), the algorithm replaces the read sequence with the corresponding sequence from the reference genome, automatically removing any single-base mismatches (i.e., sequencing errors or potential SNPs) in the process. For short deletions (CIGAR operation `'D'`), the algorithm removes the deletion from the read by copying back the deleted bases from the reference sequence. Short insertions (CIGAR operation `'I'`) are similarly smoothed by removing the inserted bases from the read. Using the default parameters, deletions and insertions are smoothed if they are shorter than 20bp. Note that smoothing insertions or deletions, i.e., removing them from the alignment, results in the extension of the `'M'` sections of the CIGAR string. Finally, soft-clipped regions (CIGAR operation `'S'`) are retained as they include potentially long inserted

or deleted sequences: any SNP or sequencing error inside clipped regions cannot be corrected as a result. As a result of the smoothing algorithm, a smoothed read's CIGAR strings will have significantly fewer edit operations than that the original read and it will consist of one or more very long `'M'` segments with large INDELs in between, potentially surrounded with soft-clipped regions. Supplementary Figure S3 illustrates the smoothing procedure on an example read. We note that the Ping-Pong algorithm will not produce any SFS that is entirely contained in a `'M'` section of a smoothed read as the corresponding sequence has been replaced base-by-base with reference genome sequence. Therefore, the number of SFS extracted from smoothed reads is significantly smaller than the number of SFS extracted from original reads.

The smoothing algorithm only works with primary alignments and non-primary alignments are ignored. This is to avoid complications arising from having multiple different smoothed version of reads with multiple alignments.

Smoothing relies on correctness of read alignments. If an alignment is thought to be inaccurate, the smoothing algorithm does not modify it. To this aim, during its execution, the algorithm keeps track of the average number of mismatches between the `'M'` segments of alignments and the corresponding reference sequence: any read that has more than 3 times the average mismatch rate is ignored, i.e., is not modified.

On a more technical note, we point out that the above modifications do not change the overall mapping of the read as the mapping positions (begin and end) remain the same. As a result, the algorithm will not change the order of the reads in a sorted BAM file. This allows us to quickly reconstruct a sorted BAM file without the need to sort it again. However, because the size of the reads may have changed, the index of the original BAM files is no longer valid for the smoothed BAM and it has to be indexed again with `samtools index`.

In our experiments, smoothing effectively reduces the number of extracted SFS by over 90%, while having effectively no impact on the SV calling pipeline's recall. Out of the 6.2M reads for the CHM13 samples, around 5M are smoothed and the rest are deemed to have unreliable mappings and are discarded. The 1.2M non-smoothed reads from CHM13 are responsible for more than 82% of all SFS extracted from that sample after smoothing. However, the SFS extracted from non-smoothed reads do not contribute to increasing the method's recall at all. Indeed excluding the SFS extracted from non-smoothed reads increases the method's precision while leaving the recall unaffected. This justifies the exclusion of non-smoothed reads from the `SVDSS` pipeline. Further analysis shows that nearly all non-smoothed reads map to centromere regions of the CHM13. Supplementary Figure S4 shows the distribution of mapping positions of reads from chr1 on both CHM13 and GRCh38. The large gap around the centromere when mapping to GRCh38 explains the poor performance of non-smoothed reads when predicting SVs against the reference genome.

In summary, read smoothing is a critical preprocessing step of the `SVDSS` pipeline. It reduces the number of retrieved SFS and increases the specificity of the extracted SFS which results in higher precision in predicting SVs without deteriorating recall. The procedure is also computationally very lightweight, as it essentially rewrites the BAM file in a single pass with minor modifications. As a result, smoothing is an effective method for increasing the specificity of SFS for SV calling and improving the computational efficiency of the pipeline.

4.3 SV Discovery

The main SV calling algorithm consists of three main steps (see Figure 1 steps 3A, 3B, 3C):

1. Superstrings constructed from the SFS strings are "placed" on the reference genome by extracting their alignments from read alignments. The superstrings are then clustered based on their aligned loci. Each cluster represents one or more SVs that are close to each other and may also contain multiple alleles.

2. Each cluster is further clustered based on length to generate up to two haplotype candidates (taking into account the diploidy of the human genome). Each haplotype cluster candidate is assembled with Partial Order Alignment (POA) to yield a consensus sequence.

3. Each haplotype candidate is locally realigned back to the reference genome region corresponding to its cluster and SVs are called based on the alignment.

We will explain each step in more details in the following subsections.

4.3.1 Superstring placement and clustering

Aligning superstrings back to the reference genome would be time-consuming and error-prone due to their relatively short lengths. However these superstrings were already (indirectly) mapped as part of the mapping of the reads they are part of. Hence in practice we do not align superstrings directly to the reference genome but instead their alignment is extracted from the alignment of their originating reads. We refer to this as superstring placement. Assuming $R[i, j]$ is a superstring that spans positions $i \dots j$ on read R, by knowing the mapping position of R, we can easily place the superstring on the reference genome by analyzing the corresponding CIGAR portion (i.e., CIGAR sections covering positions $i \dots j$). As already pointed out before, thanks to read smoothing, SFS (and consequently superstrings) cannot be entirely contained in a `'M'` section of a smoothed read alignment (CIGAR) and therefore span its `'I'` and `'D'` sections. For superstrings spanning a `'D'` section, all the bases are already placed on the reference genome and no additional computation is necessary. On the other hand, when a superstring spans a `'I'` section, it often covers just a portion of the inserted sequence. In such a case, since the inserted sequence cannot be placed on the reference genome, it is challenging to fully place the superstring. To deal with this issue, we extend each superstring that does not fully cover a `'I'` section until it fully covers it. In other words, we extend the superstring until it covers (on each side) a base that can be placed on the reference genome (i.e., that is not part of the inserted sequence).

To further boost the informative content of the superstrings and to make the following steps of the pipeline easier and more accurate, each placed superstring is further extended on the read on both sides until we reach a perfectly mappable (i.e., that can be mapped to the reference genome with no errors) and locally unique (i.e., is not repeated in the considered window) k-mer anchor. The default value for k is 7 and the default window size is 100bp on each side of the superstring. The superstrings that cannot be extended in this manner are ignored. Figure 1 3A shows this extension procedure. The k-mer anchoring idea was influenced by LongShot [66].

Finally, we cluster the superstrings based on their mapping locations: superstrings that have close enough mappings (by default less than 500bp apart) are placed in the same cluster. The resulting cluster's interval is defined as the smallest interval in the genome that completely includes all of its superstrings and the includes either a single SV or several close or overlapping SVs possibly from different haplotypes.

4.3.2 POA assembly and SV detection

Each cluster so far includes one or more close SVs. However, as the human genome is diploid, the SVs might indeed be from different haplotypes. To resolve the different haplotypes, we further split each cluster into subclusters of superstrings of similar size and sequence. This is based on the assumption that different alleles at each site have different length and sequence. The similarity of sequences is calculated using `rapidfuzz` (available at `https://github.com/maxbachmann/rapidfuzz-cpp`). The two largest resulting subclusters (in terms of number of superstrings) are selected as haplotype candidates (considering the human genome is diploid). If only one subcluster is returned, it signifies a homozygous variant. `SVDSS` then computes a consensus sequence for each subcluster using Partial Order Alignment.

Assume that a cluster c spans the interval $G[s_c, e_c]$ of the reference genome G. Most strings of the cluster only partially cover this interval (i.e., they align to positions $[s, e]$ with $s_c \leq s < e \leq e_c$) while some others span the entire interval (i.e., they align to positions covering at least $[s_c, e_c]$). In order to perform a more accurate POA, `SVDSS` extends all the strings in a cluster to be of the same length. Therefore, `SVDSS` fills the gaps preceding or following a superstring using the reference genome. For instance, if a superstring S aligns to $[s, e]$ with $s_c < s < e < e_c$, then the resulting sequence will be $G[s_G, s-1] + S + G[e+1, e_G]$ (where $+$ is the string concatenation operator). The main goal of this extension is to summarize the information

contained in a cluster and to minimize the difference between the superstrings coming from different reads. The extended superstrings in each subcluster are then aligned to each other using `abPOA` [67] to generate a consensus (Figure 1 3B).

Finally, each POA consensus sequence is realigned locally to the reference genome window corresponding to its cluster using `parasail` [68]. The alignment's CIGAR information is analyzed to call and detect insertion and deletion SVs (Figure 1 3C). A weight is assigned to each SV prediction based on the number of superstrings that support it. A higher support indicates a more confident call. By default, we filter out SV calls having less than 4 supporting superstrings. The confidence threshold can also be set at runtime using the `--min-cluster-weight` option.

4.3.3 SV Chain Filtering

Reads originating from loci in repetitive parts of the genome such as STRs may map to slightly different coordinates due to the similarity of the local sequence. This will result in multiple clusters (relatively close to each other) and multiple SV calls for the same variant but at slightly different positions. To reduce the number of false positives and eliminate such redundant calls, we perform a "chain-filtering" post-processing step. This step sorts all predicted SVs based on coordinates and filters out consecutive SVs of the same type with similar sizes, keeping only the one with the highest number of supporting superstrings.

4.4 Implementation details

As a result of its many steps and the complexity of extraction SFS, `SVDSS` is more compute-intensive than other SV discovery methods, yet remains fast due to heavy optimization and deep parallelization. In this section we elaborate on the performance of each of the steps and compare our runtime to other methods.

The FMD-index creation and querying are handled internally by the FMD implementation from [65]. FMD-index creation for the GRC38 reference genome takes around 30 minutes on 16 cores. The index can be reused for any number of samples so its creation is a one-time expense.

Read smoothing is an IO-intensive step and benefits significantly from enabling the multithreaded BAM decoding functionality built into `htslib` [69] by setting the `bgzf_mt` flag when opening a BAM file. To further improve BAM decompression performance, we require that `htslib` is built with `libdeflate` in place of the default BAM decoder. For HiFi data at 30x coverage the smoothing algorithm takes about 15 minutes to run on 16 cores.

SFS extraction is the most computationally intensive step and takes about 45 minutes on 16 threads for the CHM13 HiFi data. Finally, the SV calling steps is very fast and takes less than 8 minutes to run despite the computational load of POA and local alignment. Overall, the runtime of the `SVDSS` pipeline is less than 70 minutes for a high-coverage HiFi sample on 16 cores, excluding index creation time. In comparison, the fastest SV caller was `cuteSV`, taking 5 minutes, and the slowest was `sniffles`, taking upwards of three hours. The remaining method `debreak`, `pbsv` and `SVIM` each took between 90 to 100 minutes to run.

All tools needed less than 64 GB of memory with `SVDSS` peaking at 34GB of memory during the SV calling stage. Our SFS extraction and smoothing stages each use constant memory, however the SV calling stage uses the most memory due to simultaneous handling of several (depending on the number of threads) POA graphs and local alignment dynamic programming tables in memory.

Data Availability All described datasets are publicly available through the corresponding repositories. In our experimental evaluation we used data publicly available at:

- GRCh38 reference genome: `https://hgdownload.cse.ucsc.edu/goldenpath/hg38/bigZips/hg38.fa.gz`
- GRCh37 reference genome: `ftp://ftp-trace.ncbi.nih.gov/1000genomes/ftp/technical/reference/phase2_reference_assembly_sequence/hs37d5.fa.gz`
- HG002 PacBio HiFi data: `https://storage.googleapis.com/brain-genomics-public/research/deepconsensus/publication/deepconsensus_predictions/hg002_15kb/two_smrt_cells/HG002_15kb_222723_002822_2fl_DC_hifi_reads.fastq`
- HG002 assembly: `https://console.cloud.google.com/storage/browser/brain-genomics-public/research/deepconsensus/publication/analysis/genome_assembly/hg002_15kb/two_smrt_cells/dc`
- CMRG callset: `https://ftp-trace.ncbi.nlm.nih.gov/ReferenceSamples/giab/release/AshkenazimTrio/HG002_NA24385_son/CMRG_v1.00/GRCh38/StructuralVariant/`
- HG007 PacBio HiFi data: `https://storage.googleapis.com/brain-genomics-public/research/deepconsensus/publication/deepconsensus_predictions/hg007_15kb/three_smrt_cells/HG007_230654_115437_2fl_DC_hifi_reads.fastq`
- HG007 assembly: `https://console.cloud.google.com/storage/browser/brain-genomics-public/research/deepconsensus/publication/analysis/genome_assembly/hg007_15kb/two_smrt_cells/dc`
- CHM13 PacBio HiFi data: `https://github.com/marbl/CHM13#hifi-data`
- CHM13 T2T assembly v1.1: `https://s3-us-west-2.amazonaws.com/human-pangenomics/T2T/CHM13/assemblies/chm13.draft_v1.1.fasta.gz`

The three callset built using `dipcall` are available at `https://github.com/ldenti/SVDSS-experiments`.

Code Availability `SVDSS` is open source and publicly available at `https://github.com/Parsoa/SVDSS`. Scripts to reproduce the experimental evaluations described in the manuscript are available at `https://github.com/ldenti/SVDSS-experiments`. Other software tools used in the study are either referenced or provided as links here: `pbmm2` (`https://github.com/PacificBiosciences/pbmm2`) and `pbsv` (`https://github.com/PacificBiosciences/pbsv`).

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

Methods-only Bibliography

[65] Heng Li. Exploring single-sample SNP and INDEL calling with whole-genome de novo assembly. *Bioinformatics*, 28(14):1838–1844, 05 2012.

[66] Peter Edge and Vikas Bansal. Longshot enables accurate variant calling in diploid genomes from single-molecule long read sequencing. *Nature communications*, 10(1):1–10, 2019.

[67] Yan Gao, Yongzhuang Liu, Yanmei Ma, Bo Liu, Yadong Wang, and Yi Xing. abPOA: an SIMD-based C library for fast partial order alignment using adaptive band. *Bioinformatics*, 37(15):2209–2211, 2021.

[68] parasail: Simd c library for global, semi-global, and local pairwise sequence alignments.

[69] James K Bonfield, John Marshall, Petr Danecek, Heng Li, Valeriu Ohan, Andrew Whitwham, Thomas Keane, and Robert M Davies. Htslib: C library for reading/writing high-throughput sequencing data. *Gigascience*, 10(2):giab007, 2021.

SVDSS: Structural Variation Discovery in hard-to-call genomic regions using Sample-specific Strings from accurate long-reads

Supplementary Information

Luca Denti, Parsoa Khorsand, Paola Bonizzoni, Fereydoun Hormozdiari, Rayan Chikhi

A Callset construction

Similarly to the recent GIAB benchmark [WOH+22], to build our own callsets for the three considered individuals (HG002, HG007, and CHM13), we used the assembly-to-assembly SV caller `dipcall` [LBF+18].

First, we downloaded the HG002 and HG007 assemblies provided by `DeepConsensus` [BCS+21] and the CHM13 assembly v1.1 available at the T2T github repository [1]. Then, following the `dipcall` (v0.3) procedure, we aligned the assembly to the GRCh38 reference genome and then called SVs with `dipcall`. The command we used to run the `dipcall` pipeline is:

```
run-dipcall -x {input.bed} {params.prefix} {input.fa} {input.hap1} {input.hap2}
```

B Advantages of assembly-based SV callsets for caller comparison

The HG002 individual has been extensively analyzed by the GIAB project [ZHO+20] to produce a SV callset, that we will refer to as the "GIAB v0.6 callset". To perform a more thorough experimental analysis, we also validated all tools against this callset. The goal of this analysis was twofold: to assess callers' accuracy against a different truth set and to assess the quality of the truth set built using `dipcall` and used in our main experimental evaluation.

Since the GIAB v0.6 callset is against the `GRCh37` reference genome, we reran all tools and analyses on `GRCh37`. We aligned the input sample using `pbmm2` and then we called SVs using our pipeline, the 4 state-of-the-art callers (`pbsv`, `cuteSV` `sniffles` and `SVIM`), and the more recent `debreak`. We also generated a SV callset using `dipcall` against the `GRCh37` reference. Remarkably on `GRCh37` `dipcall` called less variations (24,760 on `GRCh37` vs 25,114 on `GRCh38`). We then evaluated each caller accuracy using `Truvari` w.r.t. the GIAB v0.6 callset and the `dipcall` callset. To perform a more exhaustive analysis, we also compared the two truthsets using `Truvari`, i.e., we compared the GIAB v0.6 callset against the `dipcall` callset and viceversa. Table S1 reports the results of this analysis. As done in our main evaluation, we report the accuracy of each caller for the entire genome, and both Tier 1 and extended Tier 2 regions.

Regardless of the regions of the genome considered, `SVDSS` once again produced the most accurate calls among the tested tools when evaluating their accuracy against the `dipcall` callset whereas `debreak` achieved the best accuracy when evaluating against the GIAB v0.6 callset. We note that `debreak` achieved the highest precision, whereas `pbsv` the best recall using GIAB v0.6 callset. This was expected since `pbsv` is one of the tools used to create the GIAB v0.6 callset [ZHO+20], hence that callset may be biased towards `pbsv` calls.

Surprisingly, all tools achieved very low precision ($< 20\%$) w.r.t. GIAB v0.6 callset on hard-to-analyze regions of the genome (i.e., the extended Tier 2 regions). This suggests that the GIAB v0.6 callset may not be considered a complete truth set in those regions. To corroborate this claim, we compared the `dipcall` callset against the GIAB v0.6 callset. Since `dipcall` calls SVs directly from the assemblies, its callset should be considered to be less biased. Yet `dipcall` achieved a very low precision w.r.t. GIAB v0.6 callset. Moreover, on the other hand, the GIAB v0.6 callset achieved very low recall when comparing it against the `dipcall` callset.

We therefore compared the SVs density and size distribution of the two considered callsets. Figure S1 reports the results of this analysis. Overall, `dipcall` reported twice as much SVs as GIAB v0.6 (24,586 vs 12,737) and the difference is more evident if comparing the SVs falling in the Extended Tier 2 regions. In

[1] `https://github.com/marbl/CHM13`

those regions, `dipcall` called 13,703 SVs whereas GIAB v0.6 reported only 3,038 variations. The difference is less pronounced on Tier 1 regions, where `dipcall` called 11,007 SVs against the 9,641 reported by GIAB v0.6.

We finally investigated why our pipeline obtains higher recall w.r.t. other approaches when comparing against `dipcall` callset whereas it achieves a bit lower recall when comparing against the GIAB v0.6 callset. Once again, the reason behind these results needs to be sought in the presence of heterozygous non-reference SVs. Indeed, as shown in Figure S2, the `dipcall` truthset contains SVs that exhibit two different alleles on the two haplotypes, whereas the GIAB v0.6 callset includes only SVs genotyped 0/1 or 1/1, i.e., SVs exhibiting a single allele. As expected, `SVDSS` called the highest number of heterozygous non-reference SVs from the `dipcall` truthset (i.e., 1/2 calls in the figure), followed by `pbsv`. On the other hand, on the GIAB v0.6 truthset, `pbsv` reported the highest number of true calls, followed by `debreak` and `SVDSS`.

At least two factors have resulted in relative improvement of recall for `SVDSS` in comparison to other approaches when evaluating against `dipcall` callset. First, our approach is less hindered by drawbacks of mapping-based approaches and as a result has a significantly better recall/precision for hard regions of genome (Extended Tier 2) in comparison to other approaches. This advantage is mostly captured when we compare against `dipcall` callsets. Second, `SVDSS` is capable to call higher number of heterozygous non-reference SVs from the `dipcall` truthset (i.e., 1/2 calls in the Figure S2) than other approaches. For example, `SVDSS` contributes to detect SVs that have different lengths on the two haplotypes of the sample, while mapping-based approaches attempt to create a consensus of the two haplotypes. This type of events is also mostly captured using `dipcall` callsets.

C Supplementary Figures and Tables

HG002							
		GIAB callset (12,737 SVs)			dipcall callset (24,760 SVs)		
Mode	Tool	P	R	F1	P	R	F1
Full Genome	SVDSS	46.0	93.6	61.7	88.3	**78.5**	**83.1**
	cuteSV	49.9	94.1	65.2	84.7	68.7	75.9
	pbsv	50.4	**95.3**	65.9	85.8	69.3	76.7
	sniffles	42.8	90.4	58.1	80.0	67.5	73.2
	SVIM	49.5	89.9	63.8	82.8	65.2	73.0
	debreak	**54.7**	93.7	**69.1**	**89.0**	67.7	76.9
	dipcall	45.5	88.7	60.2	100	100	100
	GIAB	100	100	100	88.3	45.4	60.0
Tier 1	SVDSS	88.3	96.5	92.2	95.0	**85.4**	**89.9**
	cuteSV	87.7	97.5	92.3	90.0	82.8	86.2
	pbsv	91.8	**98.2**	94.9	94.8	83.1	88.6
	sniffles	84.5	94.8	89.4	87.3	81.1	84.1
	SVIM	86.6	94.8	90.5	89.5	81.2	85.1
	debreak	**94.4**	97.4	**95.9**	**96.6**	82.5	89.0
	dipcall	81.5	93.2	86.9	100	100	100
	GIAB	100	100	100	93.0	81.4	86.8
Extended Tier 2	SVDSS	16.7	84.1	27.9	**82.6**	**72.9**	**77.4**
	cuteSV	18.9	82.5	30.8	79.1	57.1	66.3
	pbsv	18.9	**85.3**	30.9	77.2	57.9	66.2
	sniffles	14.7	75.7	24.6	74.4	56.4	64.2
	SVIM	17.8	73.4	28.7	75.6	52.1	61.7
	debreak	**20.9**	81.0	**33.2**	81.2	55.5	65.9
	dipcall	16.3	73.9	26.8	100	100	100
	GIAB	100	100	100	73.3	16.2	26.6

Table S1: Comparison of performance of SVDSS and other methods on calling SVs w.r.t. GIAB v0.6 callset and dipcall callset. Results are shown in terms of Recall, Precision, and F1. We further include dipcall's performance in covering the GIAB v0.6 callset and GIAB v0.6's performance in covering dipcall callset. Note that for the latter analysis, we have perfect accuracy (100) when a callset is compared to itself.

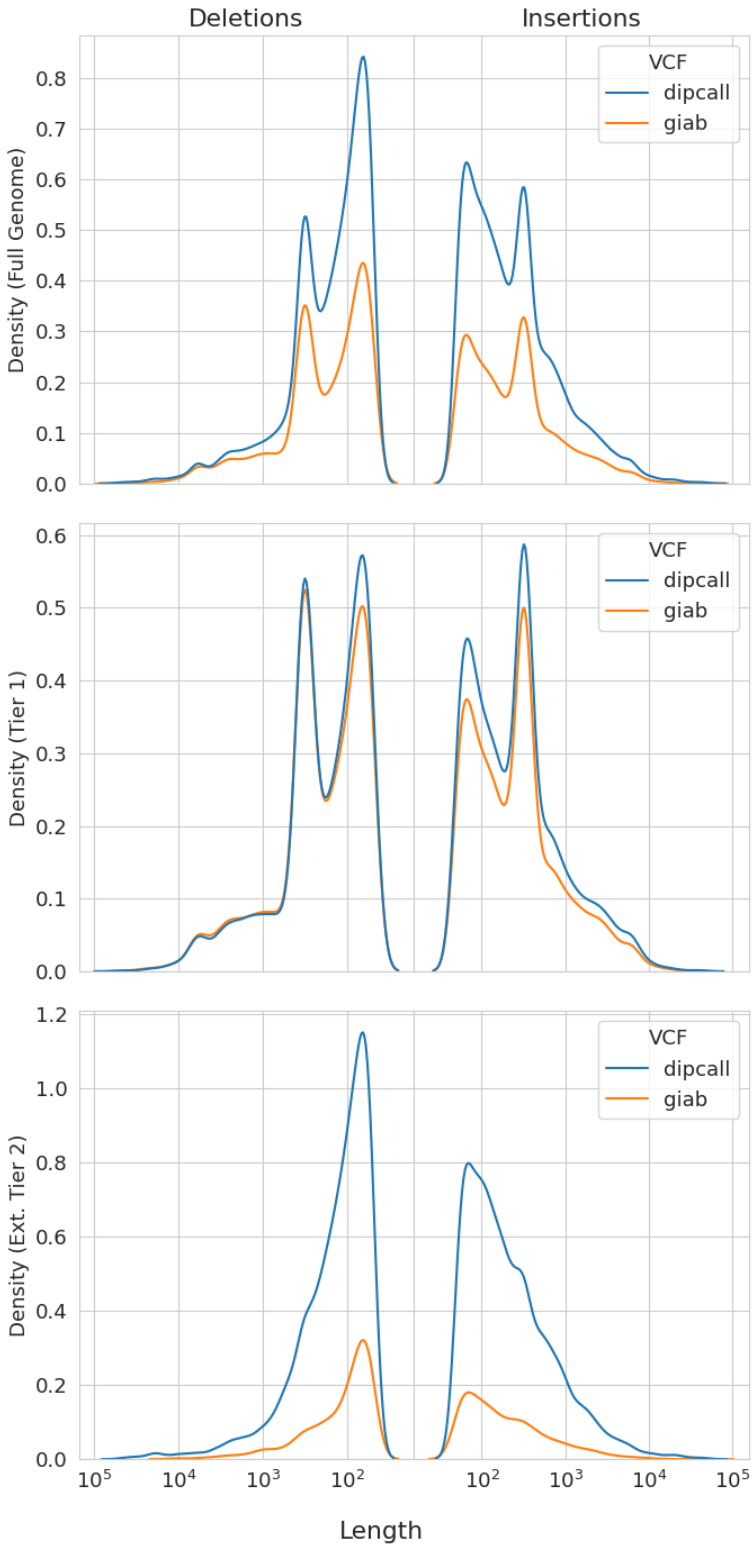


Figure S1: Size distribution of SVs reported by the GIAB project on HG002 and SVs called by `dipcall` considering the HG002 assembly provided by [BCS+21].

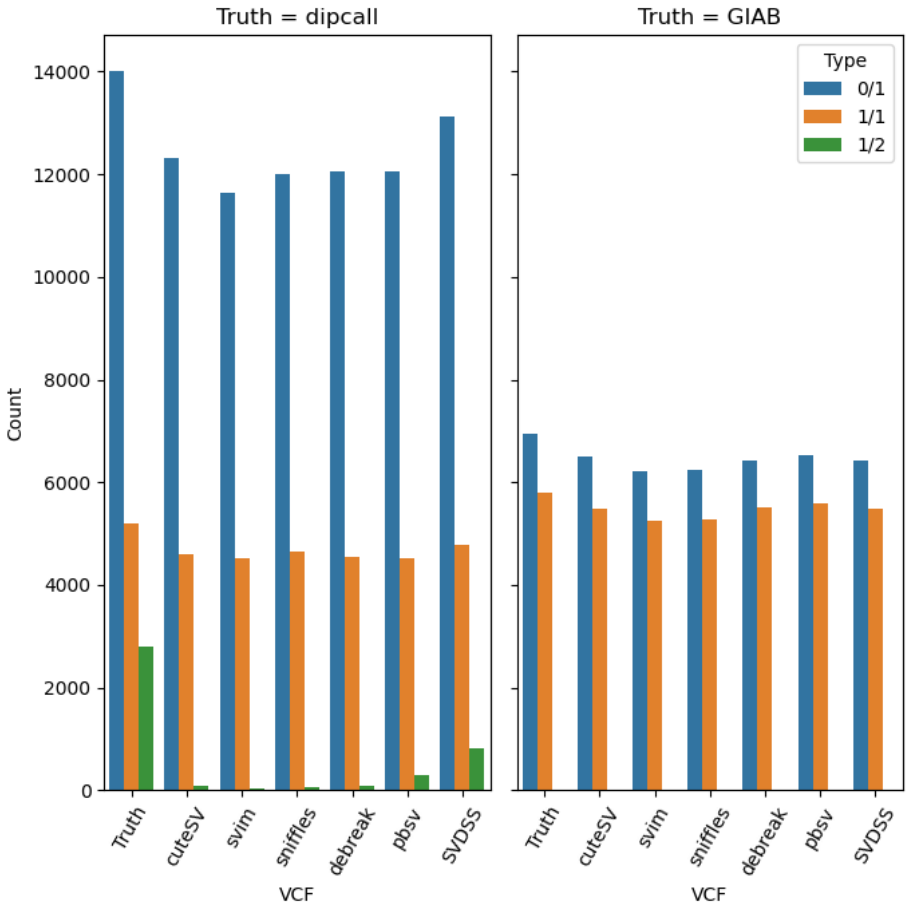


Figure S2: Number of True Positive calls reported by each tool and each truthset (i.e., GIAB and `dipcall`) on the HG002 individual. The calls are broken down into three different categories based on their true heterozygosity: 0/1 for reference/alternate calls, 1/1 for alternate calls with the same allele on both haplotypes, and 1/2 for alternate calls with different alleles on the two haplotypes.

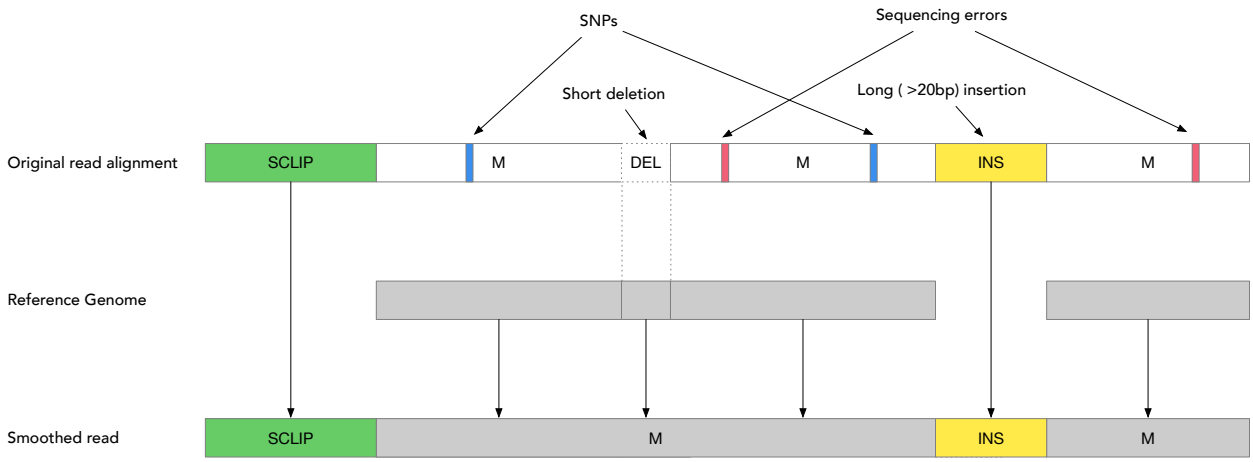
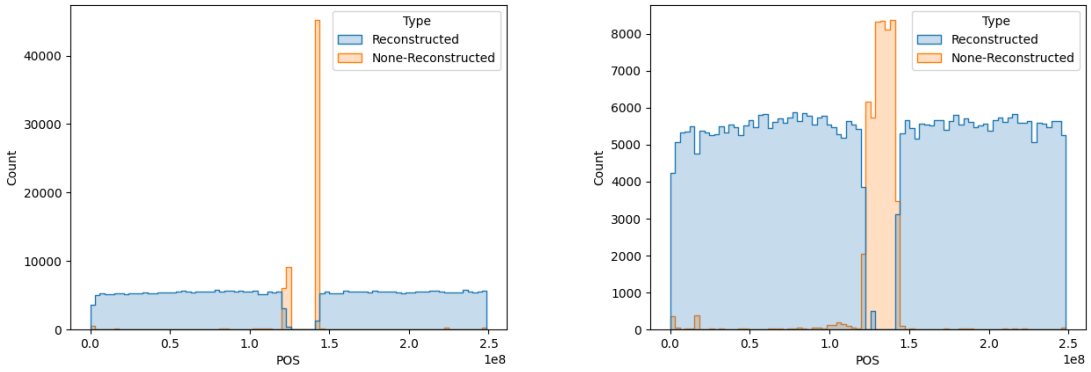


Figure S3: Illustration of the read smoothing algorithm. `M` alignment segments are smoothed from the reference genome, correcting SNPs (blue) and sequencing errors (red) in the process. Long indels are preserved while small ones (< 20bp) are removed. The small deletion is smoothed using the reference genome sequence while the large insertion (yellow) - potentially a SV - is carried over to the read. The soft-clipped section (green) is directly copied to the read.



(a) GRCh38 (b) CHM13

Figure S4: Distribution of read mapping locations on chr1. Almost all non-smoothed reads originate from centromeres of CHM13, however almost none can be properly mapped the GRCh38, resulting in an alignment gap around the centromere.

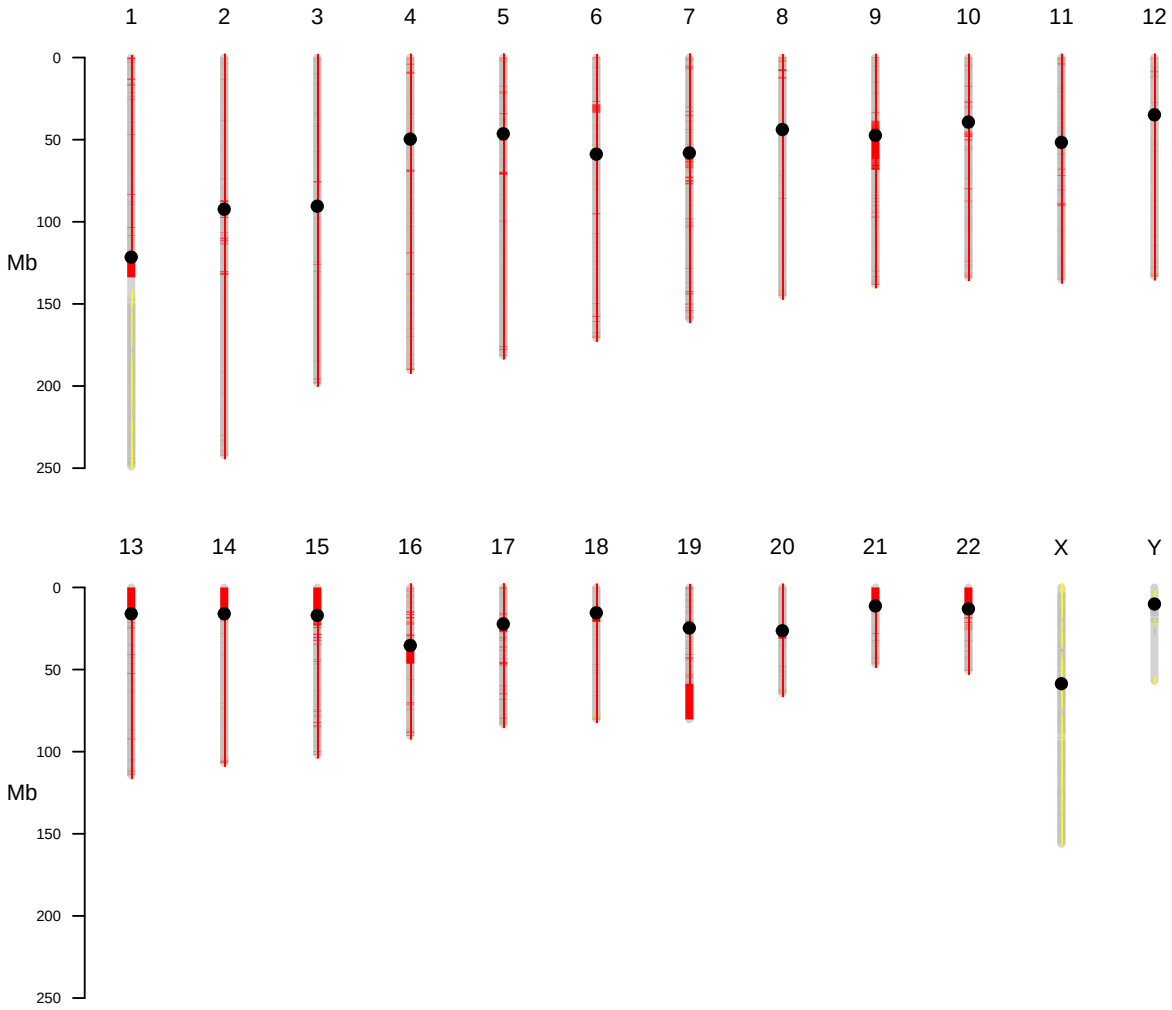


Figure S5: Tier 1 (grey areas) and extended tier 2 (red areas) regions in the reference genome GRCh38.

Tool	pbmm2			minimap2			ngmlr		
	P	R	F1	P	R	F1	P	R	F1
SVDSS	**90.1**	**76.5**	**82.7**	**91.9**	**79.3**	**85.1**	**93.0**	62.8	75.0
cuteSV	88.3	68.1	76.9	89.8	68.8	77.9	90.5	64.3	75.2
pbsv	84.9	68.6	75.9	85.0	68.7	76.0	84.9	**67.9**	**75.5**
sniffles	86.7	64.1	73.7	90.8	66.1	76.5	87.7	61.3	72.2
SVIM	84.9	64.7	73.4	87.2	65.7	74.9	81.9	64.4	72.1
debreak	**90.1**	64.2	75.0	91.3	64.8	75.8	86.0	60.8	71.2

Table S2: Comparison of performance of SVDSS and other methods when calling SVs on HG007 reads mapped with different aligners. Accuracy of each tool is reported in terms of Precision (P), Recall (R), and F-measure (F1). Results are whole-genome.

Tool	5x			10x			15x		
	P	R	F1	P	R	F1	P	R	F1
SVDSS	**92.9**	46.2	61.7	**91.6**	**70.3**	**79.5**	**90.1**	**76.5**	**82.7**
cuteSV	92.0	51.6	**66.1**	90.5	65.2	75.8	88.3	68.1	76.9
pbsv	58.6	**63.2**	60.8	66.0	68.1	67.0	84.9	68.6	75.9
sniffles	91.4	46.9	62.0	89.7	60.0	71.9	86.7	64.1	73.7
SVIM	87.4	49.9	63.5	86.3	62.3	72.4	84.9	64.7	73.4
debreak	92.2	26.0	40.6	91.2	52.3	66.5	**90.1**	64.2	75.0

Table S3: Comparison of performance of SVDSS and other methods on HG007 at different coverages. Accuracy of each tool is reported in terms of Precision (P), Recall (R), and F-measure (F1). Results are genome-wide.

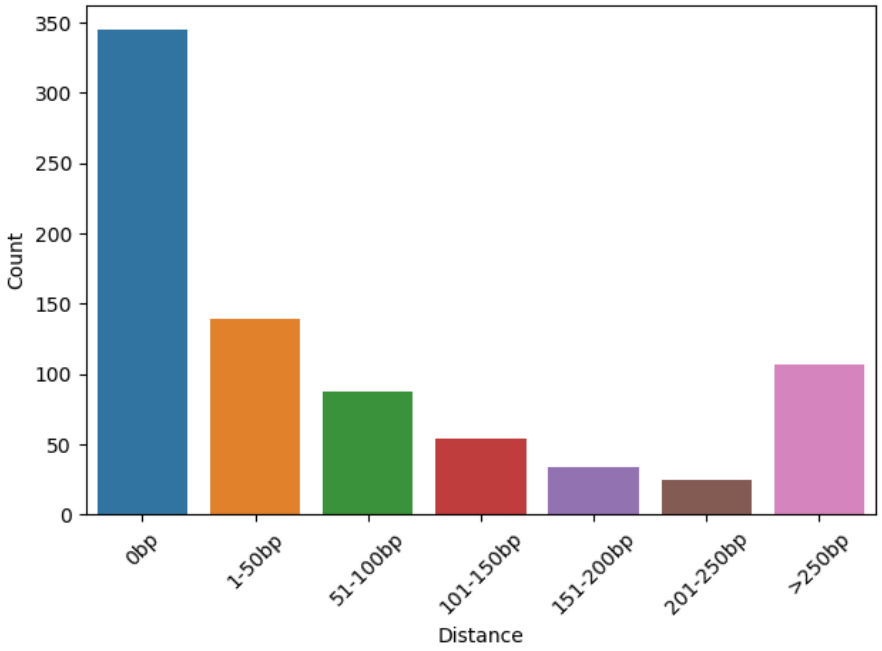


Figure S6: Bar plots showing the distance (in basepairs) of the SVs exclusively called by `SVDSS` and the closest SV (extended tier 2). A distance of 0bp means that the SV is a heterozygous non-reference SV (i.e., a SV with two alleles and genotyped 1/2).

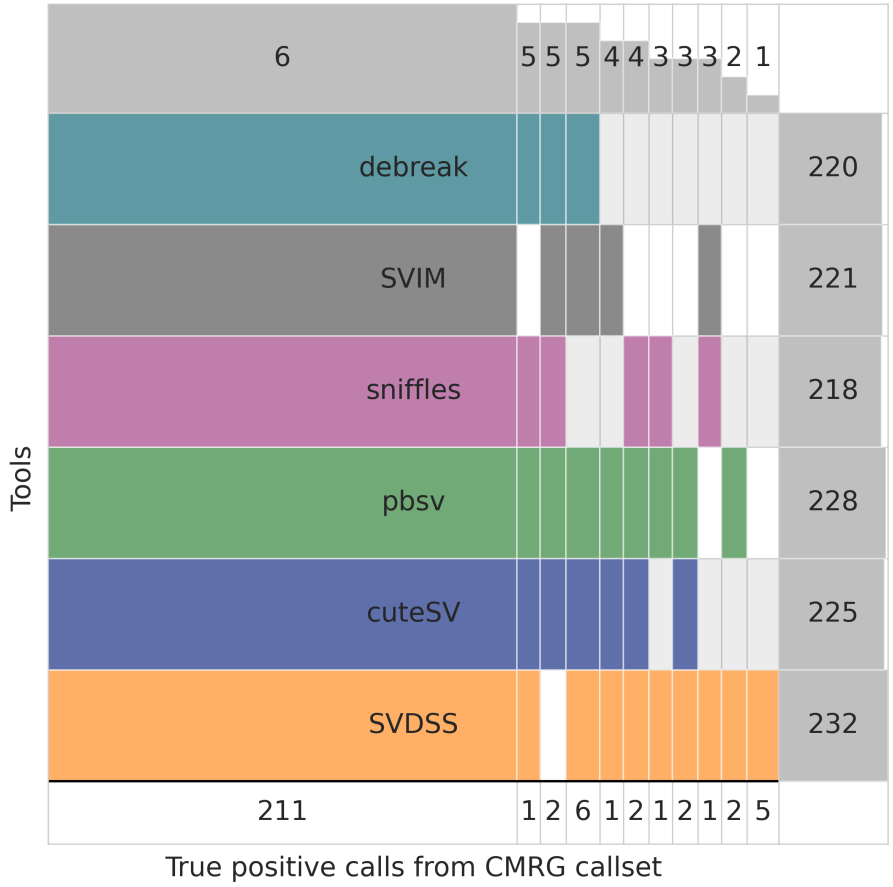


Figure S7: Supervenn diagram showing shared calls (True Positives) between different tools on the 273 medically-relevant genes considered in the CMRG callset.

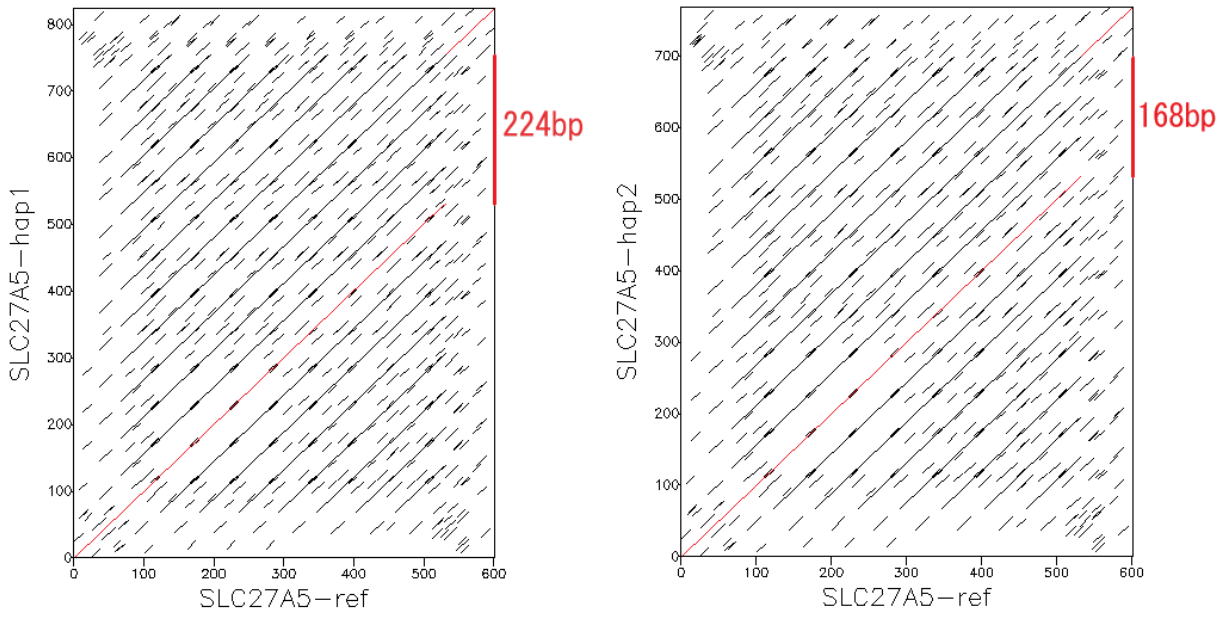


Figure S8: Dotplots of the alignment between the two HG002 high-quality haplotypes and the GRCh38 reference genome around the heterozygous SV falling in the medically-relevant gene *SLC27A5* (locus: `chr19:58487900-58488500`).

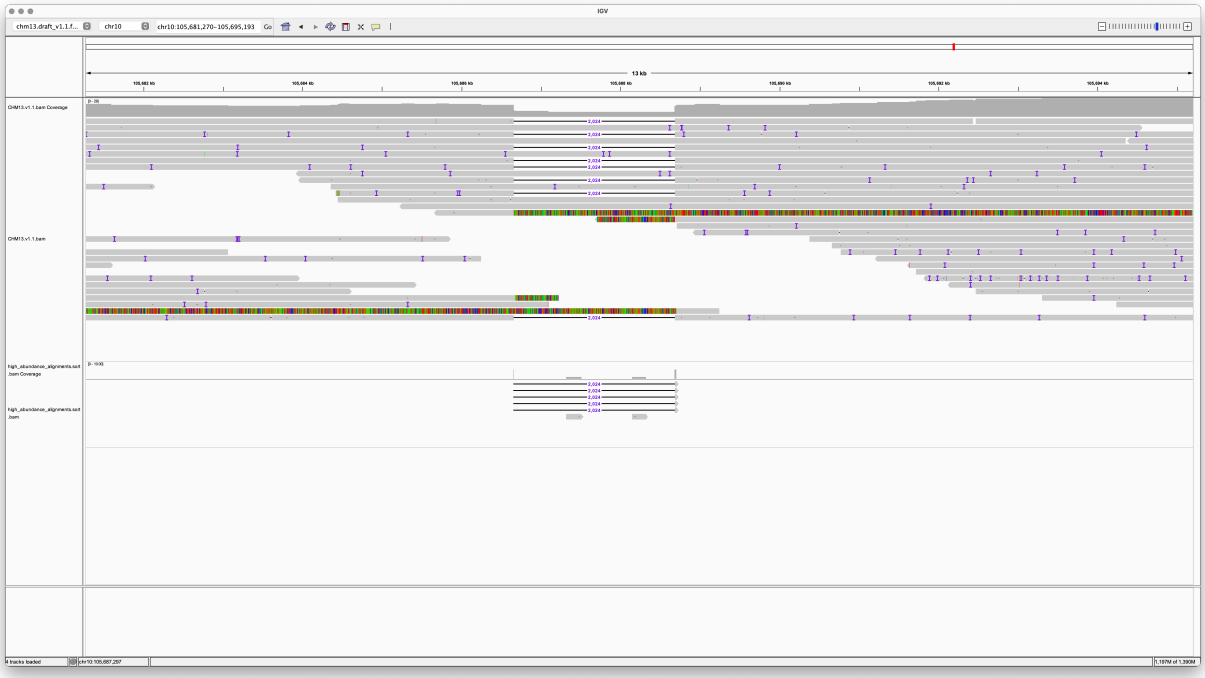


Figure S9: Example of heterozygous SV detected by our pipeline on CHM13 HiFi reads.

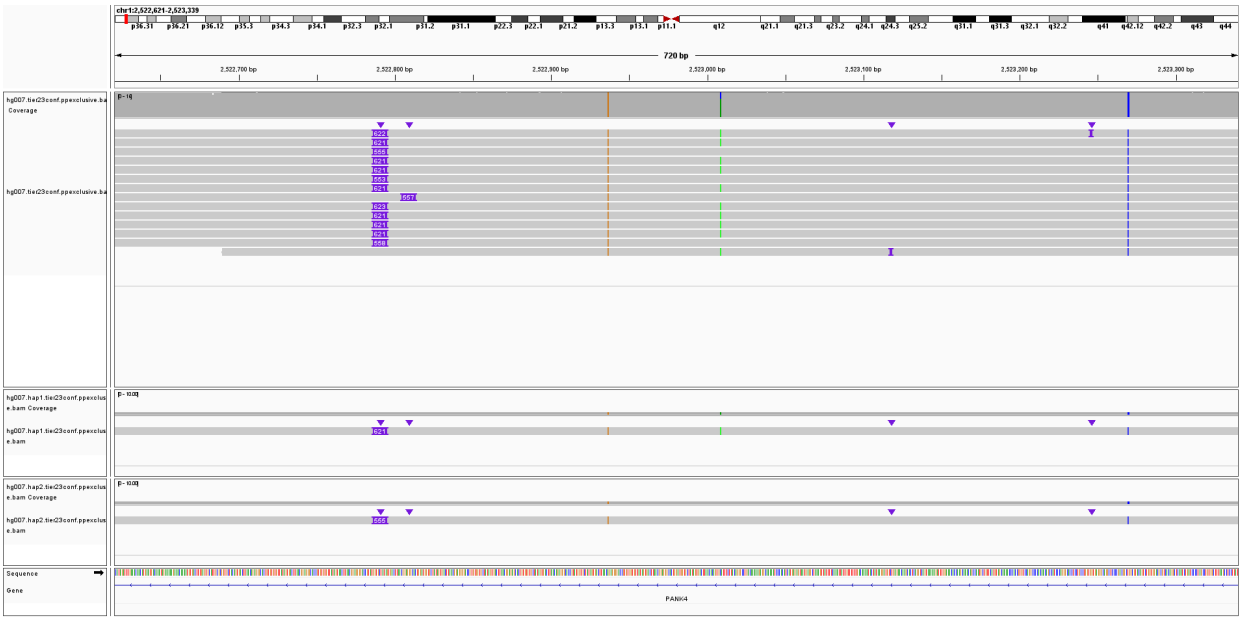


Figure S10: Heterozygous insertion in the HG007 sample (`chr1:2522791`). `dipcall` called two alleles of length 555 and 621. `SVDSS` agreed with `dipcall` correctly calling both alleles. `cuteSV`, `pbsv`, `sniffles`, `SVIM`, and `debreak`, instead, called just one allele of length 601, 621, 621, 602, and 601, respectively.

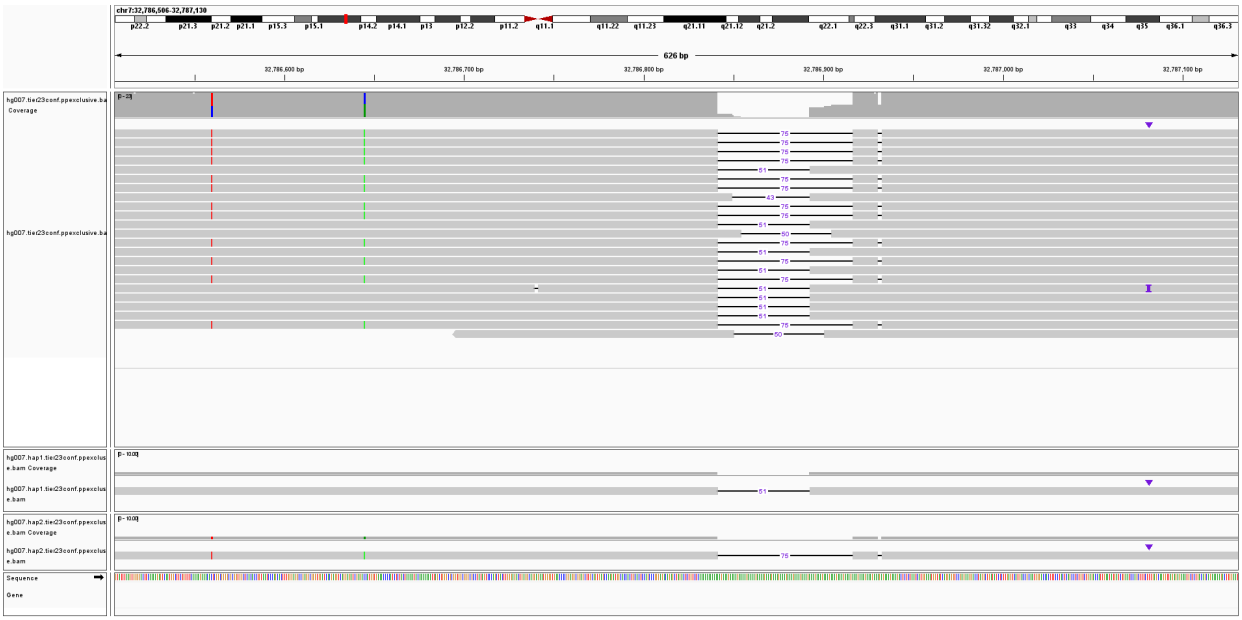


Figure S11: Heterozygous deletion in the HG007 sample (`chr7:32786841`). `dipcall` called two alleles of length 51 and 75. `SVDSS` agreed with `dipcall` correctly calling both alleles. `cuteSV`, `pbsv`, `sniffles`, `SVIM`, and `debreak`, instead, called just one allele of length 63, 75, 75, 63, and 63, respectively.

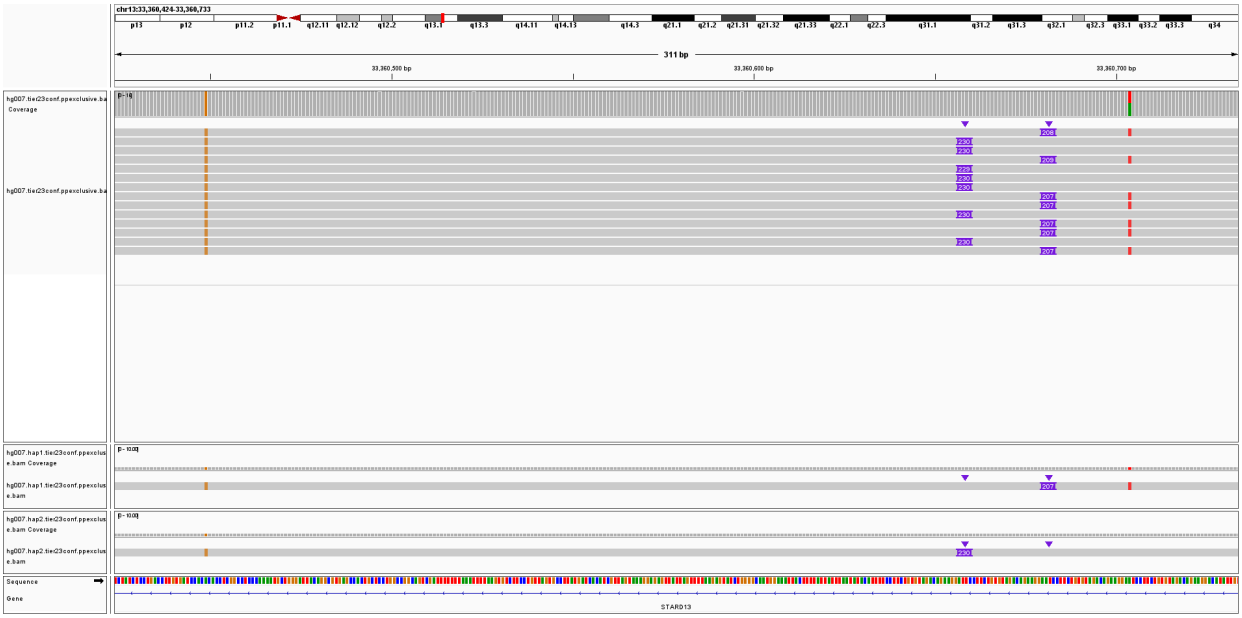


Figure S12: Two close insertion alleles in the HG007 sample (`chr13:33360658` and `chr13:33360681`). `dipcall` called two alleles of length 207 and 230. `SVDSS` agreed with `dipcall` correctly calling both alleles. `cuteSV`, `pbsv`, `sniffles`, `SVIM`, and `debreak`, instead, called just one allele of length 218, 230, 230, 218, and 218, respectively.

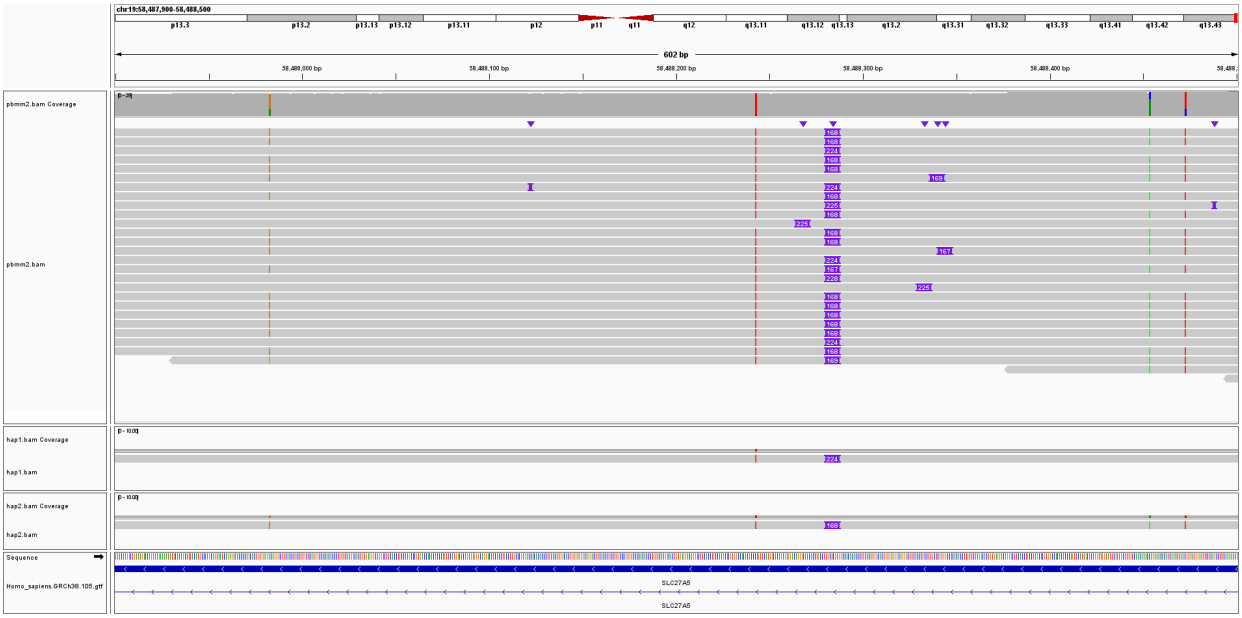


Figure S13: Full IGV image for the double insertion falling in the *SLC27A5* medically-relevant gene.

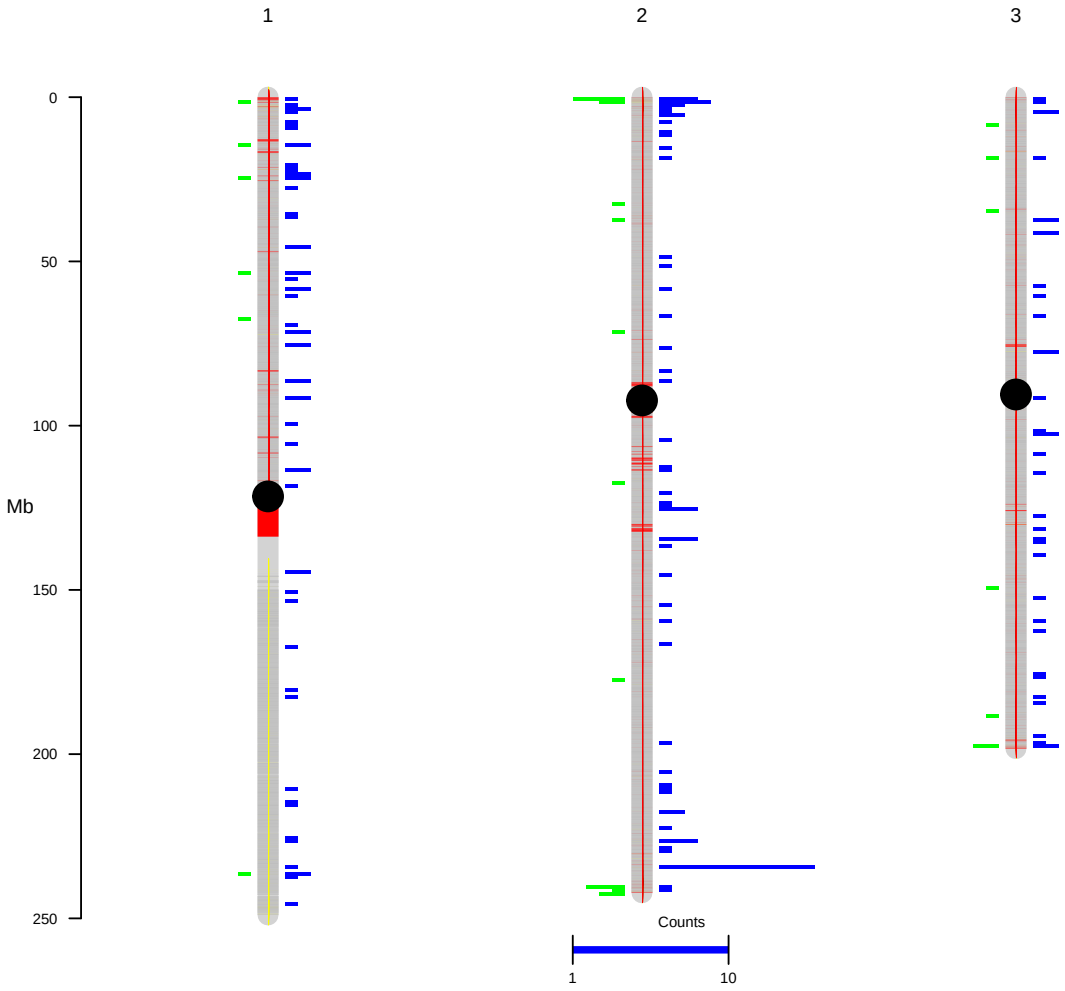


Figure S14: Comparison of SVDSS-specific and SVIM-specific Calls on Extended Tier 2 (red) on HG007. SVDSS (blue) has the highest number of specific calls on HG007 (739) while SVIM (green) has the second highest number of such calls (130).

D Versions and command lines

In this section we report the versions of the tools and the command lines we used in our experimental evaluation. We note that we limited this report to the tools we considered (i.e., mappers, callers, and truvari) and we did not report the pre/post -processing steps. For the full list of commands, we refer the reader to `https://github.com/ldenti/SVDSS-experiments`.

```
### MAPPERS ###
# minimap2, v2.22-r1101
minimap2 -ax map-hifi --MD --eqx -Y \
         -R '@RG\tID:{params.name}' \
         {input.fa} {input.fq} -o {output.sam}
# pbmm2, v1.7.0
pbmm2 align --preset CCS --sort \
      --rg '@RG\tID:{params.name}' --sample {params.name} \
      {input.fa} {input.fq} {output.bam}
# ngmlr, v0.2.7
ngmlr --rg-id {params.name} -r {input.fa} -q {input.fq} -o {output.sam}


### CALLERS ###
# pbsv, v2.6.2
pbsv discover --tandem-repeats {input.bed} {input.bam} {output.svsig}
pbsv call --ccs -t INS,DEL {input.fa} {input.svsig} {output.vcf}
# cuteSV, v1.0.11
cuteSV -s 2 --max_cluster_bias_INS 1000 --diff_ratio_merging_INS 0.9 \
       --max_cluster_bias_DEL 1000 --diff_ratio_merging_DEL 0.5 \
       {input.bam} {input.fa} {output.vcf} {params.wdir}
# svim, v1.4.2
svim alignment --min_sv_size 30 --cluster_max_distance 1.4 \
               --interspersed_duplications_as_insertions \
               --tandem_duplications_as_insertions \
               {output.odir} {input.bam} {input.fa}
# sniffles, v1.0.12
sniffles --ccs_reads -s 2 -l 30 -m {input.bam} -v {output.vcf} --genotype
# debreak, v1.0.2
debreak --rescue_large_ins --poa --min_size 30 --min_support 2 --aligner {params.aligner} \
        --ref {input.fa} --bam {input.bam} --outpath {params.odir}
# SVDSS, v1.0.0
SVDSS index --fastq {input.fa} --index {output.fmd}
SVDSS smooth --reference {input.fa} --bam {input.bam} --workdir {params.wd}
SVDSS search --index {input.fmd} --bam {input.bam} --workdir {params.wd} --assemble
SVDSS call --reference {input.fa} --bam {input.bam} --workdir {params.wd} \
           --batches {n} --min-cluster-weight 2


### TRUVARI ###
# v3.0.1
truvari bench --passonly -r 1000 -p 0.00 \
              --includebed {input.bed} -f {input.fa} \
              -c {input.vcf} -b {input.truth} -o {output}
```

Bibliography

[BCS+21] Gunjan Baid, Daniel E Cook, Kishwar Shafin, Taedong Yun, Felipe Llinares-Lopez, Quentin Berthet, Aaron M Wenger, William J Rowell, Maria Nattestad, Howard Yang, et al. DeepConsensus: Gap-Aware Sequence Transformers for Sequence Correction. *bioRxiv*, 2021.

[LBF+18] Heng Li, Jonathan M Bloom, Yossi Farjoun, Mark Fleharty, Laura Gauthier, Benjamin Neale, and Daniel MacArthur. A synthetic-diploid benchmark for accurate variant-calling evaluation. *Nature Methods*, 15(8):595–597, 2018.

[WOH+22] Justin Wagner, Nathan D. Olson, Lindsay Harris, Jennifer McDaniel, Haoyu Cheng, Arkarachai Fungtammasan, Yih-Chii Hwang, Richa Gupta, Aaron M. Wenger, William J. Rowell, Ziad M. Khan, Jesse Farek, Yiming Zhu, Aishwarya Pisupati, Medhat Mahmoud, Chunlin Xiao, Byunggil Yoo, Sayed Mohammad Ebrahim Sahraeian, Danny E. Miller, David Jáspez, José M. Lorenzo-Salazar, Adrián Muñoz-Barrera, Luis A. Rubio-Rodríguez, Carlos Flores, Giuseppe Narzisi, Uday Shanker Evani, Wayne E. Clarke, Joyce Lee, Christopher E. Mason, Stephen E. Lincoln, Karen H. Miga, Mark T. W. Ebbert, Alaina Shumate, Heng Li, Chen-Shan Chin, Justin M. Zook, and Fritz J. Sedlazeck. Curated variation benchmarks for challenging medically relevant autosomal genes. *Nature Biotechnology*, Feb 2022.

[ZHO+20] Justin M Zook, Nancy F Hansen, Nathan D Olson, Lesley Chapman, James C Mullikin, Chunlin Xiao, Stephen Sherry, Sergey Koren, Adam M Phillippy, Paul C Boutros, et al. A robust benchmark for detection of germline large deletions and insertions. *Nature Biotechnology*, 38(11):1347–1355, 2020.