

# 1 **Merfin: improved variant filtering and polishing via k-mer validation**

2  
3 Giulio Formenti<sup>\*†,1</sup>, Arang Rhie<sup>\*†,2</sup>, Brian P. Walenz<sup>2</sup>, Françoise Thibaud-Nissen<sup>3</sup>, Kishwar Shafin<sup>4</sup>,  
4 Sergey Koren<sup>2</sup>, Eugene W. Myers<sup>5</sup>, Erich D. Jarvis<sup>1</sup> and Adam M. Phillippy<sup>2</sup>

5 \* These authors contributed equally to the work.

6 † Co-corresponding authors.

7

8 <sup>1</sup> The Rockefeller University and Howard Hughes Medical Institute

9 <sup>2</sup> Genome Informatics Section, Computational and Statistical Genomics Branch, NHGRI, NIH, Bethesda,  
10 MD, USA

11 <sup>3</sup> NCBI, NLM, NIH, Bethesda, MD, USA

12 <sup>4</sup> UC Santa Cruz Genomics Institute, Santa Cruz, CA, USA

13 <sup>5</sup> Max Planck Institute of Molecular Cell Biology and Genetics, Dresden, Germany

14

## 15 **Abstract**

16 Read mapping and variant calling approaches have been widely used for accurate genotyping and  
17 improving consensus quality assembled from noisy long reads. Variant calling accuracy relies heavily on  
18 the read quality, the precision of the read mapping algorithm and variant caller, and the criteria adopted to  
19 filter the calls. However, it is impossible to define a single set of optimal parameters, as they vary  
20 depending on the quality of the read set, the variant caller of choice, and the quality of the unpolished  
21 assembly. To overcome this issue, we have devised a new tool called Merfin (*k-mer* based **fin**ishing tool),  
22 a k-mer based variant filtering algorithm for improved genotyping and polishing. Merfin evaluates the  
23 accuracy of a call based on expected k-mer multiplicity in the reads, independently of the quality of the  
24 read alignment and variant caller's internal score. Moreover, we introduce novel assembly quality and  
25 completeness metrics that account for the expected genomic copy numbers. Merfin significantly increased  
26 the precision of a variant call and reduced frameshift errors when applied to PacBio HiFi, PacBio CLR, or  
27 Nanopore long read based assemblies. We demonstrate the utility while polishing the first complete  
28 human genome, a fully phased human genome, and non-human high-quality genomes.

## 1 Introduction

2 Accurate variant calling has been a challenge in medical genomics, especially to achieve both high recall  
3 and precision in hard to measure regions<sup>1</sup>. The advent of Next Generation Sequencing (NGS) and long-  
4 read sequencing technologies streamlined variant calling algorithms<sup>2</sup>, which typically include: 1) aligning  
5 all reads to a reference genome; 2) performing variant calling from the alignment; and 3) filtering to  
6 remove false positives. The final outcome of variant calling relies heavily on the precision of this  
7 multistep procedure, which depends on: 1) the quality of the read set; 2) the precision of the read mapping  
8 algorithm; and 3) the precision of the variant caller in generating reliable calls<sup>3</sup>. To remove false  
9 positives, variant calls are often hard filtered using heuristics such as requiring a minimum coverage  
10 support, genotype quality, or other internal quality scores<sup>2</sup>. However, no consensus on the optimal  
11 parameters has been established, and the best parameters vary depending on the sequencing technology  
12 used. Therefore, the accuracy of a variant often corresponds to the theoretical limit of the algorithms and  
13 the parameters employed, and not the theoretical limit given the quality of the supporting raw data.

14 In parallel, new sequencing technologies greatly expanded our genome assembly toolkit. While the short  
15 read assemblies stumbled resolving repetitive regions<sup>4</sup>, long reads have considerably improved the  
16 contiguity of genome assemblies<sup>5</sup>. However, reduced consensus accuracy has been progressively  
17 acknowledged due to the lower base calling accuracy in long reads until the more recent PacBio Hi-  
18 Fidelity (HiFi) reads became available<sup>6</sup>. Still, lower base call accuracy remains even in HiFi reads for  
19 simple repeat sequences, particularly homopolymers<sup>7,8</sup>. Reduced consensus accuracy has detrimental  
20 impacts on many downstream analyses, such as gene annotation, which requires an accurate consensus to  
21 predict the correct coding sequence<sup>7</sup>. To mitigate this issue, “polishing” tools have been developed, such  
22 as Pilon, Arrow, Racon and Medaka<sup>9-11</sup>, while established variant calling tools such as GATK, Freebayes,  
23 DeepVariant<sup>12-14</sup> have been repurposed to detect errors and find candidate corrections. Unlike re-  
24 sequencing based methods, the assembly from the same genome is used as a reference for polishing, and  
25 thus all homozygous variants suggest corrections to be made. Once corrections are collected, the  
26 consensus can be updated using tools such as Bcftools<sup>15</sup>. The process is usually repeated with different  
27 read sets (e.g. long and short reads), until the accuracy of the consensus reaches a set standard.

28 Consensus accuracy (hereby noted as QV for simplicity) has been historically measured from the variant  
29 calling process as described above, however, bearing biases caused from mapping or variant calling. In  
30 our previous work, we presented Merqury<sup>16</sup>, an alignment-free approach to estimate base-level QV using  
31  $k$ -mers (genomic substrings of length  $k$ ). In Merqury,  $k$ -mers found only in the assembly and not in the  
32 reads are considered as errors, disregarding the expected copy number. As a result, overly represented  $k$ -  
33 mers from sequence expansion (i.e. false duplications) in the assembly are considered as correct bases.  
34 Merqury also presents a completeness metric from the portion of  $k$ -mers found in the assembly from a  
35 given reliable  $k$ -mer set in reads. However, this  $k$ -mer completeness metric does not account for the  $k$ -mer  
36 multiplicity in the reads, limiting the scope to be measured in the non-repetitive  $k$ -mer space. As a result,  
37 any two assemblies with the same distinct  $k$ -mers will score the same completeness metric, regardless of  
38 one having higher sequence collapses or expansions.

39 Ideally, the sequence of an error-free and complete genome assembly is in perfect agreement with the  
40 sequence data, assuming that genomic DNA is randomly sampled and sequenced with negligible  
41 sequencing biases. Therefore, any changes introduced during polishing should improve the assembly-read

1 agreement. This principle has been widely used to visually evaluate genome assembly copy number  
2 spectrum (e.g. copy-number spectrum analysis<sup>16,17</sup>), and more recently, used to detect errors and improve  
3 read alignment<sup>18-20</sup>. However, none of the evaluation metrics or polishing methods have fully utilized this  
4 assembly-read agreement to date.

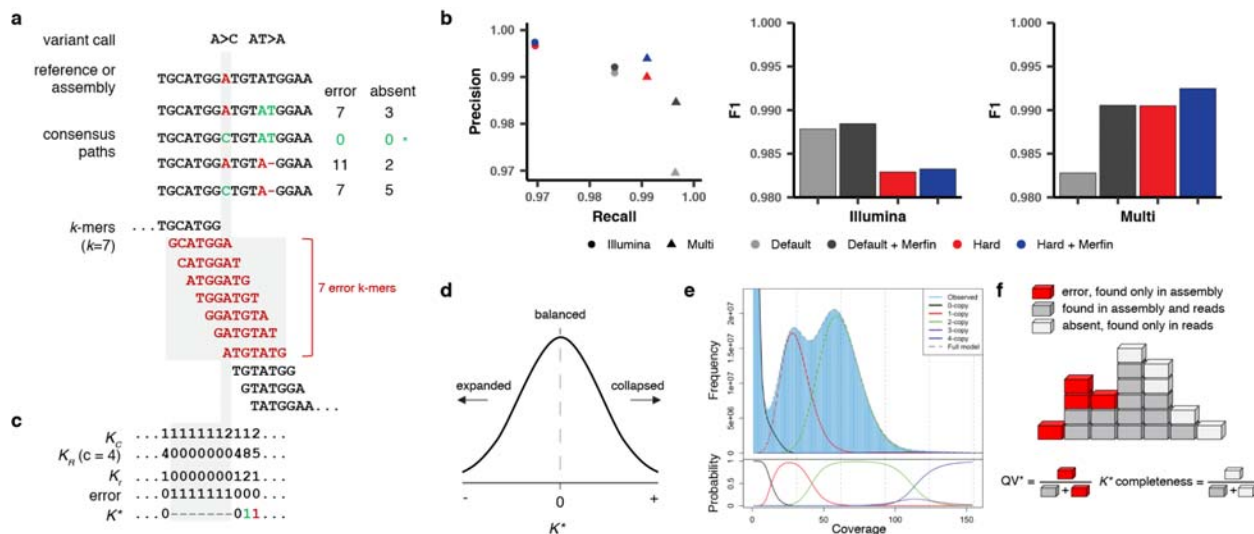
5 Here, we first introduce a  $k$ -mer based filtering approach applicable on genomic variant calls, which  
6 achieved higher F1 scores compared to parameter based hard-filtering methods. Next, we propose a  
7 revised QV and completeness score that accounts for the expected sequence copy number given a  $k$ -mer  
8 frequency, driven by our refined  $K^*$  definition<sup>21</sup> for genome assembly evaluation. Our  $K^*$  enables the  
9 detection of collapses and expansions, and significantly improves the QV when used to filter variants for  
10 polishing. We applied this approach to polish and evaluate the most complete HiFi-based assembly of  
11 human CHM13<sup>22-25</sup>, a Nanopore-based trio assembly from the Human Pangenome Reference Consortium  
12 (benchmark paper in preparation), and three non-human reference genomes from the Vertebrate Genomes  
13 Project (VGP)<sup>5</sup>, all resulting in significantly higher consensus accuracy and annotation quality. This  
14 approach is implemented as *Merfin* ( $k$ -mer based finishing tool) and is publicly available.

## 15 **Results**

### 16 **Variant call filtering for higher precision**

17 A reference genome (i.e. GRCh38) with its sequence replaced at all alternate variant calls can be  
18 considered as a “consensus” sequence and evaluated with  $k$ -mers. Unlike using a *de novo* assembled  
19 genome of the same individual as a reference, natural biological differences between the sequenced  
20 individual and the reference genome or the incomplete state of the reference (i.e. missing a segmental  
21 duplication) imposes challenges to reliably call variants. Nevertheless, it is possible to construct  
22 consensus paths from a variant or series of variants within  $k$  bps, and confirm its validity. We can score  
23 each path by the number of  $k$ -mers never found in the reads (error  $k$ -mer), and choose the best path to  
24 contain minimum error  $k$ -mers (**Fig. 1a**).

25 To test the validity of this filtering approach, we benchmarked against unfiltered (default) and hard-  
26 filtered variant calls submitted to precisionFDA challenge II, HG002<sup>1</sup>. The variants were called from  
27 Illumina reads or from multiple platforms (Illumina, PacBio HiFi, and ONT) using GRCh38 as the  
28 reference with GATK HaplotypeCaller. Hard-filtering was performed using the variant caller’s internal  
29 scores such as PASS, QD, MQ and QUAL. When comparing precision, recall, and F1 (harmonic mean of  
30 precision and recall) on a truth set of Chr. 20<sup>26</sup>, Merfin always achieved higher precision with minimal  
31 loss in recall when applied on both default and hard-filtered sets (**Fig. 1b**). The hard-filtered set had a  
32 higher precision, with the price of losing more true positives, resulting in a lower F1 score when  
33 compared to the default set. On the other hand, Merfin was able to remove additional false positives on  
34 the hard-filtered set.



**Fig. 1 | Algorithms and results used in Merfin.** **a**, Two variant calls and its potential consensus paths. The bases and k-mers in red are errors not found in the reads. The path with A>C has no error k-mers and gets chosen for genotyping (\*). For polishing, the average  $K^*$  gets computed in addition to the missing k-mers using the predicted absent k-mers. **b**, Precision, recall, and F1 from a benchmark on HG002 genotyping. Merfin always achieves higher precision and F1 scores compared to the hard-filtered approach with almost no loss in recall. Default, no filtering; Red, hard-filtering on default. **c**, K-mer frequency found in the consensus sequence ( $K_C$ ), reads ( $K_R$ ) with average coverage at 4 (c), expected copy number based on the corrected k-mer frequency ( $K_r = K_R / c$ ), and  $K^*$ . Positive and negative  $K^*$  values are colored in green and red. The highlighted region (gray) shows the same k-mers and values used to compute  $K^*$  as affected by the A base in the reference. If two alternatives bear the same number of missing k-mers the alternative with the  $K^*$  closest to zero is chosen. **d**,  $K^*$  distribution.  $K^*$  values deviated from 0 indicate collapsed (+) or expanded (-) k-mers in the assembly. **e**, Genomescope 2.0 k-mer frequency histogram with theoretical k-multiplicity curves (top) and probabilities (bottom) for 0, 1, 2, 3, and 4-copy k-mers, generated using the `--fitted_hist` option. Note that the 3-copy peak is fully contained in the 2 and 4-copy peaks. **f**, Diagram for estimating  $QV^*$  and completeness from k-mers. Each k-mer is a block colored by its state of presence. In the block tower, each column represents the identical k-mer with its state colored by its presence in the assembly, reads, or in both. Note the  $QV^*$  and  $K^*$  completeness is using all k-mers including their frequency.

## 19 Assembly evaluation

20 When a reference genome is constructed from the same individual, the k-mer multiplicity seen in the  
 21 reads is expected to match the reference. This assumption can be used for evaluating de novo assembled  
 22 genomes. In the following, we introduce our revised  $K^*$ , which we used to identify possible collapsed  
 23 and expanded regions in an assembly, and quantitative metrics for representing the copy-number  
 24 concordance and completeness of the assembly.

### 25 Identifying collapsed and expanded regions

26 The  $K^*$  metric was defined previously to detect identical collapsed repeats on each k-mer in the  
 27 assembly<sup>21</sup>. The method proposed  $K^* = K_R / K_C$ , where  $K_R$  is the frequency of a k-mer found in the reads;  
 28 and  $K_C$  is the frequency of a k-mer across the entire consensus sequence of the assembly. In regions with  
 29 no collapsed repeats,  $K^*$  will be equal to c, the average coverage of the sequenced reads. Here we revised  
 30 the  $K^*$  such that it evaluates both collapses and expansions. We propose  $K^* = (K_r - K_C) / \min(K_r, K_C)$ ,  
 31 where  $K_r$  is the expected copy number inferred from the reads (**Fig. 1c**). For a perfect genome assembly

1 and an unbiased read set,  $K^*$  is normally distributed with mean 0, and deviations from the mean reflect  
2 natural variation in the Poisson sampling process (**Fig. 1d**). Conversely, any significant deviation from  
3 the normal distribution can be interpreted either as a bias in the assembly (i.e. an assembly error) or a bias  
4 in the read set. In particular, a positive  $K^*$  implies that the assembly contains fewer copies of  $k$ -mers than  
5 suggested by the read set (collapsed), while negative  $K^*$  implies more copies in the assembly than  
6 suggested by the read set (expanded).

7 The  $K_r$  can be obtained with  $\lceil K_R / c \rceil$ , where  $c$  is the haploid (1-copy) peak of the  $k$ -mer distribution of  
8 the reads. Here we assume that rounding  $K_r$  is sufficient to account for the standard deviation associated  
9 with the Poisson process underlying read generation. While this is true in the case of a perfectly sampled  
10 sequencing set, the validity of this generalization is challenged in the presence of sampling bias,  
11 systematic error in the reads, and variable degrees of heterozygosity which prevents the clear distinction  
12 of each ploidy peak. To account for this uncertainty and improve the accuracy of the results, we modified  
13 Genomescope<sup>27</sup> to probabilistically infer  $K_r$  for each  $K_R$ , using the observed  $k$ -mer count distribution in  
14 the read set. If supplied, Merfin will use these probabilities for  $K_r \leq 4$ . (**Fig. 1e**).

#### 15 *QV\* estimation*

16 An average genome-wide QV accounting for excessive copy numbers (hereby defined as QV\*) can be  
17 obtained using  $\sum K_C - K_r$  as errors when  $K_C > K_r$  for all positions in the assembly (**Fig. 1f**). These  
18 excessive and error  $k$ -mers can be generalized as ‘errors’ and Phred-scale QV, obtained as implemented in  
19 Mercury<sup>16</sup> or YAK<sup>28</sup>.

#### 20 *Assembly completeness*

21 The sum of  $K_r - K_C$  (over all positions where  $K_r > K_C$ ) expresses absent  $k$ -mers that should be present in  
22 the assembly, and can be directly translated into a measure of assembly completeness as  $1 - \sum (K_r - K_C) /$   
23  $K_r$  (**Fig. 1f**). Importantly, contrary to other measures of assembly completeness based on a subset of the  $k$ -  
24 mers (e.g. relying only on the occurrence of distinct  $k$ -mers as implemented in Mercury<sup>16</sup>), Merfin uses all  
25  $k$ -mers, including its frequency, and computes the fraction of the expected total number of  $k$ -mers.

#### 26 **Sequence polishing**

27 The  $K^*$  becomes particularly useful in polishing. The impact of each correction or combination of  
28 corrections are assessed from a given variant call set (correction candidates) by comparing the change in  
29  $K^*$ -related metrics (**Fig. 1a,c**). In addition to the error  $k$ -mers collected in each predicted consensus path,  
30 we compute the consequent  $k$ -mer frequency change, and choose the correction only when it improves the  
31 assembly-read agreement. For example, when a suggestive correction (replacing AT with A as shown in  
32 **Fig. 1a**) introduces more error  $k$ -mers, it should not be used for polishing. Even when no error  $k$ -mers are  
33 introduced,  $K^*$  theoretically informs whether a path improves the assembly-read agreement in polishing.  
34 The current implementation evaluates each path independently, and thus only a local optimum is  
35 guaranteed. Moreover, variants within distance  $k$  are considered in all combinations, allowing Merfin to  
36 filter ambiguous variant calls. This approach is fully independent of the raw dataset employed. For  
37 instance, the assembly could be generated using long reads, and the calls evaluated using either short or  
38 long reads or both, taking advantage of the strengths of each sequencing platform, making accurate  
39 orthogonal validation possible, and maximizing the assembly-read agreement.

40

1 We first focus our evaluation of Merfin on the most complete assembly to date, generated from a nearly  
2 homozygous diploid CHM13hTERT (CHM13) cell line, simultaneously released by the Telomere-to-  
3 Telomere (T2T) Consortium<sup>22</sup>. We then provide an example of improved base calling accuracy in a  
4 Nanopore-based trio assembly from the Human Pangenome Reference Project (HPRC). We further  
5 extend the usage of Merfin to haploid and pseudo-haploid assemblies applying Merfin to three long-read  
6 assemblies (a fish, reptile, and bird) generated in the framework of the Vertebrate Genomes Project  
7 (VGP)<sup>5</sup>.

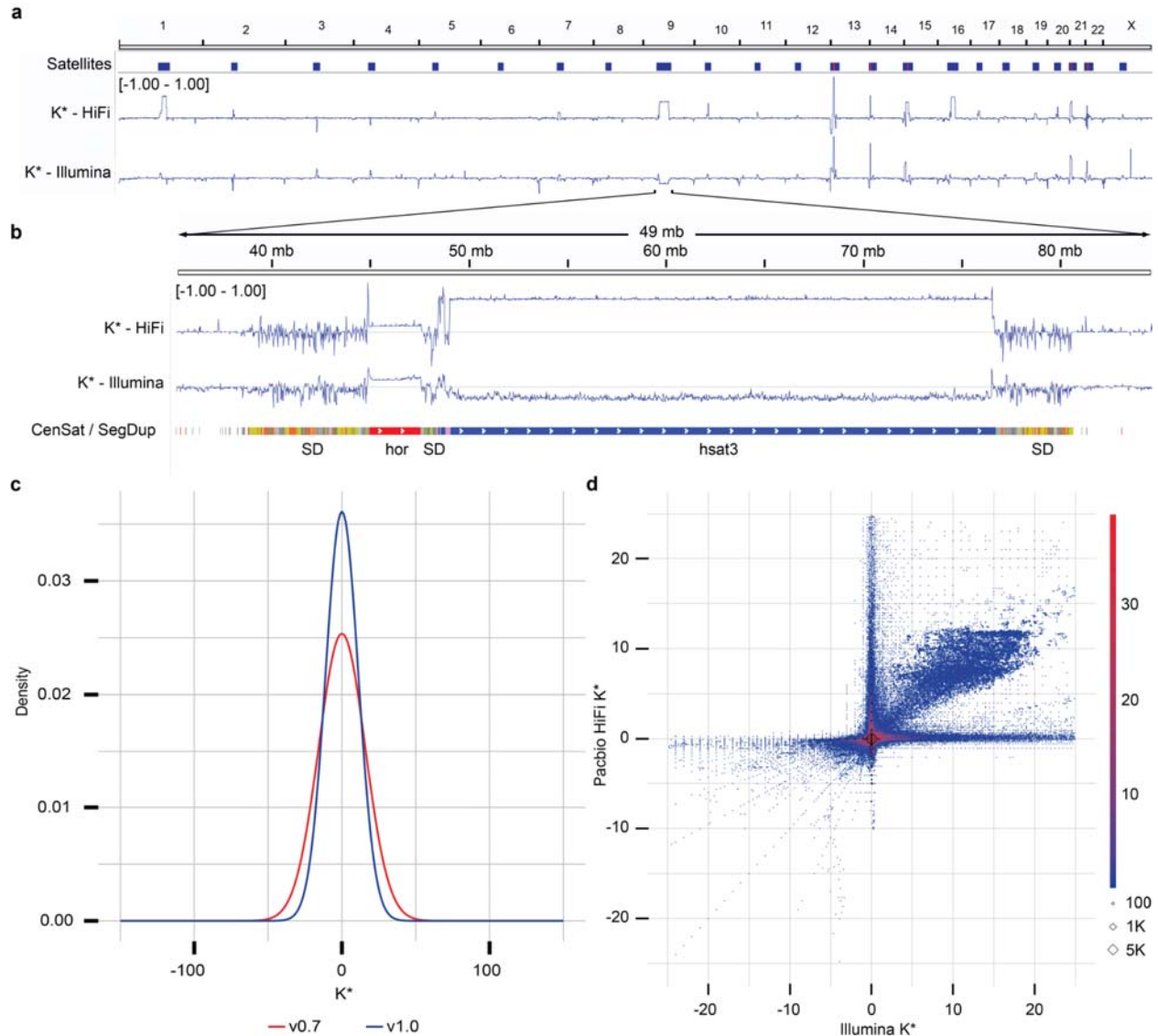
## 8 **Evaluating a complete human genome: T2T-CHM13**

9 The CHM13 cell line originates from a complete hydatidiform mole (46, XX), where both haplotypes are  
10 near-identical except for a few heterozygous variants that probably have their origin in the original mole  
11 tissue or have accumulated during cell passages<sup>29</sup>. This cell-line was used to generate the most complete  
12 high-quality human reference to date, resolving all centromeric and telomeric repeats and all segmental  
13 duplications and satellite arrays<sup>22,23</sup>. Notably, the T2T-CHM13v0.9 was polished from a variety of variant  
14 calls, with an earlier version of Merfin and improved the consensus accuracy of the final assembly<sup>25</sup>. We  
15 further evaluated candidate assemblies to identify collapses and expansions using Merfin using  $k$ -mers  
16 from HiFi and Illumina reads. We found that the T2T-CHM13v1.0 assembly shows a remarkable  
17 agreement with the raw data, with only a few regions having  $K^*$  significantly different from 0, coinciding  
18 with satellite repeats (**Fig. 2a**). Rather than being assembly errors, these disagreements were associated  
19 with context-dependent augmentation or depletion in HiFi and GC bias in Illumina<sup>22,25</sup>. Indeed,  $K^*$   
20 derived from HiFi and Illumina  $k$ -mers showed opposite behavior in some regions, i.e. the HSat3 of Chr.  
21 9 (**Fig. 2b**). These effects originating from sequencing biases were observed only on the highly repetitive  
22 regions of the genome.

23 Compared to a less complete and less accurate preliminary assembly, T2T-CHM13v0.7<sup>30</sup>, T2T-  
24 CHM13v1.0 had a higher agreement of the assembly with the  $k$ -mers derived from HiFi (**Fig. 2c**) and  
25 Illumina (**Supplementary Fig. 1**) reads. We found a general agreement in  $K^*$  between HiFi and Illumina  
26 PCR-free  $k$ -mers, including regions with sequencing bias common to the two technologies  
27 (**Supplementary Fig. 2**). In other cases, the direct comparison of the  $K^*$  computed from the two  
28 technologies highlighted technology-specific sequencing biases (**Fig. 2a,d**). At base resolution, the  $K^*$   
29 could distinguish regions with accurate consensus from base pair errors, small and large indels,  
30 heterozygous sites, and collapsed/expanded regions (**Supplementary Figs. 3a-b**).

31  
32 Both QV and QV\* measured with Merqury and Merfin improved from v0.7 to v0.9<sup>25</sup>, which involved a  
33 complete reassembly of the genome using HiFi reads and patches from v0.7 at GA-rich sequence  
34 dropouts in the HiFi reads (**Supplementary Table 1**). Merqury QV improved from v0.7 to v0.9, due to  
35 the dramatic decrease in error  $k$ -mers, however the Merfin QV\* did not change as the number of error  $k$ -  
36 mers is small compared to the number of overly-represented  $k$ -mers. Although the high amount of overly  
37 represented  $k$ -mers both in HiFi and Illumina reads may originate from sequencing biases, we argue that  
38 QV\* may be a more reliable metric, because it accounts for all expected  $k$ -mer copy numbers, reflecting  
39 the full extent of genome representation. The QV\* was also marginally influenced by the coverage  
40 through a titration experiment (**Supplementary Fig. 4, Supplementary Table 2**).

41



1

2 **Fig. 2 | CHM13 evaluation and polishing.** **a**, Genome-wide  $K^*$  for the CHM13 assembly v1.0. Satellites are

3 associated with repeat- and technology-specific biases. Yet to be resolved rDNA arrays (red) are highlighted by

4 positive  $K^*$ . **b**, Highlight of the centromeric satellite repeats (manuscript in preparation) and segmental

5 duplications<sup>23</sup> (orange most similar, yellow less, gray least) on chromosome 9. **c**, Genome-wide density distribution

6 of the  $K^*$  using HiFi  $k$ -mers. When the assembly is in agreement with the raw data, the  $K^*$  is normally distributed

7 with mean 0, and the smaller the standard deviation the higher the agreement. CHM13 v1.0 shows a less dispersed

8 distribution of the  $K^*$  compared to a less complete v0.7 assembly. **d**, Genome-wide comparison of the  $K^*$  computed

9 using HiFi vs Illumina  $k$ -mers on the CHM13 v1.0 assembly. Agreement between the assembly and the raw reads

10 supported by the two technologies is found around (0, 0). The upper right quadrant highlights where both HiFi vs

11 Illumina technologies suggest the presence of underrepresented  $k$ -mers that were mostly contributed from the un-

12 assembled rDNAs later resolved in v1.1<sup>22</sup>; the lower left quadrant highlights where both technologies suggest the

13 presence of overrepresented  $k$ -mers (with perfect agreement found on the diagonals). The axes correspond to regions

14 of substantial disagreement between the two technologies. Diamonds indicate  $k$ -mers missing from one (x or y axis)

15 or both (0, 0) technologies.

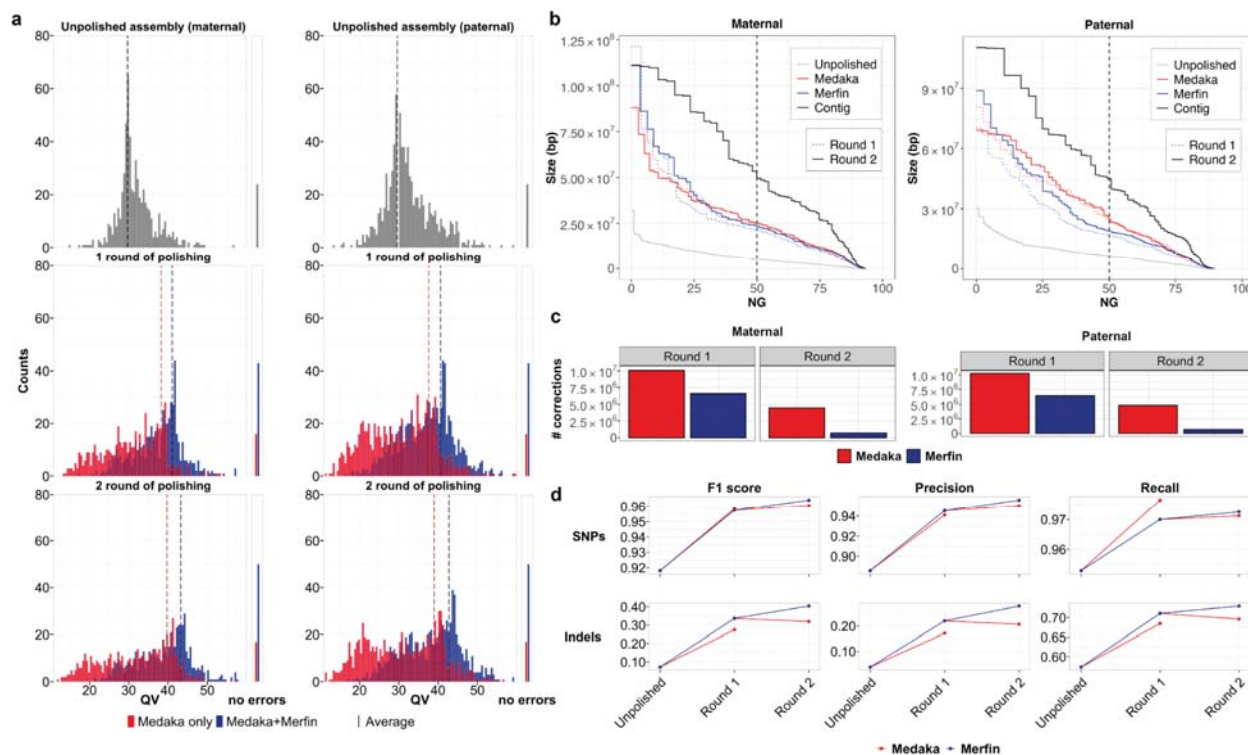
## 1 **Polishing a completely phased assembly: HG002**

2 The need for polishing is particularly evident in genome assemblies generated using noisy long reads.  
3 Therefore, we tested Merfin's variant calling filtering algorithm on a Nanopore-based assembly of human  
4 HG002 trio data generated by the HPRC using Flye<sup>31,32</sup>. We benchmarked Merfin on Medaka, by  
5 comparing polishing outcomes from Medaka with or without filtering with Merfin. In a trio setting, the  
6 optimal approach is to polish each parental assembly separately, by aligning the binned reads and  
7 performing variant calling<sup>5,33</sup>. This will prevent, or at least significantly reduce, the introduction of  
8 haplotype switches. However, our  $k$ -mer based evaluation of the corrections is best performed on a  
9 combined assembly so that it faithfully represents the expected copy-number of each  $k$ -mer given the read  
10 set. This further guards against haplotype switches since, even in case of read misbinning or mismapping,  
11  $k$ -mers from the other haplotype will not be considered underrepresented.

12 Therefore, we first called variants separately from the binned reads used in the assembly with Medaka,  
13 and then combined the variant calls and the parental assemblies for Merfin. We conducted five different  
14 experiments using read sets that differ in coverage, version of the Guppy basecaller, and read length cut-  
15 off (**Supplementary Table 3**). Two rounds of polishing were performed in all experiments, with the  
16 second round performed on the consensus from the first round generated with the additional Merfin step.  
17 Overall, in all experiments we observed comparable improvements in base calling accuracy as measured  
18 by Mercury QV when Merfin filtering was applied (**Supplementary Table 3**). This increase reflected a  
19 dramatic positive shift in the QV distribution of individual contigs, with most low-quality contigs being  
20 rescued by Merfin, and a sharp increase in the number of contigs found without errors, leading to a final  
21 Q43.2 and Q42.8 for maternal and paternal haplotypes, respectively (**Fig. 3a**). In the second round of  
22 polishing, the QV ceased to improve or even decreased when Merfin was not applied (**Fig. 3a**,  
23 **Supplementary Table 3**), suggesting that the best trade-off between errors corrected and introduced in  
24 the assembly was already reached in the first round. In contrast, the QV continued to increase relative to  
25 the first round with Merfin. Haplotype blocks as defined by Mercury increased in a comparable if not  
26 better way when using Merfin (**Fig. 3b**), while the haplotypes remained fully phased (**Supplementary**  
27 **Fig. 5**). Importantly, the results with Merfin were achieved by introducing only a fraction of the variants  
28 proposed by Medaka, making this approach more conservative than the regular polishing (**Fig. 3c**).

29 We also validated the HG002 unpolished and polished assembly by aligning each haplotype assembly to  
30 GRCh38 and deriving small variants. When benchmarked against GIAB v4.2.1 truth set<sup>26</sup>, the results  
31 show that using Merfin we get a better F1-score, particularly when INDELS are considered (**Fig. 3d**,  
32 **Supplementary Table 4**)<sup>26,34,35</sup>.





1  
 2 **Fig. 3. | HG002 human trio polishing and evaluation.** **a**, Distribution of QV scores as measured by Merqury for  
 3 maternal and paternal contigs polished with Medaka only, or with variants generated by Medaka filtered with  
 4 Merfin, from the experiment using latest basecaller and highest coverage. The first panel represents the unpolished  
 5 contigs, the mid panel the first round of Medaka polishing and filtering, and the last panel the second round applied  
 6 to the Merfin results from the previous round. The number of contigs without evidence of errors as judged by  
 7 Merqury QV are reported on the right side. **b**, Size of the haplotype blocks before and after polishing with or  
 8 without Merfin for both the maternal and paternal assemblies. First round of polishing is represented by the dotted  
 9 lines. **c**, Number of variants generated by Medaka for polishing and remaining variants after Merfin filtering for  
 10 both the maternal and paternal assemblies. **d**, Assembly-based HG002 small variant calling performance of Merfin  
 11 vs regular Medaka against GIAB truth set. Variants from the assembly are derived against GRCh38 using dipcall.

## 12 Evaluation, polishing and annotation of pseudo-haploid assemblies

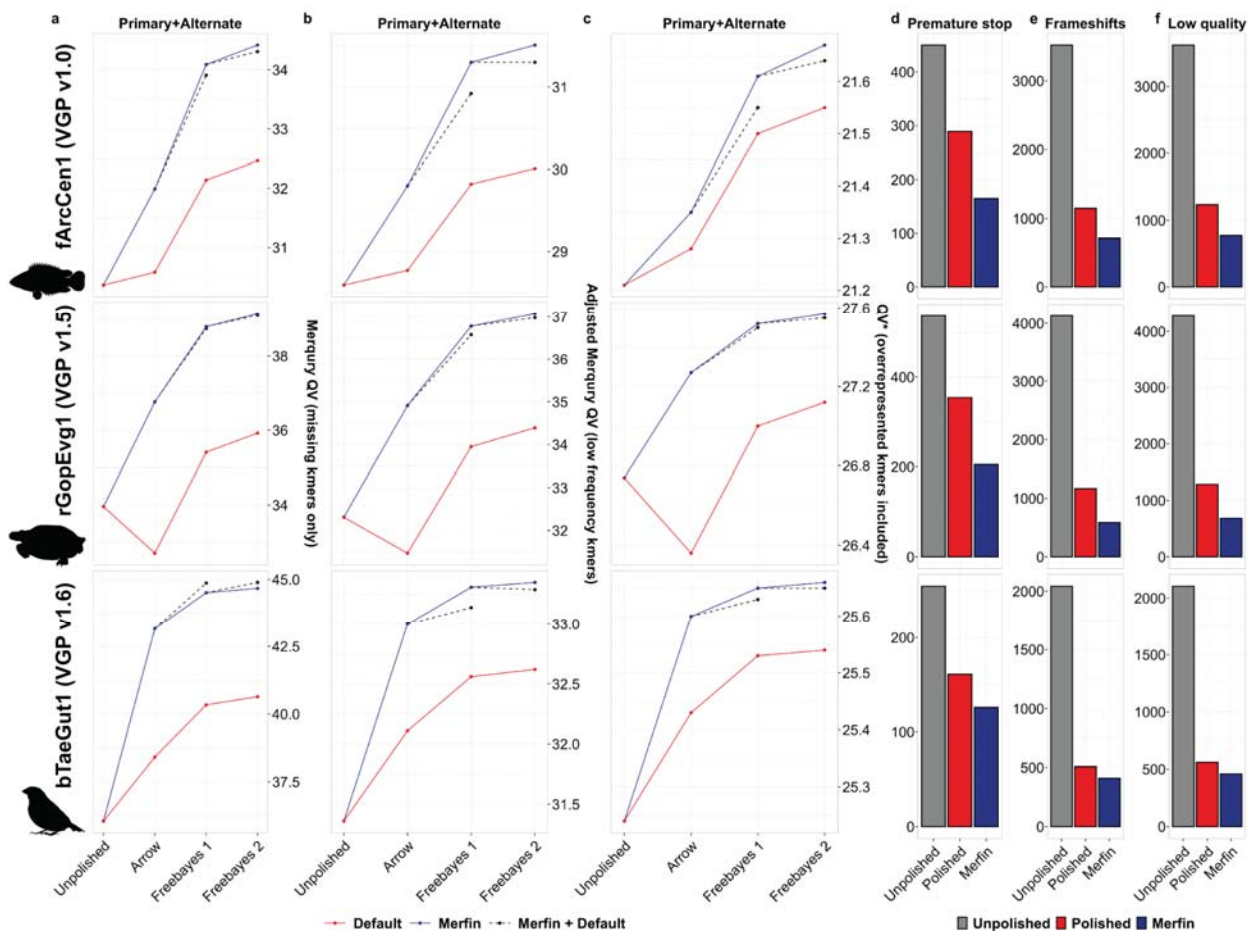
13 We next applied Merfin to the polishing steps of the VGP assembly pipeline<sup>5</sup> (**Supplementary Fig. 6**) on  
 14 pseudo-haplotypes from three species (flier cichlid, *Archocentrus centrarchus*, fArcCen1;  
 15 Goode's desert tortoise, *Gopherus evgoodei*, rGopEvg1; and zebra finch, *Taeniopygia guttata*, bTaeGut1).  
 16 Using Pacbio continuous long reads (CLR) and 10x Genomics linked-reads for polishing, we observed a  
 17 general improvement in QV as measured by Merqury (**Fig. 4a, Supplementary Table 5**). The largest  
 18 improvement was observed in the first round of Arrow polishing step using CLR. Arrow can replace low  
 19 quality sequences with patch sequences generated *de novo* from the reads that align to the region, i.e.,  
 20 independent from the quality of the original reference. We observed low coverage, sequencing biases (i.e.  
 21 homopolymer shortening), and mosaic haplotypes in the generated patches, leading to cases of lower QV  
 22 in the polished assembly (e.g. **Fig. 4a**, rGopEvg1). In all cases tested, Merfin rescued the QV decrease, or  
 23 improved the QV. In the subsequent polishing steps performed using Freebayes as the variant caller, the  
 24 benefit of running Merfin to filter the variant set was less pronounced but still present (**Fig. 4a**, dashed  
 25 lines). This was true in all cases but the zebra finch, where the default pipeline performed marginally

1 better. However, when considering low frequency  $k$ -mers as errors from the probability model in Merfin,  
2 the QV as well as QV\* increased in all cases (adjusted QV and QV\* in **Fig. 4bc, Supplementary Table**  
3 **5**). Merqury QV counts all  $k$ -mers never seen in the reads as errors, while the adjusted QV additionally  
4 counts low frequency  $k$ -mers based on the  $k$ -mer frequency spectrum as errors. The QV\* further includes  
5 overrepresented  $k$ -mers as errors. In doing so, the QV\* captures not only the base accuracy errors, but  
6 also false duplications, expressing the uncertainty associated with any particular base given the support  
7 from the raw reads.

8 Most long-read assemblers naturally generate locally phased haplotypes (e.g. Falcon-Unzip<sup>36</sup>), and it is  
9 therefore important that the polishing does not introduce haplotype switches. To test whether the increase  
10 in QV from Merfin was due to introducing haplotype switches, we tested a zebra finch (*Taeniopygia*  
11 *guttata*, bTaeGut2) pseudo-haplotype assembly for which parental sequence information is available<sup>5</sup>.  
12 When Merfin was applied to filter variants generated by Freebayes on the Longranger alignments of the  
13 10x reads, we noticed an increase in the number of haplotype switches as measured with Merqury  
14 (**Supplementary Table 2**). We realized that this was due to many heterozygous variants being called by  
15 Freebayes, when individual reads were mapped to collapsed regions or preferentially to the primary  
16 assembly which had higher base accuracy<sup>5</sup>. Indeed, such assemblies violate the read-assembly agreement,  
17 potentially leading to heterozygous calls being preferred by Merfin. Even in almost complete pseudo-  
18 haplotype assemblies, short reads can be easily mismapped, leading to spurious heterozygous calls. To  
19 overcome this issue, we decided to remove all heterozygous variants before applying with Merfin. This  
20 substantially prevented haplotype switches (**Supplementary Fig. 7**), with an increase in QV  
21 (**Supplementary Table 6**). In conclusion, we propose removing all heterozygous variants prior to Merfin  
22 as the best practice for polishing pseudo-haploid and haploid assemblies.

23 In addition, we validated our results using gene annotations, which are sensitive to consensus accuracy  
24 error, and particularly to frameshift errors caused by indel errors. We performed *de novo* gene annotation  
25 using RefSeq gene annotation pipeline (GNOMON)<sup>37</sup> on the VGP assemblies polished with the  
26 conventional VGP pipeline (v1.6) and compared against assemblies where Merfin was applied at every  
27 polishing step. In GNOMON, if a protein alignment supports a predicted model with an indel introducing  
28 frameshift or premature stop codons, the model gets labeled as ‘low quality’ with the frame marked for  
29 correction. If more than 1 in 10 coding genes in an assembly require corrections, the assembly is excluded  
30 from RefSeq. Almost all rejected assemblies used ONT or PacBio CLR reads as their backbone, based on  
31 sequence technology information provided by the submitters.

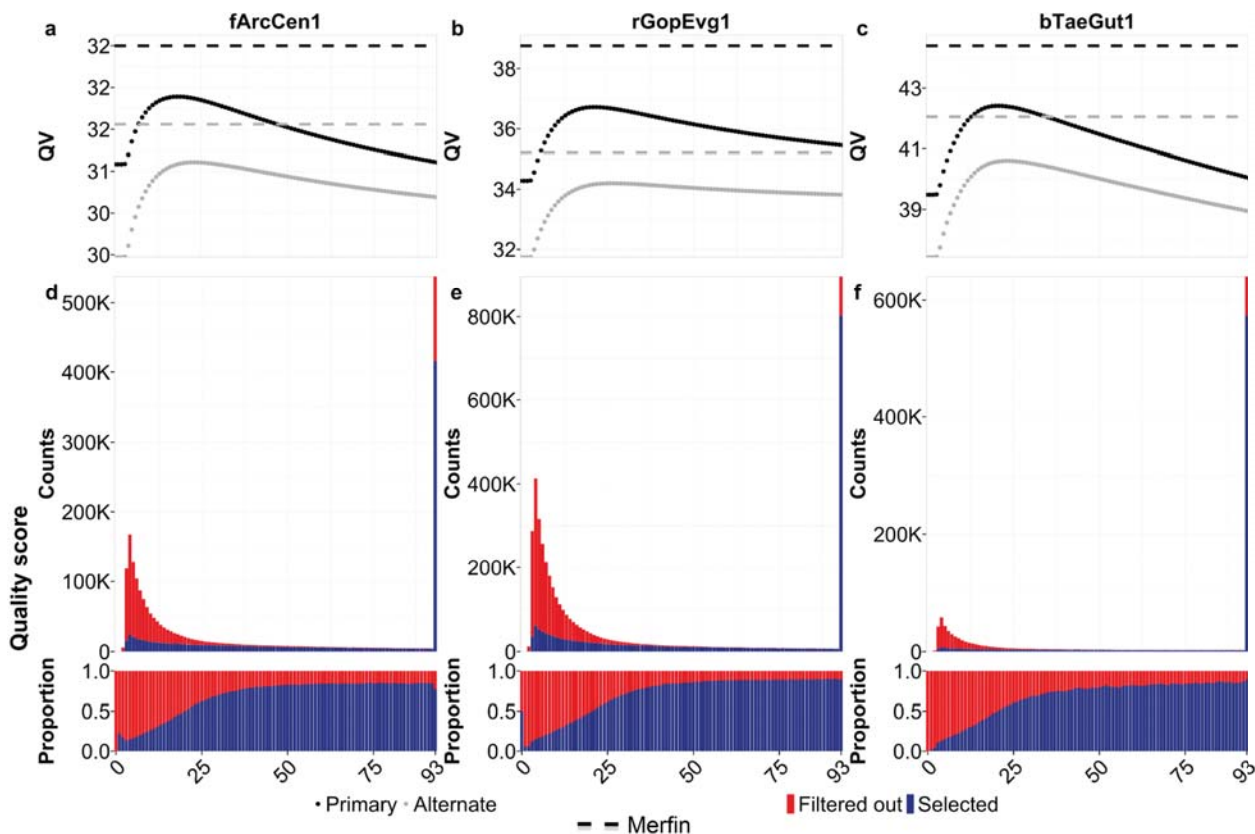
32 Again, Merfin substantially reduced the number of genes affected by frameshifts, cross-validating the QV  
33 and QV\* results (**Fig. 4d-f, Supplementary Table 7** and example in **Supplementary Fig. 8**). In  
34 particular, premature stop codons were significantly reduced with respect to the default polishing in all  
35 cases (**Fig. 4d**), with 42.9%, 42% and 21.7% reduction in fArcCen1, rGopEvg1 and bTaeGut1,  
36 respectively. Ultimately, 1% or less of genes had code breaks in all cases when using Merfin. Frameshifts  
37 were also positively affected (**Fig. 4e**), with 38%, 49.6% and 19.5% reductions in fArcCen1, rGopEvg1  
38 and bTaeGut1, respectively. Less than 3% of genes had frameshifts in all cases when using Merfin.  
39 Similarly, the number of protein-coding gene predictions labelled as ‘low quality’ were reduced (**Fig. 4f**).  
40 From these results, we now include Merfin as part of the VGP pipeline (v1.7) for its beneficial effect.



1  
2 **Fig. 4. | Polishing and evaluation of VGP pseudo-haploid assemblies.** a-c, Polishing results of primary and  
3 alternate assemblies for the flier cichlid (fArcCen1), the Goode's desert tortoise (rGopEvg1), and the zebra finch  
4 (bTaeGut1) using the VGP pipeline. Graphed are the unpolished QV values, and the Merqury QV that accounts only  
5 for missing *k*-mers (a), the Merqury QV corrected using Merfin models for 0-copy *k*-mers (b), and QV\* that also  
6 accounts for overrepresented *k*-mers (c). d-f, the general QV increase was reflected in the quality of the gene  
7 annotation, with consistent reduction in the number of genes affected by premature stop codons (d), frameshifts  
8 errors (e), and low quality protein-coding gene predictions (f).

9 Consistent with the variant filtering for genotyping, the improvements in QV with Merfin superseded any  
10 hard-filtering attempt using variant call quality score (QUAL) cutoffs at the Arrow polishing step (**Fig.**  
11 **5a-c, Supplementary Table 8**). For the primary assembly, QV\* estimates were consistently higher than  
12 the best results attainable by hard-filtering (fArcCen1: Q32.5 vs. Q31.9 at  $QUAL \geq 18$ , rGopEvg1:  
13 Q38.7 vs. Q36.7 at  $QUAL \geq 21$ , bTaeGut1: Q44.4 vs. Q42.4 at  $QUAL \geq 21$ ). The best QUAL cutoff  
14 was not necessarily consistent between species, indicating that a single cutoff cannot produce the best  
15 outcome in all cases. The alternate assembly (i.e. alternate haplotype) behaved similarly to the primary  
16 assembly, again with Merfin always performing best (fArcCen1: Q31.6 vs Q31.1 at  $QUAL \geq 23$ ,  
17 rGopEvg1: Q35.2 vs. Q34.2 at  $QUAL \geq 26$ , bTaeGut1: Q42.0 vs. Q40.6 at  $QUAL \geq 23$ ); however, it  
18 notably differed in best QUAL cutoff values to maximize QV. At increased QUAL cutoffs, both genuine  
19 and erroneous corrections are filtered out. Thus, hard-filtering cutoffs perform best when the number of

1 errors corrected exceeds the number of errors introduced at maximum. In contrast, variants selected by  
 2 Merfin had a wide range of quality scores, with the majority containing higher quality scores yet  
 3 including some below 25 (**Fig. 5d-f**). Notably, a significant fraction of variants with the highest quality  
 4 score assigned were introducing error  $k$ -mers and thus were rejected by Merfin. Potentially, accumulated  
 5 sequencing biases in long-reads could lead to erroneous variant calls but can be filtered with more  
 6 accurate  $k$ -mers from short-reads. No hard-filtering methods were able to achieve QV improvements in  
 7 polishing as observed with Merfin.



8  
 9 **Fig. 5. | Merfin results against quality scores.** a-c, QV after polishing as a function of hard-filtered quality score  
 10 cutoff in primary (black) and alternate (gray) assembly. Results achieved with Merfin are represented by the  
 11 horizontal lines for comparison. d-f, Number and proportion of variants by quality score selected by Merfin.

## 12 Discussion

13 Here, we described and demonstrated Merfin, a  $k$ -mer based tool to evaluate and filter variant calls for  
 14 improved genotyping accuracy and polishing. Importantly, Merfin allows an innovative alignment-free  
 15 evaluation and filtering of variants from any VCF generated from any dataset or variant calling method.  
 16 Merfin always successfully removes false positive calls, superseding any hard-filter based cutoff for both  
 17 genotyping and polishing. Contrary to the *plateau* effect usually observed in traditional polishing, our  $k$ -  
 18 mer-informed polishing is essentially a monotonic function, predicted to improve the consensus accuracy  
 19 until no more useful variants are produced by the variant caller. This lets polishing pipelines have a  
 20 natural stopping condition to set, i.e. to stop iterative polishing when less than a certain fraction of  
 21 produced variants are being filtered.

1  
2 In addition, using the copy number agreement between the reads and the assembly, we revised  $K^*$ , a  
3 metric to identify and analyze local expansions and collapses at each  $k$ -mer genome-wide. We devised  
4  $QV^*$  and  $K^*$  completeness, new quality metrics that account for over and under represented  $k$ -mers  
5 undetected by previous methods<sup>16</sup>. We demonstrated on a complete human genome that our approach  
6 allows orthogonal validation of both consensus sequence and variants with any sequencing read data type.

7  
8 Similarly to all  $k$ -mer-based estimates,  $K^*$  estimates are influenced by the choice of  $k$ , which is dependent  
9 on the quality of the reads. The results presented here assume high-accuracy reads for evaluation and  
10 variant filtering, and may therefore not work best with  $k$ -mers derived from noisy long-reads (i.e. CLR  
11 reads and early ONT data). Presence of sequencing biases will also result in biased  $K^*$ , such as the GC  
12 bias in Illumina reads or the GA dropouts in HiFi reads<sup>25</sup>. Yet, these effects are limited only to certain  
13 regions of the genome, and it could be potentially further mitigated by methods that correct sequencing  
14 reads for known biases<sup>38</sup>. Obviously, the ultimate solution will come with less biased, or even unbiased,  
15 sequencing reads.

16 In parallel, the completeness of the assembly also affects the  $K^*$ . Pseudo-haploid or haploid  
17 representation of a genome may potentially lead to suboptimal evaluation because of the missing copies.  
18 However, we argue this is a limitation of the assemblies themselves, rather than a limitation of the  
19 methods used to evaluate and polish them. While haploid or partially phased (e.g. FALCON-Unzip<sup>36</sup>)  
20 assemblies can be preferred for some applications, a faithful reconstruction of the complete genome  
21 should be preferred for both evaluation and comparative purposes, as well as for many biological analyses  
22 that can benefit from the presence of both haplotypes (e.g. trio binning<sup>33,39</sup>). Representing a diploid  
23 genome as a haploid or pseudo-haploid assembly introduces complications in the evaluation, since the  $k$ -  
24 mers in the consensus will not fully reflect the  $k$ -mers in the read set. Homozygous  $k$ -mers will be  
25 underrepresented, and some of the alternate haplotype  $k$ -mers will be completely missing. The recent  
26 developments in assembly graphs enable the representation of complete haplotypes with enhanced  
27 accuracy and completeness<sup>40</sup>, suggesting that assembly tools and state-of-the-art assemblies are moving  
28 toward this direction. If this condition is met, the information contained in the reads can be fully  
29 harnessed to evaluate and improve genome assemblies.

30 Merfin presents the first  $k$ -mer based variant filtering to the best of our knowledge, enabling higher  
31 precision in genotyping and improving assembly accuracy. This will become critical particularly in  
32 medical genomics and many other applications, where reliable genotyping is highly important. Polishing  
33 with Merfin will rescue assemblies built from noisy long reads, when the more recent, accurate platforms  
34 are not accessible, or when sequencing biases are subject to correction using complementary sequencing  
35 platforms.

## 36 37 **Online methods**

### 38 **Genotyping benchmark**

39 Variant calls from HG002 submitted to precisionFDA Truth Challenge<sup>1</sup> were downloaded from  
40 <https://data.nist.gov/od/id/mds2-2336>. In brief, ~35x Illumina PCRfree, ~36x PacBio HiFi, and ~47x  
41 ONT reads were aligned to the human genome reference (GRCh38) with no alternates. Variants were  
42 called with GATK HaplotypeCaller v4. Unfiltered and hard-filtered set was downloaded and

1 benchmarked with hap.py (v0.3.12-2-g9d128a9) following the best practices from a previous study<sup>26</sup>.  
2 Precision and recall were collected before and after filtering the variants with Merfin.

3  
4 PCR-free Illumina paired-end reads (2x250 bp) were obtained from NIST ([https://ftp-trace.ncbi.nlm.nih.gov/giab/ftp/data/AshkenazimTrio/HG002\\_NA24385\\_son/NIST\\_Illumina\\_2x250bps/](https://ftp-trace.ncbi.nlm.nih.gov/giab/ftp/data/AshkenazimTrio/HG002_NA24385_son/NIST_Illumina_2x250bps/))  
5 and 21-mers were collected using Meryl.  $K$ -mers with frequency  $> 1$  were used as read  $k$ -mers to avoid  $k$ -  
6 mer collisions from sequencing errors and improved computational performance.  
7  
8

### 9 **Revised $K^*$**

10  $K$ -mers are substrings of length  $k$  of a given DNA sequence. Given the assembly consensus sequence, we  
11 compute all its constituent  $k$ -mers. Similarly, we compute all  $k$ -mers represented in a set of WGS reads  
12 from the same individual. We then ask how the frequency of each  $k$ -mer in the read set is mirrored in the  
13 assembly  $k$ -mer set. If the read set is a faithful representation of the genome (i.e. in the absence of random  
14 DNA sampling and sequencing biases), then the closer the consensus sequence is to the read set, the  
15 closer it is also to the genome the reads were generated from. This principle can be usefully represented  
16 by our revised  $K^*$ , where for each  $k$ -mer in the consensus we can calculate (**Fig. 1a**):

17  $K_C = k$ -mer count in the consensus sequence

18  $K_R = k$ -mer count in the read set

19 To account for the uncertainty associated with the underlying Poisson sampling process, in any  
20 sequencing experiment the read set covers on average the original genome multiple times. It is therefore  
21 useful to determine the expected copy number of a particular  $k$ -mer in the assembly given the read set,  $K_r$ ,  
22 as:

23  $c =$  haploid peak from  $K_R$  histogram

24  $K_r =$  the  $k$ -mer count expected in the consensus based on the read set, i.e.  $\lceil K_R / c \rceil$

25 Note that  $K_r - K_C$  expresses the number of copies of any particular  $k$ -mer that is underrepresented  
26 (collapsed; positive value) or overrepresented (expanded; negative value) in the assembly.

27 With these definitions, we can now define  $K^*$  as:

28  $K^* = K_r / K_C - 1$  if  $K_r > K_C$  (collapsed  $k$ -mers)

29  $K^* = -(K_C / K_r - 1)$  if  $K_r < K_C$  (expanded  $k$ -mers)

30 Which can be reduced to:

31  $K^* = (K_r - K_C) / \min(K_r, K_C)$

32 Note that  $K^*$  converges to 0 if the  $k$ -mer frequency in the assembly matches the expected copy number in  
33 the reads. Missing  $k$ -mers (i.e. found in the assembly but not in the read set) have a special behavior with  
34  $K^*$  being “undefined” for  $K_r = 0$ .

### 35 **Probabilistic $K$ -mer copy-number estimation**

36 To estimate  $k$ -mer copy-number in the genome, we modified Genomescope2<sup>27</sup> to obtain the associated  
37 probability at each  $K_R$ . Our additions were subsequently integrated in the current version of

1 Genomescope2 (<https://github.com/tbenavi1/genomescope2.0>). Unmodified fitted model 1- to 4-copy  $k$ -  
2 mer distributions were used to infer the probability that a particular  $k$ -mer frequency observed in the read  
3 set implied a particular copy  $k$ -mer in the genome. Using this model, Merfin provides a script generating a  
4 lookup table for each  $k$ -mer frequency in the raw data with the most plausible  $k$ -mer multiplicity and its  
5 associated probability ([https://github.com/arangrhie/merfin/tree/master/scripts/lookup\\_table](https://github.com/arangrhie/merfin/tree/master/scripts/lookup_table)).

## 6 **QV estimation using the $K^*$**

7 An average genome-wide  $QV^*$  is obtained by counting all  $k$ -mers not present compared to the expected  
8 copy number estimated from the read set. We collect all  $k$ -mers excessively found in the assembly ( $K_E$ )  
9 and estimate the error rate given all  $k$ -mers in the assembly ( $K_{total}$ ).

10  $K_E = \sum K_C - K_r$  when  $K_C > K_r$  for all positions in the assembly

11 The Phred-scaled  $QV^*$  can be computed using the implementation in Merqury<sup>16</sup> or YAK<sup>28</sup>.

12 We follow the implementation in Merqury and compute the probability  $P$  that a base in the assembly is  
13 correct and in its expected frequency:

$$14 \quad P = ((K_{total} - K_E) / K_{total})^{1/k}$$

15 Which leads to error rate  $E$  being:

$$16 \quad E = 1 - P$$

17 Hence the phred scaled  $QV^*$  becomes:

$$18 \quad QV^* = -10 \log E$$

## 19 **Assembly completeness using the $K^*$**

20 To estimate completeness, we collect all  $k$ -mers that should be present but are absent from the assembly.  
21 Unlike Merqury, we account for the  $k$ -mer frequency and count any  $k$ -mer that should be added to meet  
22 the expected frequency from the reads  $K_A$ .

$$23 \quad K_A = \sum (K_r - K_C) \text{ when } K_r > K_C \text{ for all } K_r, \text{ including } K_C = 0$$

24  
25 We compute the completeness  $Comp$  given all  $K_r$ :

$$26 \quad Comp (\%) = (K_r - K_A) / K_r = 1 - K_A / K_r$$

## 27 **Sequence data**

28 For the HG002 results, data can be found at [https://github.com/human-](https://github.com/human-pangenomics/HG002_Data_Freeze_v1.0)  
29 [pangenomics/HG002\\_Data\\_Freeze\\_v1.0](https://github.com/human-pangenomics/HG002_Data_Freeze_v1.0). For the VGP datasets, PacBio CLR and 10x Genomics linked  
30 reads can be found at <https://vgp.github.io/genomeark/><sup>5</sup>.

## 31 **Evaluation of CHM13 assemblies**

32 All scripts used for CHM13 evaluation can be found here:  
33 <https://github.com/gf777/misc/tree/master/merfin>. Briefly, we generated genome-wide  $K^*$  tracks using  
34 Merfin option `-dump` (`merfin_dump.sh`).  $K$ -mer counts databases for both the assemblies and the raw

1 Illumina and HiFi reads were computed using Meryl (<https://github.com/marbl/meryl>). Peak values of  
2 106.8 and 31.8 derived as the *k*cov value from Genomescope2 were used for Illumina and HiFi *k*-mers,  
3 respectively. The tracks were converted to bigWig and loaded in IGV<sup>41</sup> for visualization. We used a  
4 custom script (`simplify_dump.sh`) to count the number of bases with the same *K*\* values for both Illumina  
5 and HiFi *k*-mers, which were then used to generate the genome-wide *K*\* comparison. The titration  
6 experiment was performed downsampling the reads with the ‘`seqtk sample`’ command  
7 (<https://github.com/lh3/seqtk>).

## 8 **Variant calling and polishing of HG002 assemblies**

9 Variant calling and polishing of HG002 assemblies was performed using medaka 1.2.6  
10 (<https://github.com/nanoporetech/medaka>) using the models specified in **Supplementary Table 3** for  
11 each dataset. Medaka was first run in the consensus mode (`medaka_consensus`) and subsequently in the  
12 variant mode (`medaka_variant`) to generate the vcf of the variant calls. Medaka filtered variant set was  
13 then used in conjunction with bcftools v1.9 in the consensus mode with the `-H 1` option to generate a  
14 consensus sequence. The same procedure was followed for the Merfin assemblies, except that Merfin was  
15 used to filter Medaka vcf prior to consensus generation. Polishing was repeated twice, and in the second  
16 round the assembly polished with Merfin was used as reference.

## 17 **Assembly-based small variant calling assessment**

18 We used dipcall (<https://github.com/lh3/dipcall>) to generate the small variants from the assembly. Dipcall  
19 takes a diploid assembly and a reference genome to produce a variant call file (VCF) that contains all  
20 variants that are present in the assembly compared to the reference. We then compared the variant calls  
21 against GIAB truth set v4.2.1 using hap.py (<https://github.com/Illumina/hap.py>). The GIAB variant  
22 calling truth set for HG002 sample can be found in: ([ftp://ftp-  
23 trace.ncbi.nlm.nih.gov/giab/ftp/release/AshkenazimTrio/HG002\\_NA24385\\_son/NISTv4.2.1/](ftp://ftp-trace.ncbi.nlm.nih.gov/giab/ftp/release/AshkenazimTrio/HG002_NA24385_son/NISTv4.2.1/)). We used  
24 the following commands for the evaluation:

```
25  
26 ./run-dipcall <output_prefix> <GRCh38.fa> <pat.fa> <mat.fa> -t 8 -x hs38.PAR.bed  
27  
28 docker run -it pkrusche/hap.py:latest /opt/hap.py/bin/hap.py  
29 HG002_GRCh38_1_22_v4.2.1_benchmark.vcf.gz DIPCALL_OUTPUT.dip.vcf.gz -f  
30 HG002_GRCh38_1_22_v4.2.1_benchmark_noinconsistent.chr20.bed -r  
31 GCA_000001405.15_GRCh38_no_alt_analysis_set.fna -o OUTPUT --pass-only --engine=vcfeval --  
32 threads=32
```

## 33 **Variant calling and polishing of VGP assemblies**

34 While the original assemblies were generated with different versions of the VGP pipeline<sup>5</sup>  
35 (<https://github.com/VGP/vgp-assembly/tree/master/pipeline>), to polish the assemblies of the flier cichlid  
36 (v1.0), the Goode's thornscrub tortoise (v1.5), and the zebra finch (v1.6) with Merfin we used the VGP  
37 pipeline v1.6 (**Supplementary Fig. 7**). In the first round of polishing, long Pacbio CLR reads were  
38 aligned with pbmm2 v1.0.0, variants were called with variantCaller v2.3.3 (arrow) with the ‘`-o  
39 ${asm}.vcf`’ option. A custom script  
40 ([https://github.com/aranghie/merfin/blob/master/scripts/reformat\\_arrow/](https://github.com/aranghie/merfin/blob/master/scripts/reformat_arrow/)) included in Merfin was used to  
41 properly format the vcf file. 21-mer databases for both the assemblies and the 10x linked-reads were  
42 generated with Meryl. 10x barcodes were trimmed from the reads using the script available in Meryl. The



1 haploid 21-mer coverage and the lookup tables were computed using our modified Genomescope2 script  
2 included in Merfin:

3  
4 Rscript \$merfin/lookup.R \${asm}.21.meryl.hist 21 \${asm}.21.lookup 2  
5

6 Similarly to HG002, the consensus was generated with bcftools v1.9 using the filtered vcf generated by  
7 Merfin. The same strategy was applied for the other polishing steps, except that Longranger v2.2.2 was  
8 used for mapping the 10x Genomics linked-reads and Freebayes v1.3.1 for variant calling. For Variant  
9 calling and polishing of zebra finch trio, Freebayes calls were filtered using Bcftools v1.9 with the -  
10 i'(GT="AA" || GT="Aa")' option prior to Merfin filtering. *K*-mer counts databases for both the assemblies  
11 and the raw Illumina reads were computed using Meryl, and Merfin was run with a peak value of 35.2  
12 derived as the *k*cov value from Genomescope2.

### 13 **Evaluation of the assemblies**

14 QV and phasing analyses of HG002 and zebra finch trios were performed using Merquy<sup>16</sup>  
15 (<https://github.com/marbl/merquy/>) in the trio mode using 21-mers and default parameters. Similarly,  
16 primary and alternate scaffolds of the VGP assemblies were separated and Merquy QV was estimated on  
17 both using 21-mers and default parameters.

### 18 **Gene annotation of VGP assemblies**

19 Annotation was performed as previously described<sup>5</sup>, using the same transcript, protein and RNA-Seq  
20 input evidence for the annotation of the unpolished, polished and Merfin assemblies of each species. For  
21 *Taeniopygia guttata*, a total of 100,000 *Taeniopygia guttata* ESTs, GenBank and known RefSeq and 10  
22 billion same-species reads for over 13 tissues were aligned to the genome, in addition to all Genbank *Aves*  
23 proteins, known *Aves*, human and *Xenopus* RefSeq proteins, and RefSeq model proteins for *Parus major*,  
24 *Gallus gallus*, *Columbia livia* and *Pseudopodoces humilis*. For *Gopherus evgoodei*, 1.22 billions RNA-  
25 Seq reads from 5 tissue types from *Gopherus* and *Chelonoidis* species were aligned to the assemblies in  
26 addition to all known RefSeq proteins from human, *Xenopus*, and *Sauropsida*, and model RefSeq proteins  
27 from *Chrysemys picta*, *Pelodiscus sinensis*. For *Archocentrus centrarchus*, 476 million same species  
28 RNA-Seq reads from 9 tissue types were aligned to the assemblies in addition to all *Actinoipterygii*  
29 GenBank proteins, human and *Actinoipterygii* known RefSeq proteins and *Oryzias latipes*, *Oreochromis*  
30 *niloticus*, *Monopterus albus*, *Xiphophorus maculatus* model RefSeq proteins.

### 31 *Plots*

32 A combination of bash and Rscript was used for data analysis and visualization. All scripts used to  
33 generate the plots are available at <https://github.com/gf777/misc/tree/master/merfin/paper/figures>.

34

### 35 **Availability**

36 A stable release and the C++ source code for Merfin, and examples from this work are available under  
37 Apache License 2.0 (<https://github.com/arangrhie/merfin>). The only dependency is the *k*-mer counter  
38 Meryl, which comes with the release. Merfin can be run in five modes: 1) the -filter mode scores each  
39 variant, or variants within distance *k* and their combinations by error *k*-mers for improved genotyping; 2)  
40 the -completeness mode generates completeness metrics; 3) the -dump mode computes  $K_C$ ,  $K_R$ ,  $K^*$  for  
41 each base in the assembly along with QV and QV\* for each sequence; 4) the -hist mode provides a  $K^*$

1 histogram and genome-wide QV and QV\* averages; 5) the -polish mode scores each variant, or variants  
2 within distance  $k$  and their combinations by the  $K^*$  for polishing. Merfin is fully parallelized using  
3 OpenMP. The  $K^*$  tracks obtained from each platform for the CHM13 v1.0 assembly are available in the  
4 associated UCSC browser (<http://t2t.gi.ucsc.edu/chm13/dev/hub.txt>).

## 5 **Author Contributions**

6 A. R., G. F., B. P. W. and E. M. conceived and implemented Merfin. G. F. and A. R. performed the  
7 validation analyses. K. S. performed the GIAB variant calling analysis on HG002. F. T. generated the  
8 gene annotations for the VGP genomes. S. K. contributed to the conceptual development. G. F. and A. R.  
9 wrote the manuscript. G. F., A. R., E. D. J. and A. M. P. conceived the study. All authors reviewed,  
10 edited, and approved the manuscript.

11

## 12 **Competing Interests statement**

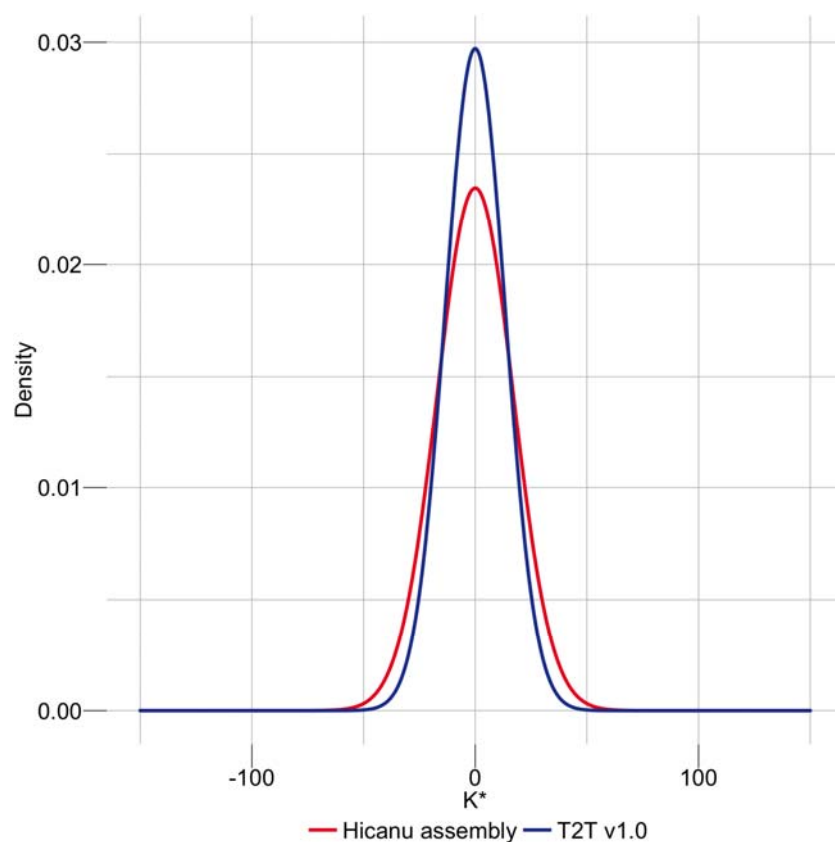
13 The authors declare no competing interest.

14

## 15 **Acknowledgments**

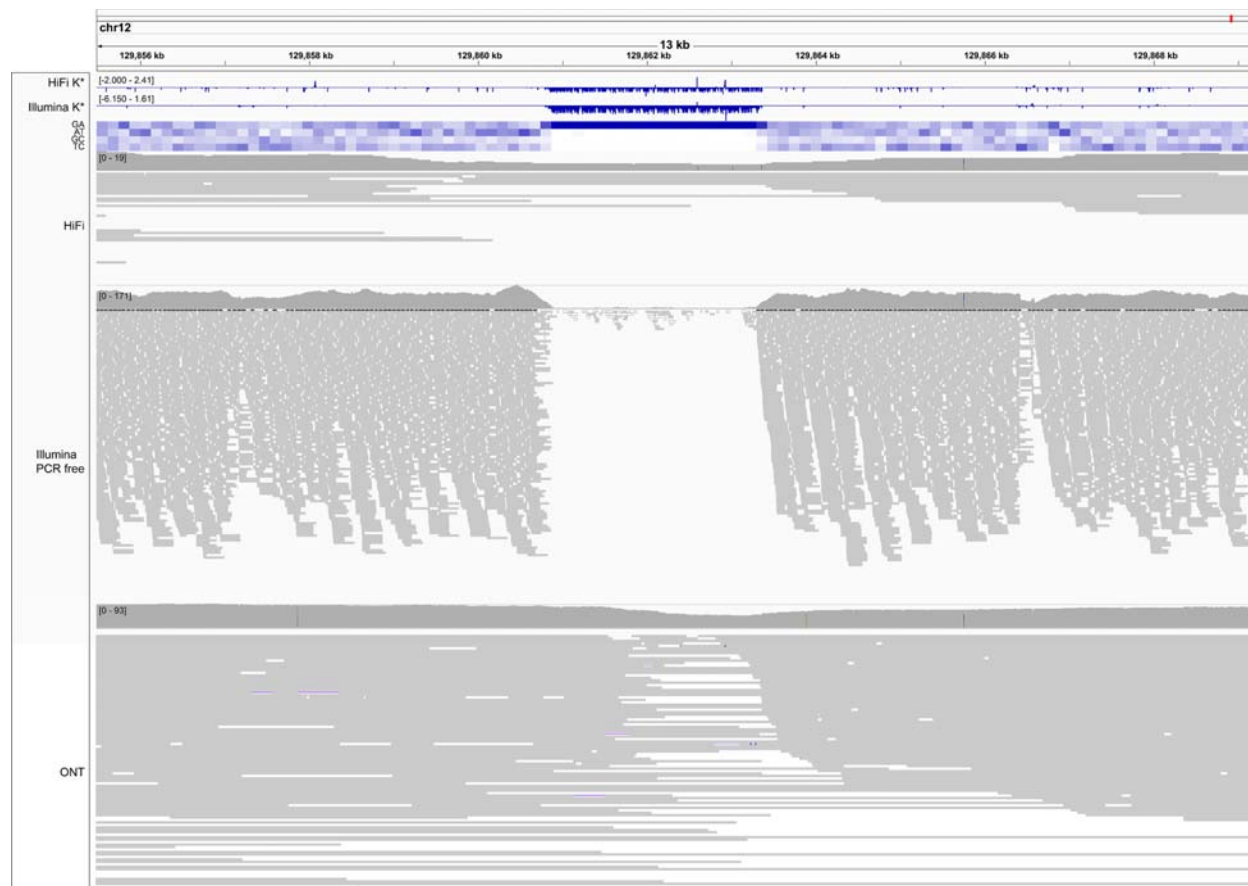
16 We thank T. Rhyker Ranallo-Benavidez and Michael C. Schatz for the useful discussion on adapting  
17 Genomescope2 models. We also thank the communities of the T2T, HPRC and VGP consortia for their  
18 constant support. G. F. and E. D. J. were supported by Rockefeller University and HHMI funds. A. R., B.  
19 P. W., S. K., and A. M. P. were supported by the Intramural Research Program of the National Human  
20 Genome Research Institute, National Institutes of Health. The work of F.T-N was supported by the  
21 Intramural Research Program of the National Library of Medicine, National Institutes of Health. K. S.  
22 was supported by NIH/NHGRI (R01HG010485, U41HG010972, U01HG010961, U24HG011853,  
23 OT2OD026682). E. W. M. was partially supported by the German Federal Ministry of Education and  
24 Research (01IS18026C)

## 1 Supplementary Figures



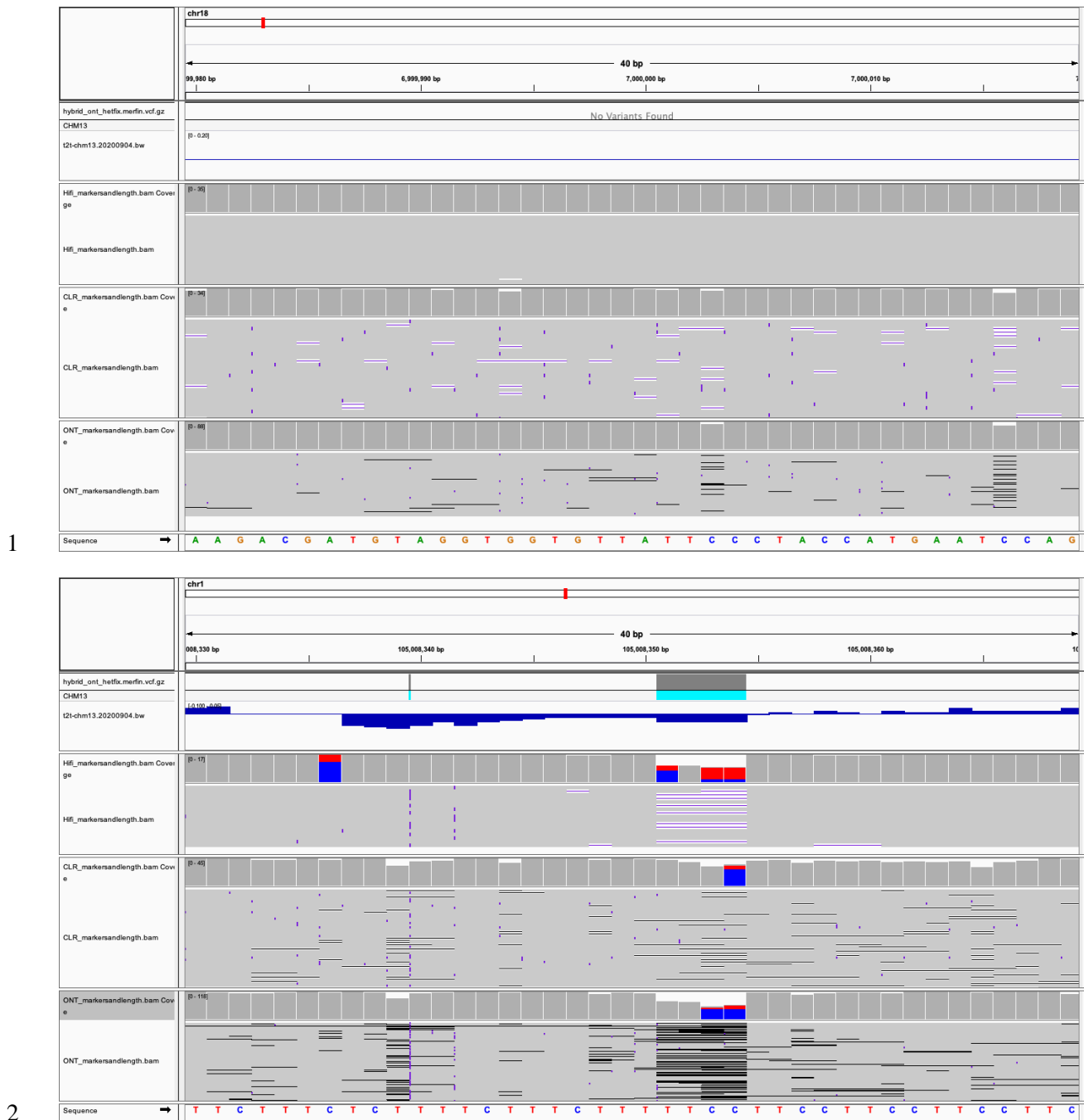
2

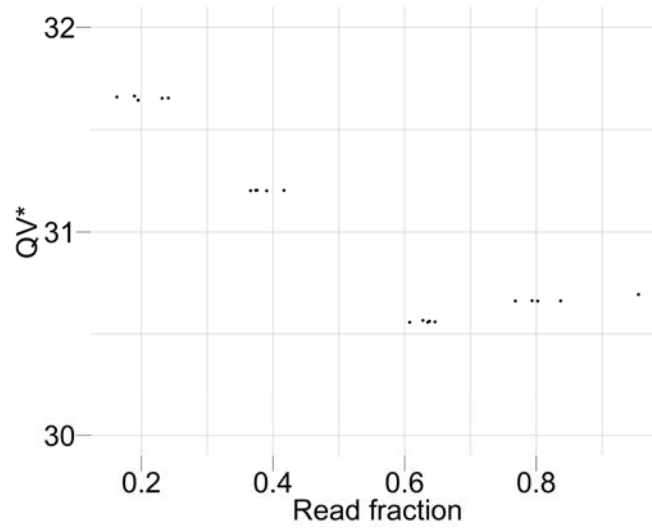
3 **Supplementary Figure 1 | Genome-wide density distribution of the  $K^*$  using Illumina  $k$ -mers.** When  
4 the assembly is in agreement with the raw data, the  $K^*$  is normally distributed with mean 0, and the  
5 smaller the standard deviation the higher the agreement. CHM13 v1.0 shows a less dispersed distribution  
6 of the  $K^*$  compared to a regular HiCanu assembly.



1  
2 **Supplementary Figure 2 | A region of negative K\* highlighting sequencing bias.** An example  
3 of low coverage in both HiFi and Illumina reads associated with high guanine content, and  
4 specifically a GA-rich repeat (heatmap). GA bias has been reported in Pacbio HiFi data, and  
5 results in gaps in the assembly that in CHM13 were filled with Nanopore data<sup>22</sup>. The K\* both  
6 from HiFi and Illumina *k*-mers (top tracks) recapitulate the coverage drop. Nanopore coverage  
7 appears less affected. Position Chr12:~129,862,000 bp.

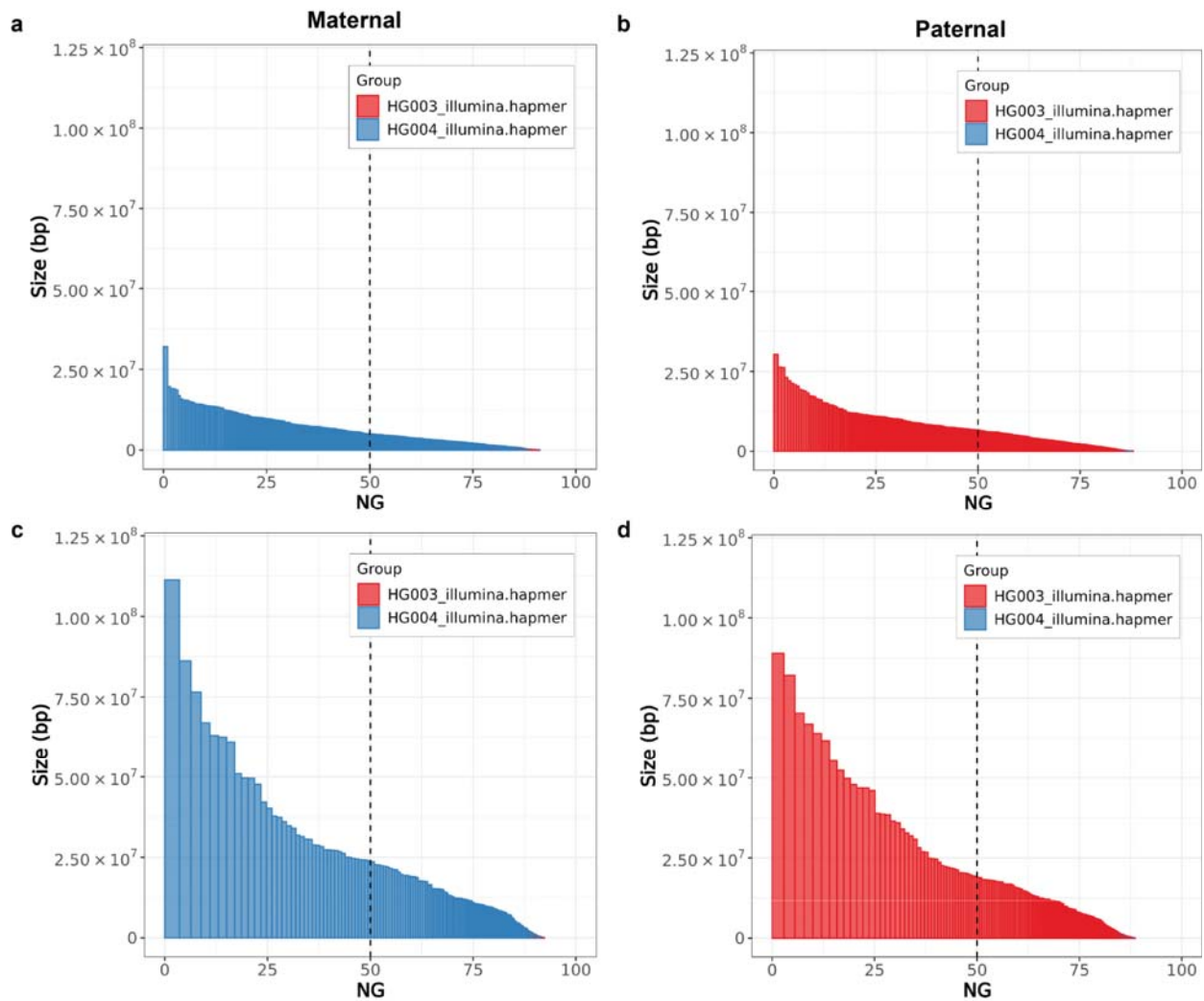
8  
9





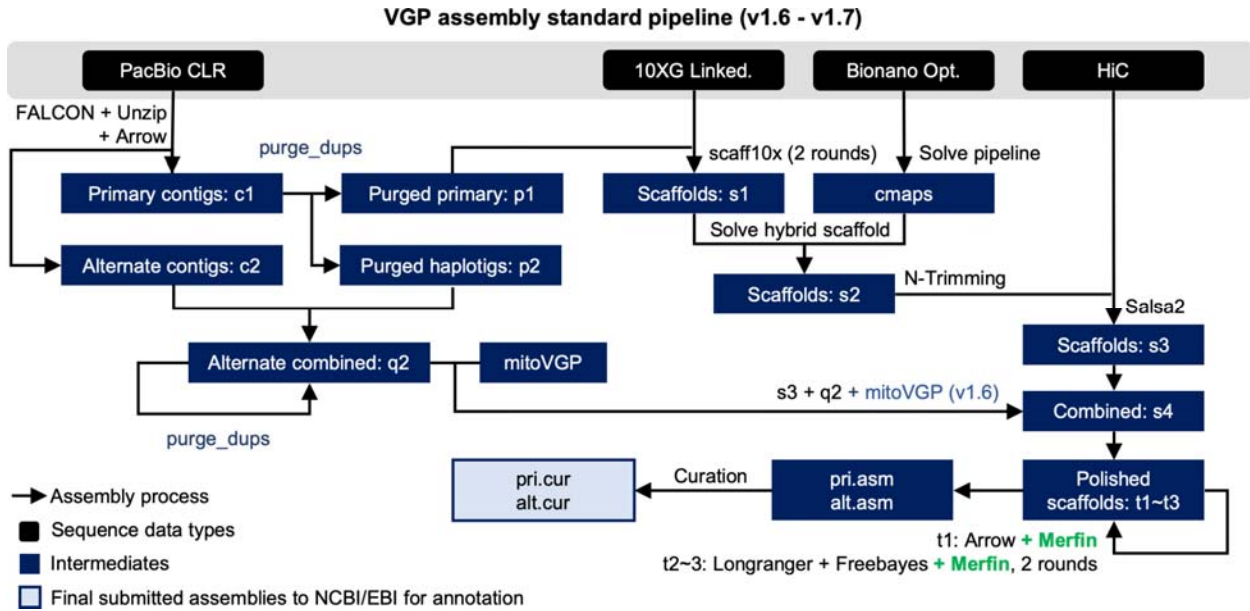
1

2 **Supplementary Figure 4 | Coverage titration experiment and impact on QV\*.** The QV\* is only  
3 marginally influenced by the coverage of the dataset being considered.

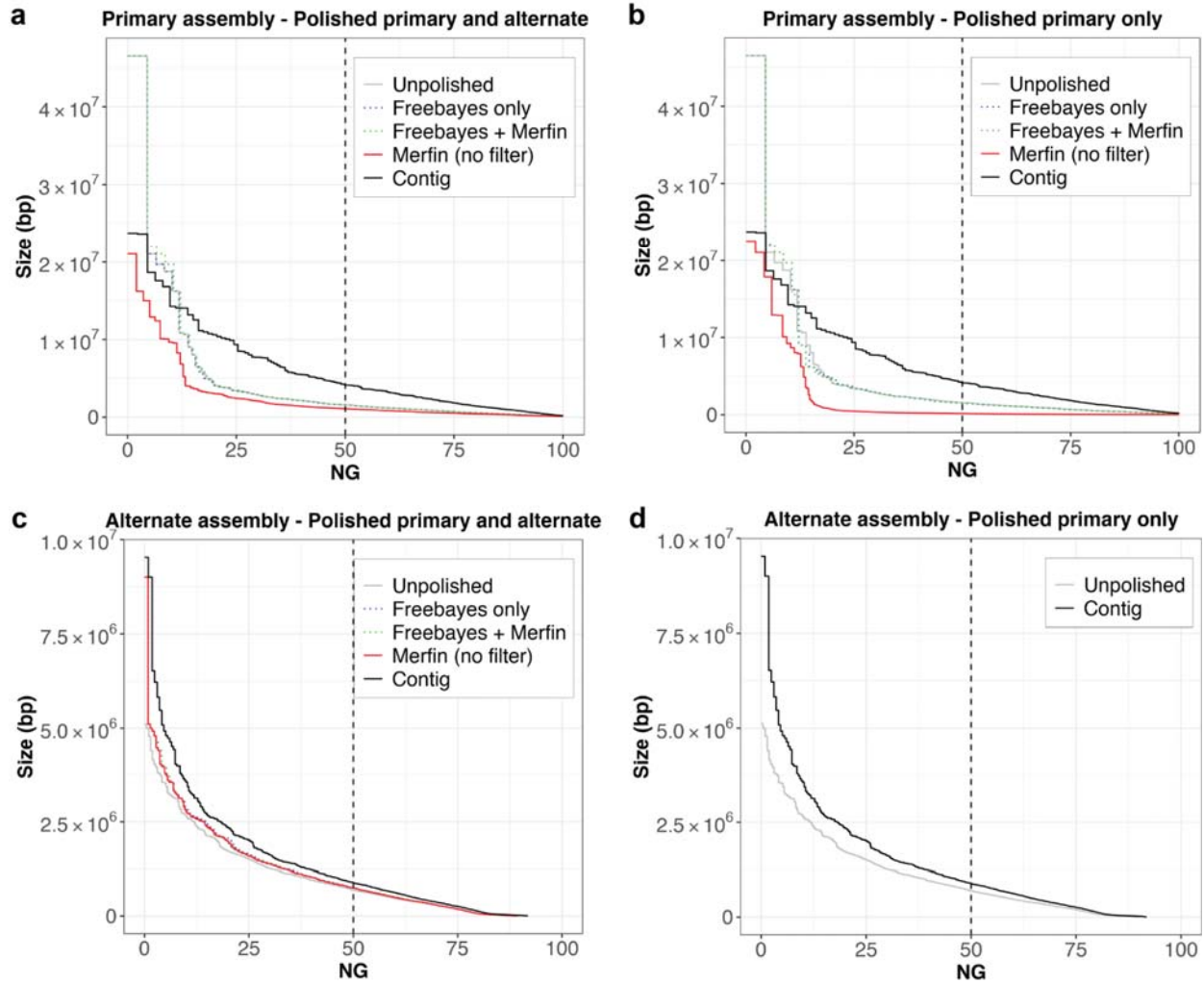


4

1 **Supplementary Figure 5 | Haplotype phasing before and after polishing with Merfin.** In both  
 2 parental assemblies, the haplotypes remained fully phased, and the size of the blocks significantly  
 3 increased compared to the unpolished version (**a,b**) after polishing with Merfin (**c,d**). A theoretical human  
 4 genome size of 3.1 Gbp was used to normalize NG\* values.

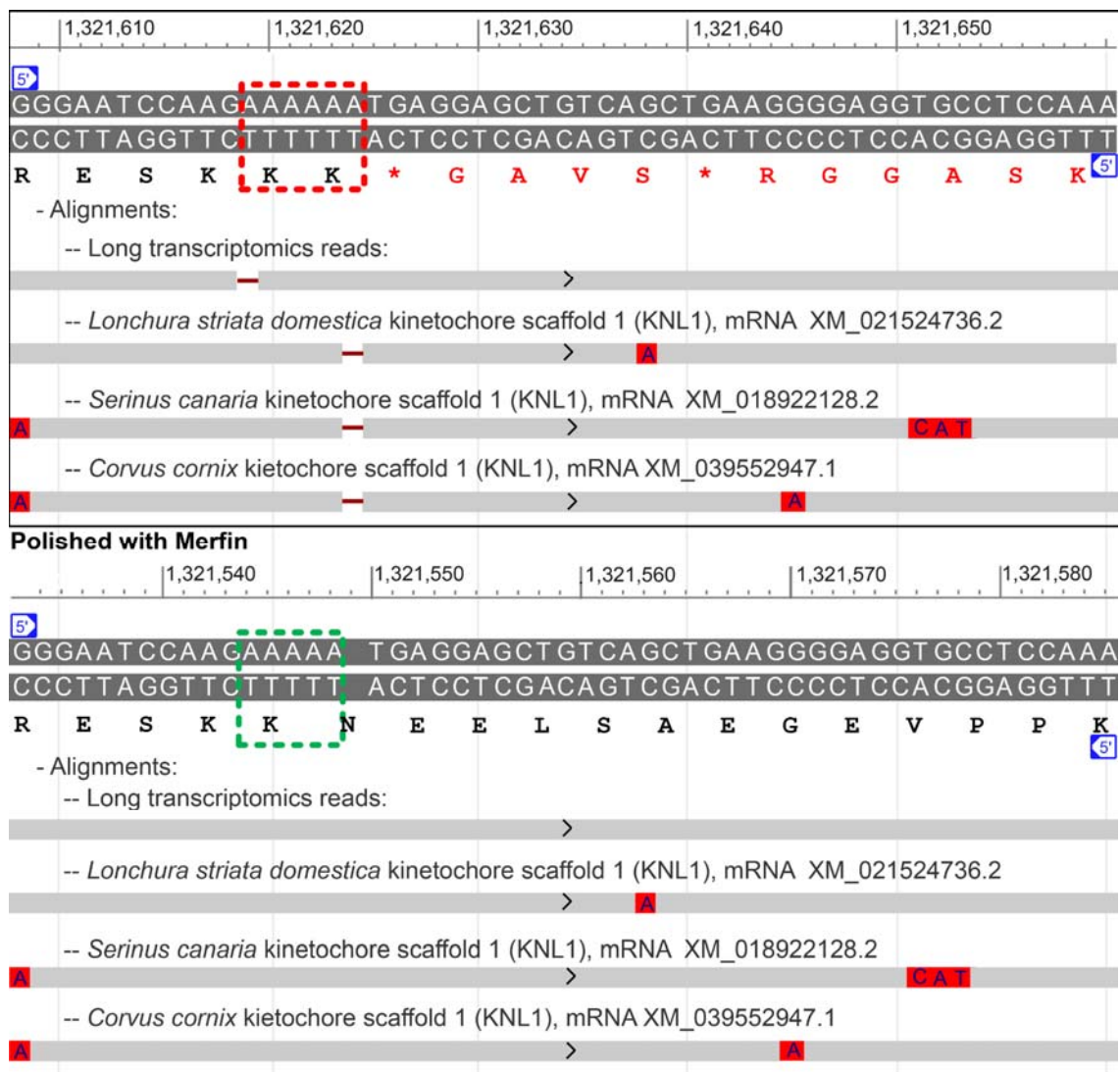


5  
 6 **Supplementary Figure 6 | VGP assembly pipeline.** Compared to the previous v1.6, the introduction of  
 7 Merfin in v1.7 (green) resulted in a minimal change of the workflow, but in a generalized improvement in  
 8 QV scores and gene annotations. Pipeline available at <https://github.com/VGP/vgp-assembly>.



1  
2 **Supplementary Figure 7 | Phase block analysis of zebra finch pseudo-haplotype assembly.** **a**, Phase  
3 blocks in the primary assembly after mapping the reads to both the primary and alternate assemblies. **b**,  
4 Phase blocks in the primary assembly after mapping the reads to both the primary only. **c**, Phase blocks in  
5 the alternate assembly after mapping the reads to both the primary and alternate assemblies. **d**, Phase  
6 blocks in the alternate assembly. In all cases, the application of Merfin filtering minor heterozygous  
7 variants (green) leads to block sizes better or comparable to prior polishing methods alone (blue).  
8 Unpolished assembly in gray. Results of Merfin without filtering in red. A genome size of  $\sim 1.03$  Gbp  
9 derived from Genomescope2 was used to normalize NG\* values.





1  
 2 **Supplementary Figure 8 | Effect of merfin correction on the kinetochore scaffold 1 protein (KNL1)**  
 3 **annotation. a,** Deleterious presence of an extra A around position 1,321,620 of scaffold\_7 (red box) in  
 4 the polished, non-merfin-corrected sequence is indicated by a 1-base gap in the alignments of zebra finch  
 5 PacBio IsoSeq SRR8695295.20794.1 and KNL1 transcripts from three other Passeriformes songbirds.  
 6 This insertion causes a disruption in the frame and a premature stop codon in the translated sequence (see  
 7 amino acid sequence in red). **b,** Corresponding span in the merfin-corrected assembly, with gapless  
 8 alignments of the IsoSeq read and Passeriformes transcripts, and uninterrupted translation.

## 1 **References**

- 2 1. Olson, N. D. *et al.* precisionFDA Truth Challenge V2: Calling variants from short- and long-reads in  
3 difficult-to-map regions. *bioRxiv* 2020.11.13.380741 (2021) doi:10.1101/2020.11.13.380741.
- 4 2. Koboldt, D. C. Best practices for variant calling in clinical sequencing. *Genome Med.* **12**, 91 (2020).
- 5 3. Guo, Y., Ye, F., Sheng, Q., Clark, T. & Samuels, D. C. Three-stage quality control strategies for  
6 DNA re-sequencing data. *Brief. Bioinform.* **15**, 879–889 (2014).
- 7 4. Giani, A. M., Gallo, G. R., Gianfranceschi, L. & Formenti, G. Long walk to genomics: History and  
8 current approaches to genome sequencing and assembly. *Comput. Struct. Biotechnol. J.* **18**, 9–19  
9 (2020).
- 10 5. Rhie, A. *et al.* Towards complete and error-free genome assemblies of all vertebrate species. *Nature*  
11 **592**, 737–746 (2021).
- 12 6. Wenger, A. M. *et al.* Accurate circular consensus long-read sequencing improves variant detection  
13 and assembly of a human genome. *Nat. Biotechnol.* **37**, 1155–1162 (2019).
- 14 7. Watson, M. & Warr, A. Errors in long-read assemblies can critically affect protein prediction.  
15 *Nature biotechnology* vol. 37 124–126 (2019).
- 16 8. Nurk, S. *et al.* HiCanu: accurate assembly of segmental duplications, satellites, and allelic variants  
17 from high-fidelity long reads. *Genome Res.* **30**, 1291–1305 (2020).
- 18 9. Walker, B. J. *et al.* Pilon: an integrated tool for comprehensive microbial variant detection and  
19 genome assembly improvement. *PLoS One* **9**, e112963 (2014).
- 20 10. Hepler NL, Delaney N, Brown M, Smith ML, Katzenstein D, Paxinos EE, Alexander D. An  
21 Improved Circular Consensus Algorithm with an Application to Detect HIV-1 Drug-Resistance  
22 Associated Mutations (DRAMs). *Poster presentation*.
- 23 11. Vaser, R., Sović, I., Nagarajan, N. & Šikić, M. Fast and accurate de novo genome assembly from  
24 long uncorrected reads. *Genome Res.* **27**, 737–746 (2017).
- 25 12. McKenna, A. *et al.* The Genome Analysis Toolkit: a MapReduce framework for analyzing next-  
26 generation DNA sequencing data. *Genome Res.* **20**, 1297–1303 (2010).

- 1 13. Garrison, E. & Marth, G. Haplotype-based variant detection from short-read sequencing. *arXiv [q-*  
2 *bio.GN]* (2012).
- 3 14. Poplin, R. *et al.* A universal SNP and small-indel variant caller using deep neural networks. *Nat.*  
4 *Biotechnol.* **36**, 983–987 (2018).
- 5 15. Li, H. A statistical framework for SNP calling, mutation discovery, association mapping and  
6 population genetical parameter estimation from sequencing data. *Bioinformatics* **27**, 2987–2993  
7 (2011).
- 8 16. Rhie, A., Walenz, B. P., Koren, S. & Phillippy, A. M. Merqury: reference-free quality, completeness,  
9 and phasing assessment for genome assemblies. *Genome Biol.* **21**, 245 (2020).
- 10 17. Mapleson, D., Garcia Accinelli, G., Kettleborough, G., Wright, J. & Clavijo, B. J. KAT: a K-mer  
11 analysis toolkit to quality control NGS datasets and genome assemblies. *Bioinformatics* **33**, 574–576  
12 (2017).
- 13 18. Kundu, R., Casey, J. & Sung, W.-K. HyPo: Super Fast & Accurate Polisher for Long Read Genome  
14 Assemblies. *Cold Spring Harbor Laboratory* 2019.12.19.882506 (2019)  
15 doi:10.1101/2019.12.19.882506.
- 16 19. Jain, C. *et al.* Weighted minimizer sampling improves long read mapping. *Bioinformatics* **36**, i111–  
17 i118 (2020).
- 18 20. Jain, C., Rhie, A., Hansen, N., Koren, S. & Phillippy, A. M. A long read mapping method for highly  
19 repetitive reference sequences. *bioRxiv* 2020.11.01.363887 (2020) doi:10.1101/2020.11.01.363887.
- 20 21. Phillippy, A. M., Schatz, M. C. & Pop, M. Genome assembly forensics: finding the elusive mis-  
21 assembly. *Genome Biol.* **9**, R55 (2008).
- 22 22. Nurk, S. *et al.* The complete sequence of a human genome. *bioRxiv* (2021).
- 23 23. Mitchell R. Vollger, Xavi Guitart, Philip C. Dishuck, Ludovica Mercuri, William T. Harvey, Ariel  
24 Gershman, Mark Diekhans, Arvis Sulovari, Katherine M. Munson, Alexandra M. Lewis, Kendra  
25 Hoekzema, David Porubsky, Ruiyang Li, Sergey Nurk, Sergey Koren, Karen H. Miga, Adam M.  
26 Phillippy, Winston Timp, Mario Ventura, Evan E. Eichler. Segmental duplications and their

- 1 variation in a complete human genome. *bioRxiv* (2021).
- 2 24. Gershman, A. *et al.* Epigenetic patterns in a complete human genome. *bioRxiv* (2021).
- 3 25. Mc Cartney Michael Alonge+ Chirag Jain Giulio Formenti Arkarachai Fungtammasan Kishwar  
4 Shafin Benedict Paten Karen H. Miga Andrey V. Bzikadze Alla Mikheenko Glennis A. Logsdon  
5 Jonathan MD Wood Kerstin Howe Alaina Shumate Ivan Sović Justin M. Zook Sergey Koren Adam  
6 M. Phillippy Arang Rhie, A. M. Chasing Perfection: Validation and Polishing Strategies for  
7 Telomere-to-Telomere Genome Assemblies. *bioRxiv* (2021).
- 8 26. Krusche, P. *et al.* Best practices for benchmarking germline small-variant calls in human genomes.  
9 *Nat. Biotechnol.* **37**, 555–560 (2019).
- 10 27. Ranallo-Benavidez, T. R., Jaron, K. S. & Schatz, M. C. GenomeScope 2.0 and Smudgeplot for  
11 reference-free profiling of polyploid genomes. *Nat. Commun.* **11**, 1432 (2020).
- 12 28. Cheng, H., Concepcion, G. T., Feng, X., Zhang, H. & Li, H. Haplotype-resolved de novo assembly  
13 using phased assembly graphs with hifiasm. *Nat. Methods* **18**, 170–175 (2021).
- 14 29. Huddleston, J. *et al.* Discovery and genotyping of structural variation from long-read haploid  
15 genome sequence data. *Genome Res.* **27**, 677–685 (2017).
- 16 30. Miga, K. H. *et al.* Telomere-to-telomere assembly of a complete human X chromosome. *Nature* **585**,  
17 79–84 (2020).
- 18 31. Zook, J. M. *et al.* Extensive sequencing of seven human genomes to characterize benchmark  
19 reference materials. *Scientific data* vol. 3 160025 (2016).
- 20 32. Kolmogorov, M., Yuan, J., Lin, Y. & Pevzner, P. A. Assembly of long, error-prone reads using  
21 repeat graphs. *Nat. Biotechnol.* **37**, 540–546 (2019).
- 22 33. Koren, S. *et al.* De novo assembly of haplotype-resolved genomes with trio binning. *Nat. Biotechnol.*  
23 (2018) doi:10.1038/nbt.4277.
- 24 34. Li, H. *et al.* A synthetic-diploid benchmark for accurate variant-calling evaluation. *Nat. Methods* **15**,  
25 595–597 (2018).
- 26 35. Wagner, J. *et al.* Benchmarking challenging small variants with linked and long reads. *bioRxiv*

- 1           2020.07.24.212712 (2020) doi:10.1101/2020.07.24.212712.
- 2   36. Chin, C.-S. *et al.* Phased diploid genome assembly with single-molecule real-time sequencing. *Nat.*  
3       *Methods* **13**, 1050–1054 (2016).
- 4   37. Gnomon - the NCBI eukaryotic gene prediction tool.  
5       [https://www.ncbi.nlm.nih.gov/genome/annotation\\_euk/gnomon/](https://www.ncbi.nlm.nih.gov/genome/annotation_euk/gnomon/).
- 6   38. Benjamini, Y. & Speed, T. P. Summarizing and correcting the GC content bias in high-throughput  
7       sequencing. *Nucleic Acids Res.* **40**, e72 (2012).
- 8   39. Yang, C. *et al.* Evolutionary and biomedical insights from a marmoset diploid genome assembly.  
9       *Nature* (2021) doi:10.1038/s41586-021-03535-x.
- 10 40. Cheng, H., Concepcion, G. T., Feng, X., Zhang, H. & Li, H. Haplotype-resolved de novo assembly  
11       with phased assembly graphs. *arXiv [q-bio.GN]* (2020).
- 12 41. Robinson, J. T. *et al.* Integrative genomics viewer. *Nat. Biotechnol.* **29**, 24–26 (2011)