Supplementary Information: Fully phased sequence of a diploid human genome determined de novo from the DNA of a single individual

Ilya Soifer[1], Nicole L. Fong[1], Nelda Yi[1], Andrea T. Ireland[1], Irene Lam[1], Matthew Sooknah[1], Jonathan S. Paw[1], Paul Peluso[2], Gregory T. Concepcion[2], David Rank[2], Alex R. Hastie[3], Vladimir Jojic[1], J. Graham Ruby[1], David Botstein[1], and Margaret A. Roy[1]

[1]Calico Labs, South San Francisco, CA 94080

[2]Pacific Biosciences, Menlo Park, CA 94025

[3]Bionano Genomics, San Diego, CA 92121

Contents

1 Assembly from PacBio reads

1.1 Running FALCON pipeline

FALCON, the open-source phased diploid assembler introduced in Chin *et al.* (2016), was used to assemble PacBio reads. Configuration settings used in FALCON runs were as follows:

1.1.1 FALCON parameters

```
[General]
input_fofn = input.fofn
input_type = raw

length_cutoff = 5000
length_cutoff_pr = 23000

pa_concurrent_jobs = 64
cns_concurrent_jobs = 384
ovlp_concurrent_jobs = 384

pa_HPCdaligner_option =  -v -dal128 -t16 -e0.75 -M24 -l4800 -k18 -h480 -w8 -s100
ovlp_HPCdaligner_option =  -v -dal128  -M24 -k24 -h1024 -e.96 -l2500 -s100

pa_DBsplit_option = -x500 -s400
ovlp_DBsplit_option = -s400

falcon_sense_option = --output_multi --min_idt 0.70 --min_cov 4 \
                      --max_n_read 200 --n_core 8
falcon_sense_skip_contained = False

overlap_filtering_setting = --max_diff 60 --max_cov 60 --min_cov 0 --n_core 12

pwatcher_type = fs_based
job_type = slurm
jobqueue = standard
sge_option_da = -p %(jobqueue)s --nodes=1 --ntasks=1 \
                      --cpus-per-task=4 --mem-per-cpu=8g
sge_option_la = -p %(jobqueue)s --nodes=1 --ntasks=1 \
                      --cpus-per-task=16 --mem-per-cpu=8g
```

```
sge_option_pda = -p %(jobqueue)s --nodes=1 --ntasks=1 \
                         --cpus-per-task=6 --mem-per-cpu=8g
sge_option_pla = -p %(jobqueue)s --nodes=1 --ntasks=1 \
                         --cpus-per-task=16 --mem-per-cpu=8g
sge_option_fc = -p %(jobqueue)s --nodes=1 --ntasks=1 \
                         --cpus-per-task=16 --mem-per-cpu=8g
sge_option_cns = -p %(jobqueue)s --nodes=1 --ntasks=1 \
                         --cpus-per-task=8 --mem-per-cpu=8g
```

1.1.2 Quiver parameters

```
[General]
pwatcher_type = fs_based
job_type = slurm
job_queue = standard

[Unzip]
input_bam_fofn = input_bam.fofn
smrt_bin=/sw/bin/
sge_quiver = -p standard --nodes=1 --ntasks=1 \
        --cpus-per-task=16 --mem-per-cpu=16g
sge_track_reads = -p standard --nodes=1 --ntasks=1 \
        --cpus-per-task=8 --mem-per-cpu=16g
quiver_concurrent_jobs = 16
```

1.1.3 FALCON-Unzip parameters

```
[General]
pwatcher_type = fs_based
job_type = slurm
job_queue = tbmem
jobqueue = tbmem

[Unzip]
input_fofn = input.fofn
input_bam_fofn = input_bam.fofn

smrt_bin=/sw/bin/

sge_phasing = -p standard --nodes=1 --ntasks=1 \
```

```
                  --cpus-per-task=8 --mem-per-cpu=8g
sge_quiver = -p standard --nodes=1 --ntasks=1 \
                  --cpus-per-task=16 --mem-per-cpu=8g
sge_track_reads = -p standard --nodes=1 --ntasks=1 \
                  --cpus-per-task=8 --mem-per-cpu=8g
sge_blasr_aln = -p standard --nodes=1 --ntasks=1 \
                  --cpus-per-task=16 --mem-per-cpu=8g
sge_hasm = -p tbmem --nodes=1 --ntasks=1 \
                  --cpus-per-task=40 --mem-per-cpu=16g

unzip_blasr_concurrent_jobs = 64
unzip_phasing_concurrent_jobs = 64
quiver_concurrent_jobs = 64
```

1.2 Correction of errors using linked short reads

The strategy for correcting (polishing) the assembly using linked short reads consisted of three steps. First, short reads from linked-reads dataset from WI-38 cells were aligned to the assembly and variants were called (see 1.2.1). Second, discordant variants between the reads and the assembly were identified and the assembly allele was replaced with the discordant allele from the reads (see 1.2.2). Last, the efficiency of polishing was evaluated by comparing highly conserved genes in the unpolished and polished assembly (see 1.2.3).

1.2.1 Alignment of the linked short reads to the assembly

10x Genomics Chromium linked-read data were aligned to the PacBio assembly using the 10x Genomics Longranger pipeline (v 1.6.3) with default parameters. To accommodate the Longranger software, the PacBio assembly was combined into a limited number of FASTA entries using the Bionano optical maps (connecting scaffolded contigs with appropriately-sized stretches of Ns; see below) with further arbitrary concatenation into artificial scaffolds using stretches of 500 Ns. Linked read data (ca. 90-fold coverage) was aligned to the hybrid assembly using the Longranger pipeline with default parameters, identifying $>$ 4.5 million short variants. Since the linked reads were generated from the same genome as the PacBio assembly, discordant variants were assumed to reflect errors in either the assembly or variant-calling. As shown in Fig. S1A, the discordant variants were predominantly insertions and deletions, which was consistent with the most common error types described for PacBio based assemblies (Chaisson *et al.*, 2015; Gordon *et al.*, 2016).

1.2.2 Polishing pipeline

The polishing pipeline replaced reference alleles with their corresponding alternative alleles for all discordant variants called by Longranger with quality score > 50. In some cases when heterozygous location in a low complexity region in WI-38 genome was deleted in the assembly, Longranger called two *concordant* variants instead of a single bi-allelic *discordant* one (see Fig. S1C for an example). To identify these cases, pairs of concordant variants that are equivalent to a single bi-allelic discordant variant (e.g. two different insertions on the same position) were searched for using approach from (Assmus *et al.*, 2013). Since the parsimonious explanation for these cases was an erroneous deletion in the assembly of a heterozygous SNP in the genome, this set of cases was added to the Longranger-identified set of discordant variants and corrected as well. Fig. S1B and Table S1 summarize the results of polishing the assembly with different versions of the polishing pipeline and estimate the error rate of the assembly (see section 1.2.3 for the description of the evaluation pipeline).

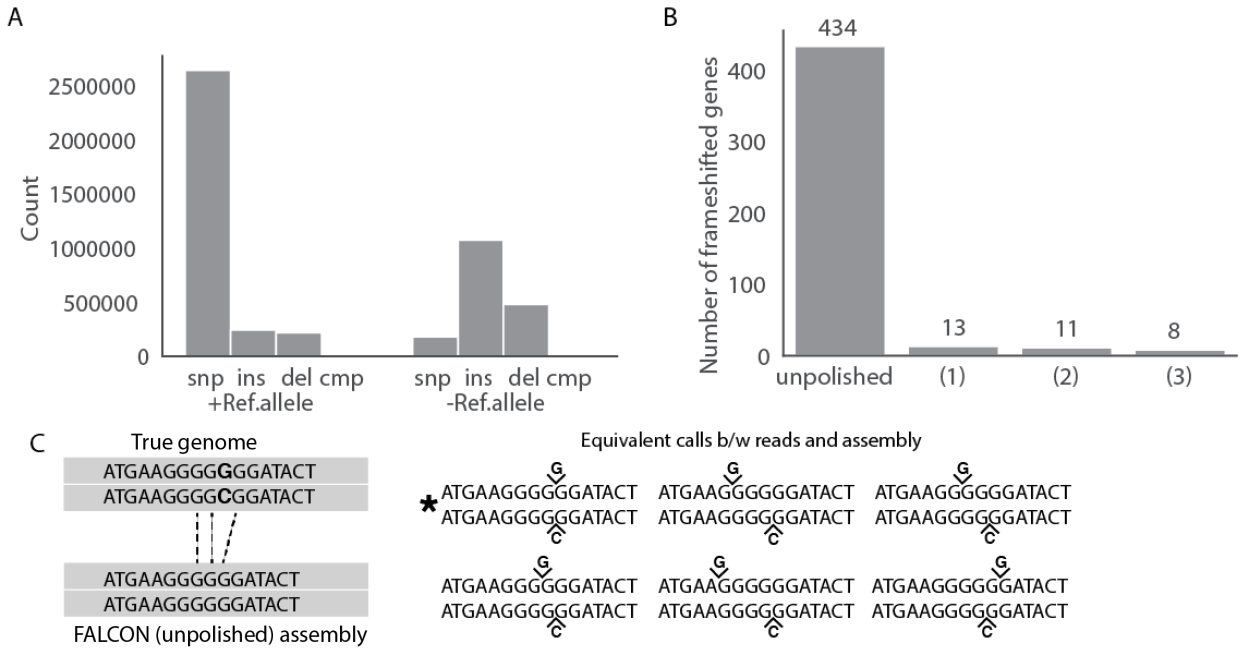

Figure S1: Correction of the PacBio assembly with linked short reads. (A) Number and type distribution of short variants identified between the linked reads and the assembly. (B) Evaluation of polishing pipeline with different parameters. Bar plot shows number of transcripts with frameshifts in the original assembly (unpolished PacBio assembly), after correcting all concordant and discordant variants between the linked reads and the assembly (1), after correcting only discordant variants that Longranger returned (2) and the final set (3) - correcting all discordant variants supplemented with a set of pairs of concordant variants that are equivalent to a discordant variant with two alleles. (C) Example: two concordant variants that were equivalent to a bi-allelic discordant variant. Consider an erroneous deletion of a heterozygous SNP in the assembly (left, compare WI38 genome and WI38 assembly). Since the SNP occurred in a low complexity region in the genome, there are six equivalent possible variant calls from the alignment of linked reads to the assembly (right panel). In this case, the one that requires a single deletion event in the assembly (*) was chosen under the assumption that it is the most parsimonious explanation.

The errors in the assembly were predominantly insertions and deletions. Since indels in the coding sequence introduce frameshifts and frameshifts are almost always deleterious to the function of the protein, very few real frameshifts in the essential genes in WI-38 genome were expected. Therefore, the effectiveness of polishing was evaluated by counting frameshifts in essential genes, similar to the approach used for evaluation of the recent gorilla genome assembly (Gordon *et al.*,

2016).

1.2.3 Evaluation of indels using essential-gene frameshifts

First, to identify a list of highly conserved transcripts, BUSCO (Simão *et al.*, 2015) based on OrthoDB (release 9) was used to define a set of highly conserved mammalian genes with a single isoform. BUSCO (version 3) defines 4097 single copy conserved genes. Genes lacking an annotated transcript in the RefSeq annotation v105 were filtered, leaving 3996 genes. Of this set, only genes containing a single annotated isoform in RefSeq were retained. This generated a list of 2153 transcripts that were not expected to have any indels in the assembly.

Second, these transcripts were aligned to the assembly using GMAP (Wu and Watanabe, 2005) version 2017-11-15 with parameters `-n 10`.

Third, expected protein sequences were extracted from the alignments, translated in three possible reading frames and assessed for the number of 10-mers from the reference sequence of the protein that were contained in each reading frame. If the coding sequence in the assembly contains a frameshift, more than a single reading frame in the protein would contain kmers from the reference protein sequence. Four hundred thirty-four transcripts in the original assembly contained frameshifts (Fig. S1B) suggesting an unpolished assembly error rate of approximately 1:2000 bp. When the polished genome was evaluated, only eight genes (out of 2153) contained frameshifts (Fig. S1B). These frameshift cases were manually examined as described in the main text (Table S2).

1.3 Correction of intermediate-size assembly errors

Since short-read assembly error detection is not efficient at detecting assembly errors longer than ten bases (Wenger *et al.*, 2019), a separate strategy was applied to correct these errors. Such errors can be detected as homozygous (discordant) structural variants between the assembly and the raw reads. Hence, the raw PacBio reads were aligned to the assembly and discordant structural variants were sought.

1.3.1 PacBio based assembly error detection.

To detect both concordant and discordant structural variants in the WI-38 assembly, PacBio read-based SV detection was applied. NGM-LR (Sedlazeck *et al.*, 2018) is an aligner that was shown

to be beneficial for detection of structural variations due to convex score function. NGM-LR was run with the parameters `-x pacbio -R 0.01`. The two pipelines Sniffles (Sedlazeck *et al.*, 2018) and PBSV (pbsv) were used to call structural variants. Sniffles was run with parameters `--report_seq --genotype`, and PBSV was run with default configuration file modified to return structural variants longer than 10 bases (`svlength=10`).

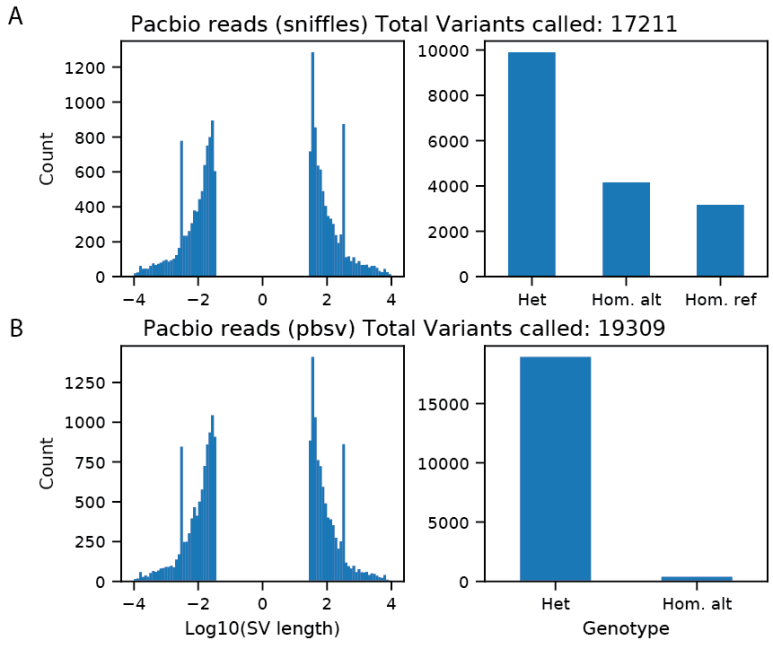

Figure S2: SV calls from different SV calling pipelines. (A) from Sniffles pipeline and (B) from PBSV pipeline. Left panels show length distribution of SVs (negative - deletion), and right panels show the genotype of the SVs.

The SV calls between the two pipelines were largely consistent, with PBSV calling slightly more structural variants. Encouragingly, the size distribution of all structural variants was largely similar to what had been previously observed (Fig. S2A,B, left panel). However, genotype calls from the two pipelines were strikingly different. While almost all of the SV calls from PBSV were heterozygous, there were substantially more SVs that Sniffles determined to be homozygous (Fig. S2A,B, right panel).

To examine the genotype calls, the alignments and the calls were manually examined in IGV browser. Many PacBio reads that did not support the discordant Sniffles SV calls, while in most cases all reads supported discordant PBSV calls. (Fig. S3). Thus, PBSV genotyping calls were deemed significantly more reliable for the versions of the software used at that time.

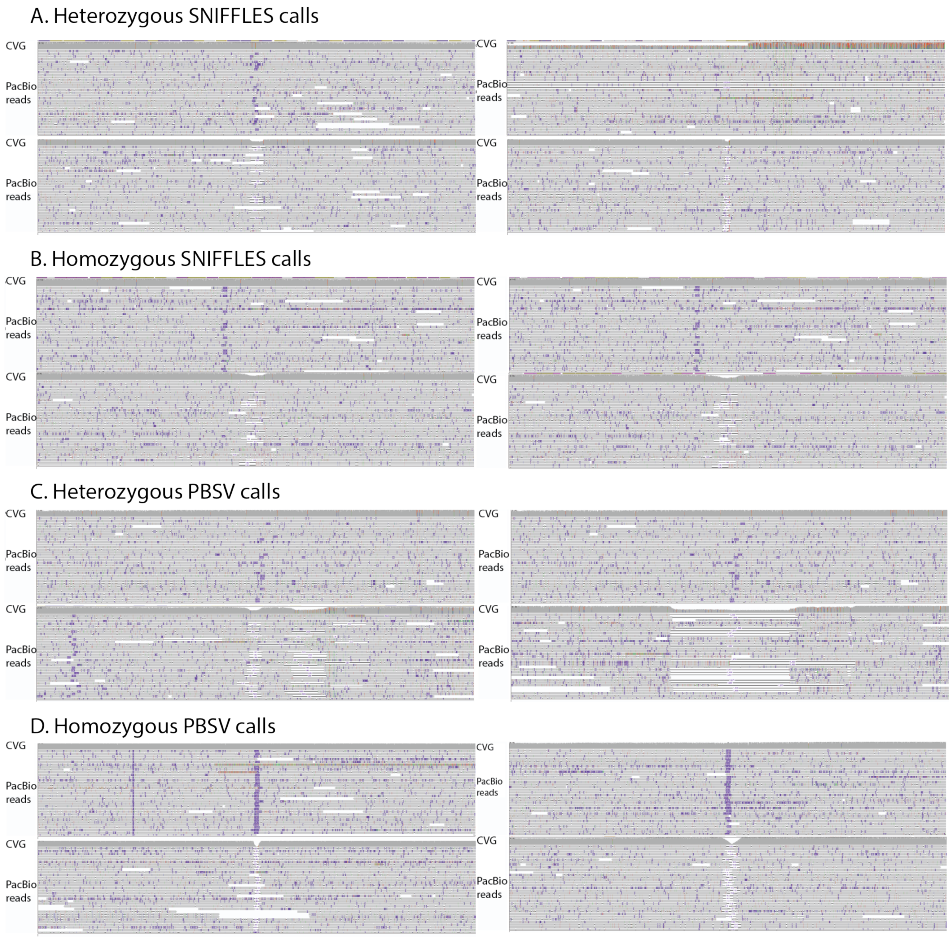

Figure S3: IGV screenshots of representative homozygous and heterozygous Sniffles (A,B) and PBSV (C,D) variants. In each panel the top row shows two examples of insertion call and the bottom row shows two examples of deletions call.

Genotyping thresholds of PBSV pipelines were slightly tuned to exclude assembly errors that were occurring in the regions covered with very few or a lot of PacBio reads (Fig. S4A). This generated a list of 477 discordant structural variants of various sizes (mosty small, Fig. S4B) with the total length of 27 kb, suggesting that the assembler was much more precise in the average precision of the assembly relative to the short range precision.

1.3.2 Classification of assembly errors.

To evaluate areas that were difficult to assemble, the assembly errors and structural variants were classified and compared the class distributions. Below are the classification criteria (note that the first three criteria are not mutually exclusive):

1. Low complexity areas: `dustmasker` masked 30% bases within the assembly error/structural variant, or 200 bases upstream of the error/variant, or 200 bases downstream of the error/-variant.

2. Collapsed repeats areas: if there are at least 20 bases out of 10 kb upstream or downstream of the assembly error/structural variant that have coverage of at least 150 (two-fold than the overall coverage of the assembly).

3. Missing/added tandem repeat: Tandem repeat finder (version 4.06) was run on the area ±200 bases around the assembly error/structural variant with parameters `2 5 7 80 10 50 2000 -l 6` and the assembly error/structural variant was overlapping the sequence identified as tandem repeat.

4. Other: None of the above.

As Fig. S4C shows, our assembly pipeline made errors predominantly in the areas prone to structural variation, although it made slightly more errors in the areas of collapsed repeats, and fewer errors than structural variations occured in the areas of low complexity.

2 Optical mapping and hybrid assembly

Hybrid assemblies that combined optical maps with the PacBio assembly were constructed in four steps.

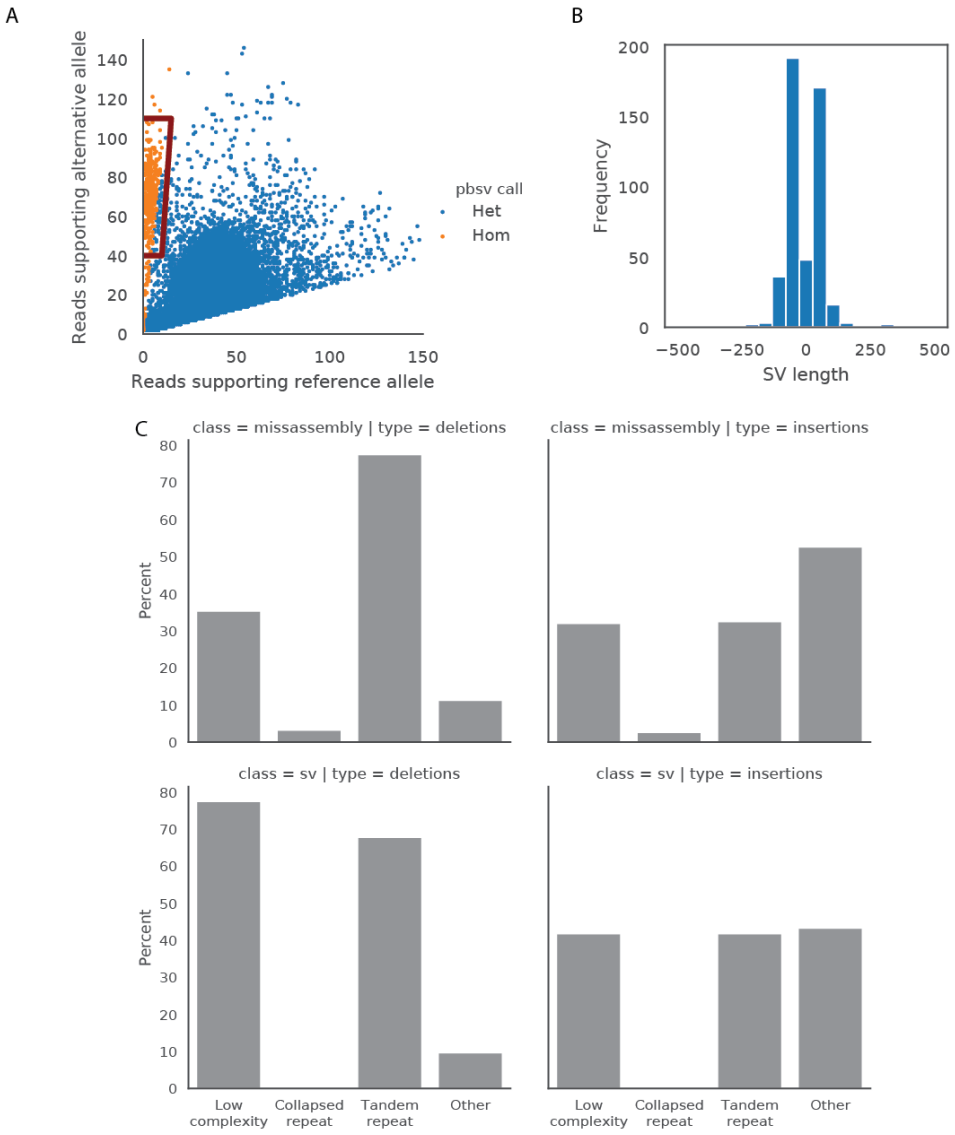

Figure S4: (A) Definition of assembly errors. Thresholds for calling SV homozygous (red box) compared to PBSV defaults (red dots - homozygous PBSV calls, blue dots - heterozygous). (B) Length distribution of assembly errors. (C) Classification of misasssemblies (discordant structural variants) and concordant structural variants.

First, consensus optical maps were assembled separately from the optical maps using Bionano Genomics Solve pipeline (v. 3.2.1_04122018) with parameters `-f 0.10 -d -U -T 240` -N 6 -j 240 `-y -i 5 -a` for BspQI and BssSI maps and `-d -U -N 6 -y -i 5 -F 1 -a` for DLE1 maps.

Second, hybrid scaffolding was performed from the polished PacBio assembly and BssSI and BspQI consensus maps.

Third, conflicts between the assemblies were resolved. After the initial execution of the first two steps, 134 conflicts with the BssSI consensus map and 133 conflicts with the BspQI consensus map were resolved in favor of the polished PacBio assembly. Two hundred six conflicts were resolved in favor of the optical maps. This resulted in a hybrid assembly with N50 of 92.8 Mb (versus 28.2 Mb in the PacBio only assembly).

Fourth, another round of scaffolding was performed by combining the output of the third step and the consensus optical map from DLE-1 chemistry that was previously shown to improve scaffold contiguity due to reduced DNA molecule cleavage during staining (Formenti *et al.*, 2018). At this point, we did not resolve conflicts and this resulted in slight increase in the assembly contiguity to N50 of 96Mb consistent with expectations.

In order to evaluate the completeness of our assemblies, we ran BUSCO pipeline (Simão *et al.*, 2015) on the polished PacBio assembly, the hybrid assembly and the 10x linked read-based assembly produced by Supernova pipeline (Fig. S5), which showed that the completeness of the hybrid assembly is comparable to the reference genome.

3 Chromosome sorting

3.1 Evaluation of sequencing libraries

FACS sorting was performed with parameters that would deposit a defined number of chromosomes per well of a 384-well plate (Main text Fig. 3), and each well was assigned unique library indexes. To determine the actual number of chromosomes per well, the chromosome sorted data was initially analyzed as follows:

1. Sequencing reads were aligned to GRCh38 using `bwa mem`.
2. For each well, we identify enriched chromosomes (see below).
3. For each chromosome and for every pair of wells containing this chromosome, we calculated the

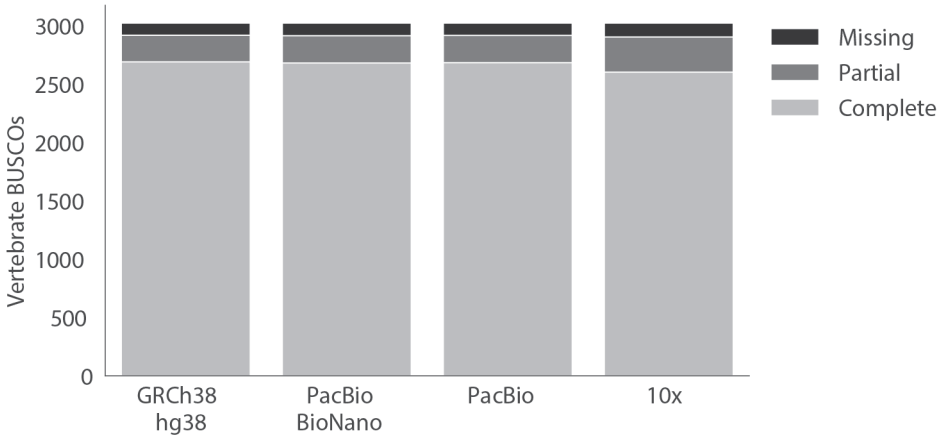

Figure S5: Results of the BUSCO pipeline run on the pure PacBio, PacBio + Bionano (hybrid) and Supernova assemblies. Vertebrate BUSCO set was used. Results of BUSCO run on the reference genome are shown for comparison.

number of agreements versus the number of disagreements on the alleles of the chromosome's SNPs.

3.1.1 Definition of enriched chromosomes

If a well contained a chromosome, it was determined to be "enriched" for that chromosome. For every well, we searched for a cutoff that would separate the chromosomes that were enriched (contained) in the well from the chromosomes that were not. The enrichment was measured by calculating a coverage of the chromosome relative to a set of the chromosomes. To calculate the relative coverage of the chromosome, we counted the number of reads that mapped uniquely to the chromosome per unit length (coverage) and divided it by the average coverage of a set of chromosomes. Let cov_i denote average coverage of chromosome i. We define relative coverage of chromosome i with respect to set S as:

$$\text{RelativeCoverage}(\text{cov}_i, S) = \frac{\text{cov}_i}{\sum_{j \in S} cov_j}. \quad (1)$$

Intuitively, the relative coverage of non-enriched chromosome to the set of non-enriched chromosomes is substantially lower than relative coverage of an enriched chromosome to the set of non-enriched chromosomes. This intuition motivates an algorithm which iterates over chromosomes in decreasing order of coverage in order to build a set of enriched chromosomes. Once a

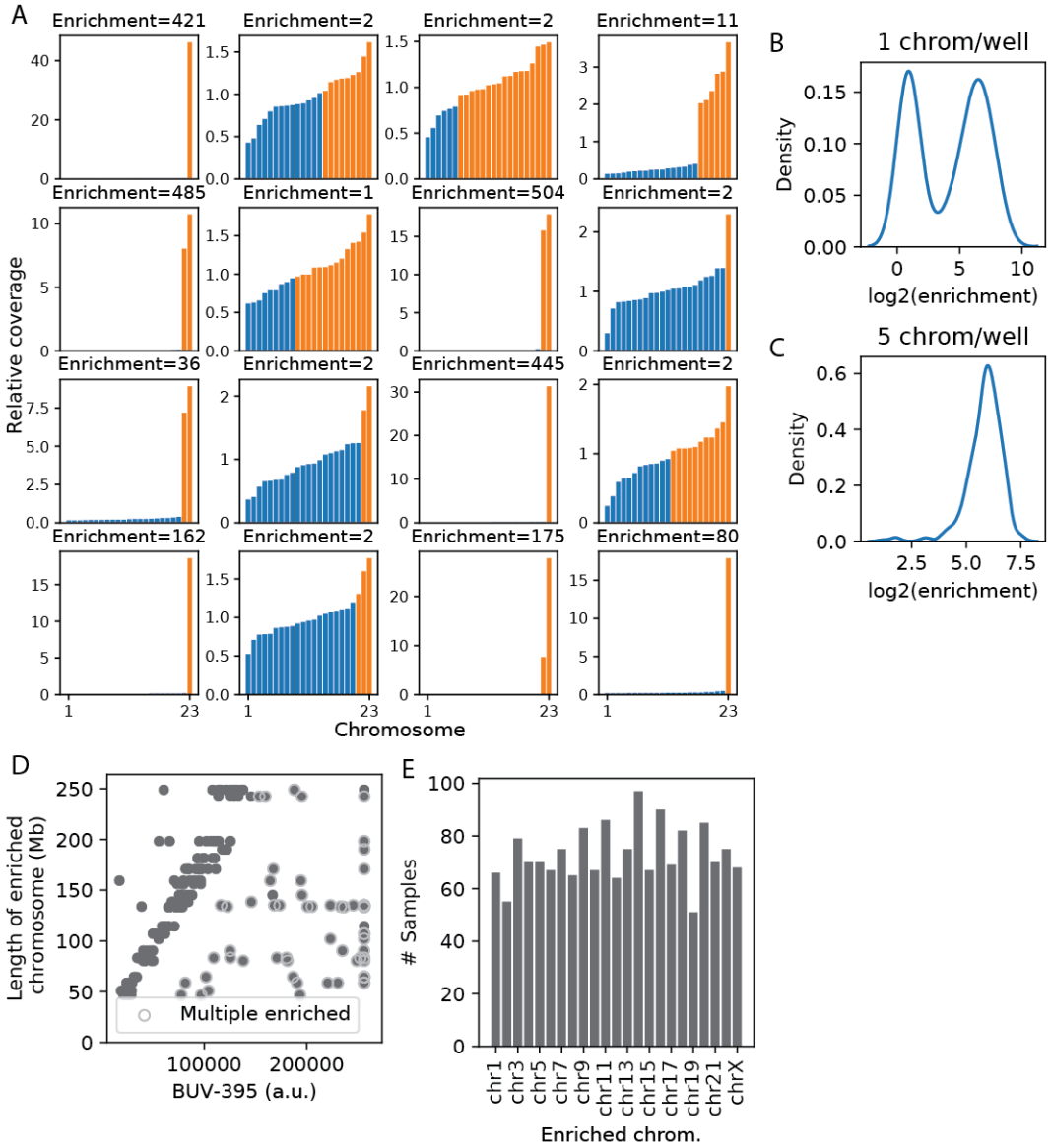

Figure S6: (A) Determination of enrichment threshold for 16 wells from library of 1 chromosome per well. Orange and blue bars denote enriched and non-enriched scaffolds, respectively. Note that the wells with enriched chromosomes (enrichment > 10) are quite distinct from non-enriched wells (enrichment ≤ 10). (B) Histogram of enrichments in 1 chromosome/well libraries. (C) Histogram of enrichments in 5 chromosome/well libraries. (D) The length of the enriched chromosome is well correlated to the intensity of the event. Note that the outliers from the trend are cases when multiple chromosomes are enriched in the well. (E) Enriched chromosome is relatively unbiased. Bar graph of number of times each chromosome was enriched.

chromosome cannot be safely added to the enriched set, it and all of the chromosomes with lower coverage, are deemed non-enriched. This algorithm is given below (Algorithm S1).

Algorithm S1 Determining enrichment cutoff

```
procedure FindCutoff
    covs := CalculateCoverages();
    covs := sort(covs);
    for cutoff:=length(covs)-1 do
        enriched := covs[cutoff+1:];
        non_enriched := covs[1:cutoff-1];
        c1 := RelativeCoverage(enriched, non_enriched);
        c2 := RelativeCoverage(covs[cutoff], non_enriched);
        if c1-c2 > c2-1 then return cutoff;▷ converged, coverage of the current chromosome is
like background
    return cutoff;
```

As shown in (Fig. S6A), this algorithm found separation between the enriched and the non-enriched chromosomes. We called a library in the well "good" if it achieved relative coverage of enriched chromosomes to the non-enriched chromosomes of at least 10 (see Equation 1). Indeed, in some libraries (Fig. S6B,C) there was clear separation between the libraries containing an enriched chromosome and those that did not. The enriched chromosome in the 1 chromosome/well samples was well correlated with the intensity of the sorted event (Fig. S6D) and there was no strong bias in the identities of the enriched chromosomes (Fig. S6E).

3.2 Defining linkage groups

The initial analysis of the chromosome-sorted data relative to the hybrid assembly was done similarly, but the reads were aligned to the hybrid assembly and for each sample we determined which scaffolds were enriched in it. Our initial goal was to assign scaffolds to chromosomes (linkage groups) using the fact that scaffolds that belong to the same chromosome should be enriched in the same wells. Determination of enrichment for short scaffolds was noisy, thus only 47 scaffolds longer than 5Mb were considered in the first step.

For each of the 47 scaffolds, we we calculated its enrichment pattern within wells (i.e. binary vectors with ones for samples enriched for the scaffold) and formed linkage groups between pairs of

enrichment patterns if pairwise correlation between the patterns was > 0.98 (Enrichment patterns between scaffolds in the same linkage group were virtually identical, Fig. S7B). After breaking the chimeric contig, we recovered 24 linkage groups. The last group contained a single scaffold that contained a gap > 11 Mb and less than 100 kb of sequence and was discarded.

To assign scaffolds longer than 100 kb to the linkage groups, we first calculated their enrichment pattern. Relative coverage of the scaffold was calculated using non-enriched long scaffolds as the background and scaffolds with relative coverage higher than 10 were considered enriched. To assign a scaffold to a linkage group, we calculated the p-value for the overlap between the enrichment pattern of the scaffold and the enrichment pattern of the linkage group assuming hyper-geometric distribution for the expected intersection as a null. The scaffold was assigned to a linkage group if there was a single linkage group with a p-value of overlap less than $3.1 * 10^{-5}$ (i.e. $< 2^{-15}$; Fig. S7C).

4 Phasing chromosomes using Expectation-Maximization

4.1 Representation of homologs

In our chromosome sorting approach, small number of chromosomes were sorted into individual wells of 384-well plates. Sequencing coverage from each individual well was sparse, so any one library from a single well contained only a subset of heterozygous SNPs, even for the enriched chromosomes. Since no single library could fully reconstruct a homolog, data from multiple libraries (corresponding to multiple wells) were aggregated through the construction of a probabilistic model. We pose the problem of homolog phasing as maximum likelihood estimation in this model.

We were ultimately interested in defining homolog assignment of each partially phased block to reduce the cost of the chromosome-sorted sequencing. Thus, we did not attempt to determine the allele present in the well for each heterozygous polymorphism, but, instead, only determined the alleles of SNPs known from the short-read-based calling to be heterozygous in the genome of WI-38. Simple genotype caller from SAMtools was used to determine the allele of each SNP present in the well.

For simplicity, for each well we only used SNPs with a single allele present in that well (those that the genotype caller determined to be 'homozygous'). This was true for the vast majority (>95% of SNPs) due to both the sparsity of chromosomes across the wells (in most wells, only a

single homolog was present) and the sparsity of sequencing across each chromosome in each well. Since we were phasing heterozygous SNPs on the genome assembly, our model assumed all SNPs to be biallelic, with one allele equal to the reference (the reference being the unphased WI-38 assembly).

For every SNP, we selected a 0 allele to be a reference allele. A homolog was represented by a sequence of binary values, one for each heterozygous site. We denoted value 0 as `Ref`, and value 1 as `Alt`.

As an example: if homolog A was defined by sequence $s_A = (\texttt{Ref}, \texttt{Ref}, \texttt{Alt}, \texttt{Alt}, \texttt{Ref}, ...)$ then homolog B was defined by sequence $s_B = (\texttt{Alt}, \texttt{Alt}, \texttt{Ref}, \texttt{Ref}, \texttt{Alt}, ...)$. However, this discrete representation was not well suited to optimization and did not provide information about the uncertainty of homolog reconstruction.

4.2 Probabilistic view of homolog and sample encoding

To be able to apply a continuous optimization algorithm, we introduced a probabilistic representation of the problem.

In this representation, our goal was to estimate variables $\mathbf{z} = (z_1, \ldots, z_k, \ldots)$ that equalled zero if the true allele at a particular site k of homolog A was equal to the reference allele, and one otherwise.

Since the level of noise varied between the wells and between the chromosomes sorted into wells, we introduced the uncertainty parameters $\mathbf{\Phi} = \{\Phi_l\}$ for every sample (well)/chromosome pair l.

Explicitly, if g_{lk} was the genotype call for a position k of sample l,

$$P(g_{lk} = z_k \mid \mathbf{z}, \mathbf{\Phi}) = P(g_{lk} = z_k | z_k, \Phi_l) = \Phi_l.$$

Intuitively, Φ_l is expected to be close to either zero or one, dependent on the homolog of the chromosome enriched in the well and to 0.5 if the well contains more than one or zero homologs of the chromosome.

Given a particular $\mathbf{z}$ and $\mathbf{\Phi}$, the likelihood of the observed genotype calls in all wells $\mathcal{G} = \{g_{lk}\}$ is

$$P(\mathcal{G} \mid \mathbf{z}, \mathbf{\Phi}) = \prod_{k,l} P(g_{lk} \mid z_k, \Phi_l) = \prod_{k,l} \Phi_l^{[g_{lk}=z_k]} (1 - \Phi_l)^{[g_{lk} \neq z_k]}, \tag{2}$$

where

$$[x] = \begin{cases} 1, & \text{if } x \text{ is true} \\ 0, & \text{otherwise.} \end{cases}$$

Hence, complete data log-likelihood is given by

$$L(\mathbf{\Phi}; \mathbf{z}, \mathcal{G}) = \log P(\mathcal{G}, \mathbf{z} \mid \mathbf{\Phi}) = \sum_{k,l} \left([g_{lk} = z_k] \log \Phi_l + [g_{lk} \neq z_k] \log(1 - \Phi_l) \right), \tag{3}$$

and the observed data log-likelihood (when z are not observed, but only their probabilities are estimated), is given by

$$L(\mathbf{\Phi}; \mathcal{G}) = \log P(\mathcal{G} \mid \mathbf{\Phi}) = \sum_{k} \log \left(\sum_{z_k = \{0,1\}} P(z_k) \prod_{l} P(g_{lk} \mid z_k, \mathbf{\Phi}) \right). \tag{4}$$

Our goal was to maximize the observed data log-likelihood with respect to parameters $\mathbf{\Phi}$. Expectation-Maximization (EM) algorithm was applied to accomplish this.

4.3 Maximum-likelihood estimation using Expectation-Maximization

Expectation-Maximization algorithm can be seen to optimize a lower bound on the observed log-likelihood

$$\begin{aligned} L(\mathbf{\Phi}; \mathcal{G}) &\geq \sum_{k} q(z_k) \log \prod_{l} P(g_{lk} \mid z_k, \mathbf{\Phi}) - \sum_{k} q(z_k) \log q(z_k) && (5) \\ &= E_{\mathbf{z}}[L(\mathbf{\Phi}; \mathbf{z}, \mathcal{G})] - \sum_{k} E_{z_k}[\log q(z_k)] && (6) \end{aligned}$$

where $q(z_k)$ is any probability distribution over $\{0, 1\}$ and E_{z_k} is the expectation calculated with respect to q_{z_k}.

$$E_{\mathbf{z}}[L(\mathbf{\Phi}; \mathbf{z}, \mathcal{G})] = \sum_{k} E_{z_k} \left(\log \prod_{l} P(g_{lk}, z_k \mid \Phi_l) \right)$$

.

The two steps of the EM algorithm increase this lower bound. The E-step of the algorithm computes, for each well k, the optimal $q(z_k)$ given current estimate of $\mathbf{\Phi}$. The M-step computes the optimal $\mathbf{\Phi}$ given current $q(z_k)$, across all wells k.

4.4 E-step

We denote the current estimate of $\mathbf{\Phi}$ as $\tilde{\mathbf{\Phi}}$. Given $\tilde{\mathbf{\Phi}}$, $q(z_k)$ which maximizes (6) is equal to $P(z_k \mid \tilde{\mathbf{\Phi}}, \mathcal{G})$. We derived closed form expression for this conditional probability.

$$\begin{aligned} P(z_k = 0 \mid \mathcal{G}, \tilde{\mathbf{\Phi}}) &= \frac{P(z_k = 0, \mathcal{G} \mid \tilde{\mathbf{\Phi}},)}{P(\mathcal{G} \mid \tilde{\mathbf{\Phi}})} \\ &= \frac{P(\mathcal{G} \mid z_k = 0, \tilde{\mathbf{\Phi}}) P(z_k = 0)}{P(\mathcal{G} \mid z_k = 0, \tilde{\mathbf{\Phi}}) P(z_k = 0) + P(\mathcal{G} \mid z_k = 1, \tilde{\mathbf{\Phi}}) P(z_k = 1)}, \end{aligned} \tag{7}$$

where,

$$P(\mathcal{G} \mid z_k = 0, \tilde{\mathbf{\Phi}}) = \prod_l \tilde{\Phi}_l^{[g_{lk} = z_k]} (1 - \tilde{\Phi}_l)^{[g_{lk} \neq z_k]},$$

Thus, assuming $P(z_k = 0) = P(z_k = 1) = 1/2$,

$$P(z_k = 0 \mid \tilde{\mathbf{\Phi}}, \mathcal{G}) = \frac{\prod_l \tilde{\Phi}_l^{[g_{lk} = z_k]} (1 - \tilde{\Phi}_l)^{[g_{lk} \neq z_k]}}{\prod_l \tilde{\Phi}_l^{[g_{lk} = z_k]} (1 - \tilde{\Phi}_l)^{[g_{lk} \neq z_k]} + \prod_l \tilde{\Phi}_l^{[g_{lk} \neq z_k]} (1 - \tilde{\Phi}_l)^{[g_{lk} = z_k]}}.$$

We derived $P(z_k = 1 \mid \mathcal{G}, \tilde{\mathbf{\Phi}})$ analogously.

4.5 M-step.

Given current $q(z_k)$, optimal $\mathbf{\Phi}$ which maximizes the bound (6) depends only on parts of the bound which involve $\mathbf{\Phi}$

$$E_{\mathbf{z}}[L(\mathbf{\Phi}; \mathbf{z}, \mathcal{G})] = \sum_k E_{z_k}(\log \prod P(g_{lk}, z_k \mid \mathbf{\Phi})) = \sum_{k,l} E_{z_k}(\log P(g_{lk}, z_k \mid \mathbf{\Phi}))$$

We expanded this expression to reveal explicit dependence on $\mathbf{\Phi}$.

$$\begin{aligned} E_{\mathbf{z}}[L(\mathbf{\Phi}; \mathbf{z}, \mathcal{G})] &= \sum_{k,l} E_{z_k}(\log P(g_{lk}, z_k)) \\ &= \sum_{g_{lk}=0} (q(z_k = 0) \log P(g_{lk} = z_k) + \; q(z_k = 1) \log P(g_{lk} \neq z_k)) \\ &+ \sum_{g_{lk}=1} (q(z_k = 1) \log P(g_{lk} = z_k) + \; q(z_k = 0) \log P(g_{lk} \neq z_k)) \\ &= \sum_{g_{lk}=0} (q(z_k = 0) \log \Phi_l + \; q(z_k = 1) \log(1 - \Phi_l)) \\ &+ \sum_{g_{lk}=1} (q(z_k = 1) \log \Phi_l + \; q(z_k = 0) \log(1 - \Phi_l)) \end{aligned} \tag{8}$$

finally we rearranged the (8) to make update derivation straightforward:

$$E_{\mathbf{z}}[L(\mathbf{\Phi};\mathbf{z},\mathcal{G})] = \sum_l \log\Phi_l \left(\sum_{g_{lk}=0} q(z_k=0) + \sum_{g_{lk}=1} q(z_k=1)\right) + \log(1-\Phi_l)\left(\sum_{g_{lk}=1} q(z_k=0) + \sum_{g_{lk}=0} q(z_k=1)\right). \tag{9}$$

Defining $S_1 \equiv \sum_{g_{lk}=0} q(z_k=0) + \sum_{g_{lk}=1} q(z_k=1)$ and $S_2 \equiv \sum_{g_{lk}=1} q(z_k=0) + \sum_{g_{lk}=0} q(z_k=1)$ and substituting into (9) we have

$$E_{\mathbf{z}}[L(\mathbf{\Phi};\mathbf{z},\mathcal{G})] = \sum_l \log\Phi_l S_1 + \log(1-\Phi_l)S_2. \tag{10}$$

Differentiating (10) with respect to Φ_l yields

$$\frac{\partial E_{\mathbf{z}}(L)}{\partial \Phi_l} = \frac{S_1}{\Phi_l} - \frac{S_2}{1-\Phi_l}. \tag{11}$$

Equating the partial derivative to zero yields:

$$\text{argmax}_{\Phi} E_{\mathbf{z}}(L) = \frac{S_1}{S_1+S_2}. \tag{12}$$

4.6 Pre-processing

We found that performance of the EM was sensitive to the input SNP set. Specifically, the following noise sources affected phasing:

1. Homozygous SNPs that were being called as heterozygous
2. SNPs that were being called in the areas of unresolved duplications in the assembly. In these areas, there were many SNPs that were being called as heterozygous but probably resulted from two homologous homozygous areas in the genome that were not resolved by the assembly.

To exclude these we:

1. Excluded all non-substitution SNPs or SNPs not passing Longranger filters. This retained 2,625,045 SNPs
2. Excluded heterozygous SNPs that were supported by less than 15 linked reads (given 90x coverage) and with ratio of ref./alt. larger than 2. This retained 2,136,579 SNPs.

3. Excluded SNPs that occured in the areas (of total size 27.4 Mb) of the assembly where the coverage of PacBio reads was larger than 120-fold (the average coverage of PacBio reads was 80-fold). This retained 2,062,365 SNPs.

Note that the filtering at this step was deliberately aggressive, since at the later stage the phasing information was aggregated within the phase block and phased the SNPs which could not be phased with certainty by chromosome sorting. Finally, we excluded SNPs called on scaffolds that were not assigned to linkage groups. These scaffolds generally corresponded to repetitive regions and therefore could have belonged to any of multiple chromosomes.

4.7 Initialization of the EM algorithm

EM was performed on each linkage group separately. The performance of the EM generally did not depend on the initialization point, but good initialization sped up optimization considerably. To initialize the EM, we defined a distance metric for pair of wells as follows: let $P_A = \{p_1^A, \ldots, p_n^A\}, P_B = \{p_1^B, \ldots, p_m^B\}$ be sets of polymorphic locations that were observed in wells A and B respectively. Let $\{g^A_{p_1^A}, \ldots, g^A_{p_n^A}\}, \{g^B_{p_1^B}, \ldots, g^B_{p_m^B}\}$ the observed alleles in both wells. We define a metric

$$d(A,B) = \frac{\sum_{l \in P_A \cap P_B} [g_l^A = g_l^B]}{|P_A \cap P_B|}$$

and calculate a matrix of pairwise distances between all pairs of wells enriched for the same linkage group. We used this distance matrix to perform agglomerative clustering (using `AgglomerativeClustering` tool from `sklearn`) on the wells using average linkage and clustering the wells into two clusters (presumable according to the homolog enriched in each well). We initialized Φ for each well as 0.7 and 0.3 dependent on the cluster that it belonged to. With this initialization, EM converged in less than 10 iterations. Initialization values in the intervals $(0.5, 1)$ and $(0, 0.5)$ respectively produced very similar final results, but affected the convergence rate.

5 Comparison of WI38 and GRCh38

Next, we compared the haplotypes of the final WI-38 assembly to the human reference genome GRCh38. As described in the main text, the fully-phased assembly allowed for straightforward genotyping of short and long polymorphisms by alignment of the WI-38 assembly to the reference genome. Below, we expand on the following points in the process:

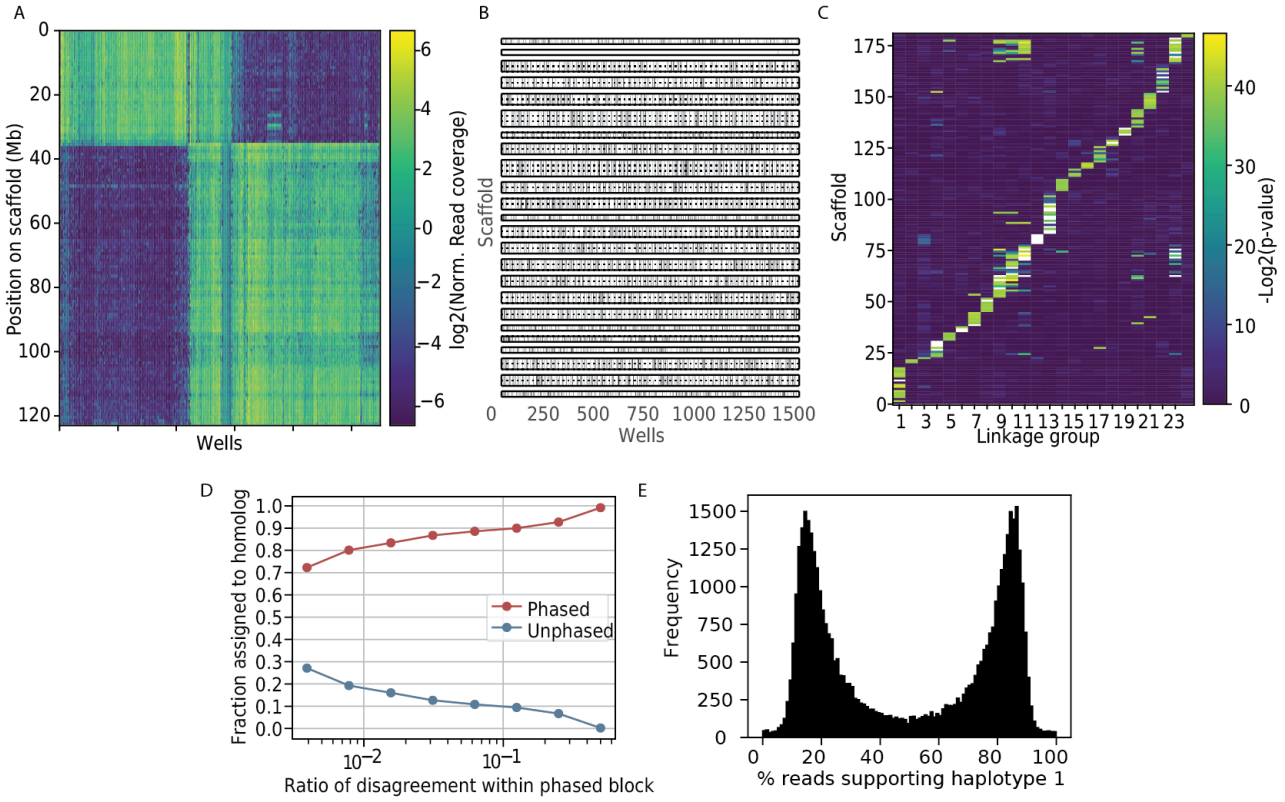

Figure S7: Identifying assembly errors that created a chimeric scaffold using chromosome sorted data. (A) Heatmap showing read coverage in each well enriched for the chimeric scaffold. Well patterns were clustered by hierarchical clustering. Note that the first 35 Mb of the scaffold were enriched in different set of wells when compared to the last 85 Mb. The scaffold was broken at position 35,274,265 nt. (B) Assigning long scaffolds to linkage groups. Black vertical lines denote wells where each scaffold was enriched, dashed lines separate individual scaffolds and black boxes denote linkage groups. (C) Assigning all scaffolds to linkage groups. Same as Fig. 5A in the main text, but all scaffolds longer than 100 kb are shown. (D) 90% of variants were phased using sorted chromosomes. Scatter plot showing fraction of phased and unphased variants for different thresholds of agreement fraction of homolog assignment between the individual variants inside a phased block. (E) CrossStitch pipeline was able to phase structural variants confidently. Figure shows histogram of percentage of PacBio reads supporting assignment of the structural variant to haplotype 1. Note that majority of SVs can be confidently assigned to the haplotype.

1. comparison of short variant calling that is based on the WI-38 assembly versus read-based short variant calling;

2. generation of the diploid WI-38 genome;

3. SV calling from the diploid WI-38 genome.

5.1 Short polymorphisms between GRCh38 and WI-38

We used paftools and Minimap2 Li (2018) to align diploid assembly to the reference genome in order to call short variants. We compared variants by alignment of the assemblies and variants called by aligning linked reads from WI-38 to the reference genome. Overall, the accuracy of the variant recall was very high – almost all variants called from short reads were also called by the assembly. The precision was also high, albeit somewhat lower than recall (Fig. S8). Notably, discrepancies tended to be concentrated in subtelomeric regions and around centromeres, where variants calls are expected to be challenging. Additionally, we observed a relatively large number of unique variants called from the assembly on the homopolymer stretches, as previously described (Wenger *et al.*, 2019).

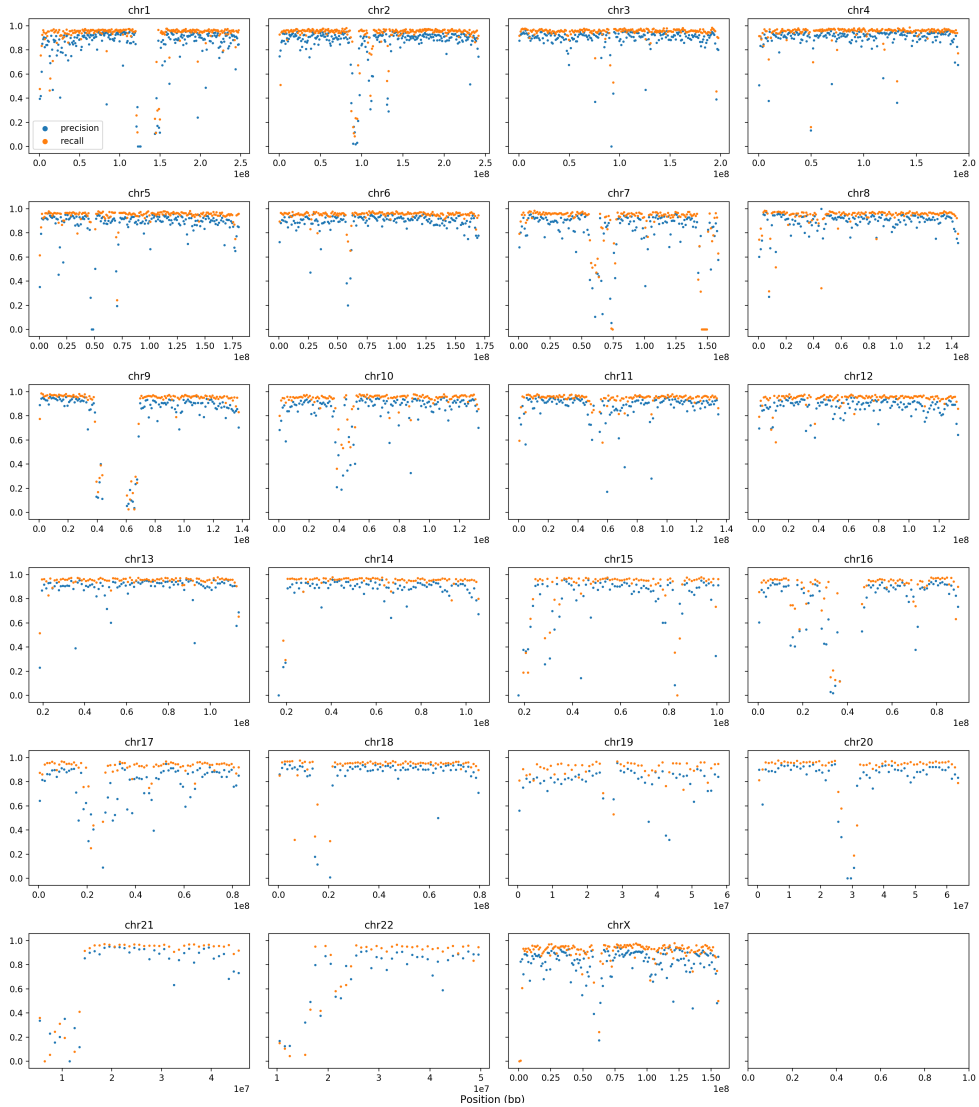

Figure S8: Disagreements of short 10x Genomics linked read variant calls (SNPs and indels of length ≤ 30) between the phased assembly and short linked reads aligned to GRCh38, binned by chromosomal position. Here we treat the 10x Genomics calls as ground truth, so "precision" refers to the fraction of assembly variants that are also present in 10x Genomics, and "recall" refers to the fraction of 10x Genomics variants that are also present in the assembly. Variants were called from the assembly using Minimap2 and paftools. Variants were called from 10x Genomics reads using Longranger. Only variants with QUAL$\geq$30 and FILTER=PASS were counted.

5.2 Structural variation between GRCh38 and WI-38

5.2.1 Generation of the diplpoid WI-38 genome

Shorter phase blocks deriving from linked-read technology (10X) were analyzed using data from chromosome sorting in order to generate chromosome-length haplotypes. Structural variants therefore remained mosly unphased by the efforts described in the previous sections. The CrossStitch pipeline was used to assign structural variants called by alignment of PacBio reads to the discovered phased blocks.

To phase structural variants using PacBio reads we first aligned PacBio reads to the partially phased assembly using NGM-LR and called structural variants using Sniffles. Next, Sniffles output was filtered in the following way:

1. Structural variants with abnormal PacBio read coverage of the SV breakpoint ($>$ 130-fold) or with abnormal ratio of support of alternative allele to reference allele ($alt_{coverage} < total_{coverage}/2 - 25$) were filtered out.

2. Structural variants spanning gaps on the assembly were filtered out.

The sequences of insertions and inversions were then refined using the realignment script in the CrossStitch pipeline and phased using CrossStitch with 10x phased SNPs as input with the following minor modifications:

1. Only SV calls with SV types `INS, DEL, INV` were used, other SVs (308) were filtered out as their sequence was poorly determined and thus hard to phase.

2. After phasing the SVs, CrossStitch removes overlapping SNPs and structural variants. The process was modified as follows: if a Sniffles SV call was within 100 bases of an Illumina SV call of longer than 10 bases, we filtered out the Sniffles SV call. Otherwise, if an Illumina SV call of less than 10 bases or a SNP/INDEL overlapped a PacBio SV call, we removed the Illumina call.

3. vcf2diploid script that outputs the two versions of the assembly was modified so that inversions were also included in the diploid genome.

As shown on Fig. S7E, the majority of the structural variants were confidently assigned to haplotype.

5.2.2 Alignments and filtering

Scaffolds of the diploid assembly were aligned to GRCh38 using Minimap with default parameters. The alignments were processed in the following steps:

1. Alignments of inversions sometimes had low mapping quality scores. We identified such alignments by the following criteria: a) shorter than 5,000 bp; b) flanking sequences aligned with quality > 30; and c) the flanking sequences aligned in the same order and orientation, the middle sequence was inverted relative to the flanking sequences. These alignments were given mapping quality of 60. This was done to prevent filtering of inverted alignments that could later be identified as inversions.

2. Alignments with mapping scores less than 30 were filtered out.

3. Alignments to the Y chromosome were filtered out, since the WI-38 is a female cell line.

4. Repetitive regions in the reference genome tend to align to multiple short regions of the assembly. These alignments can lead to spurious variant calls. Thus we filtered all alignments that did not align to at least 100 bases of the reference that align to a single assembly region.

5. Some regions of the assembly (predominantly in the repetitive regions) sometimes aligned to multiple locations. We filtered out all alignment segments that do not contain 100 bases of the assembly that have only a single alignment.

5.2.3 Calling insertions and deletions

Insertions and deletions were called from either the CIGAR string or when the aligner split the assembly sequence into two segments and the distance on the assembly between the ends of the segments was different from the distance on the reference ((Marçais *et al.*, 2018)). Insertions and deletions that spanned gaps in the assembly or reference were removed. Note, that we only called insertion or deletion if the direction of alignment of the consecutive segments was the same - i.e. if the two segments both aligned to the Watson or to the Crick strand of the reference genome. Cases where the insertion or deletion size was larger than 1 Mb, when the two consecutive alignment segments aligned to different strands of the genome and when the alignment segments were on separate chromosomes were treated separately (see 5.2.5)

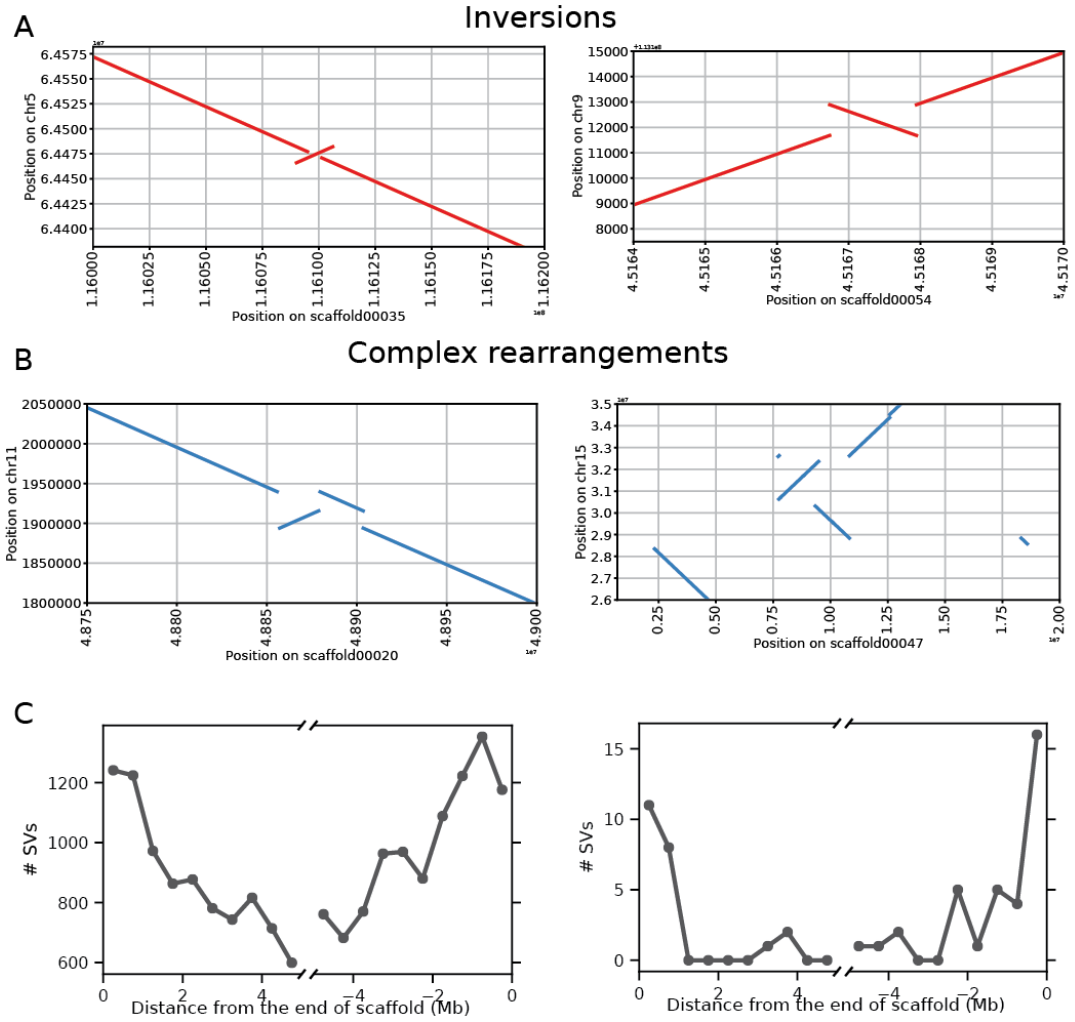

Figure S9: SV calls examples. (A) Examples of inversions. Alignment on the left was interpreted as inversion of chr5:64,474,000-64,477,000, alignment on the right - inversion of chr9:113,111,900-113,112,900. (B) Examples of complex SVs. Left - complex rearrangement at chr11:1,900,000-1,945,000, right - complex rearrangement at chr15:28,500,000-32,500,000. (C) Enrichment of structural variation close to ends of scaffolds for all (left) and large (right) structural variations.

5.2.4 Calling inversions

Inversions were called by identifying sections of the assembly that aligns to the reference in the opposite direction from the flanking sequences (Fig. S9A). To filter inversion calls arising from spurious alignments, we required that the flanking alignment segments were at least 10 times longer than the inverted segment.

5.2.5 Complex rearrangements

Complex SV calls included cases of alignment breakpoints that spanned two different chromosomes in the reference, distant sites (>1 Mb) on the same chromosome or when the assembly sequence was split into two segments that aligned to opposite strands of the reference genome. In many cases, however, such alignments happened in highly repetitive regions and were difficult to interpret (Fig. S9A), so we implemented an aggressive filtering strategy both to remove spurious alignments and facilitate interpretation:

1. All complex breakpoints between an unplaced contig and a chromosome were removed .

2. All breakpoints when one of the flanking alignments was shorter than 10 kb were removed. This excluded called inversions since they generated one short alignment by definition.

3. We identified pairs of alignment breakpoints with a distance less than 5Mb such that the intervening assembly segment aligned to a different region of the reference (translolcation) (Fig. S9B). The region between the two breakpoints was called "COMPLEX REARRANGEMENT"

4. Sets of close complex breakpoints within 100 kb were merged (this merged SVs on highly rearranged areas) (Fig. S9B).

5. Ends of contigs were highly enriched in SV calls (within 1 Mb; Fig. S9C). Contigs tend to terminate in repetitive regions that tend to be highly divergent between subjects, thus we hypothesized that many of the structural variations close to the ends of contigs arise from spurious alignments between highly divergent repetitive areas. These SVs were marked "Low confidence" in the provided output (Table S8).

6 SV calls from optical maps - related to Fig. 7

To confirm the structural variant calls we made from BssSI and DLE-1 optical maps respectively (Fig. S10) we aligned optical maps generated from the other labelling chemistries to the reference optical maps. As can be seen on Figure S10, the alignments of the optical maps from the other chemistries largely confirms the SV call from BssSI and DLE-1 maps. Optical maps, therefore, can facilitate SV calling in the difficult-to-sequence areas as well as resolve complex SV calls.

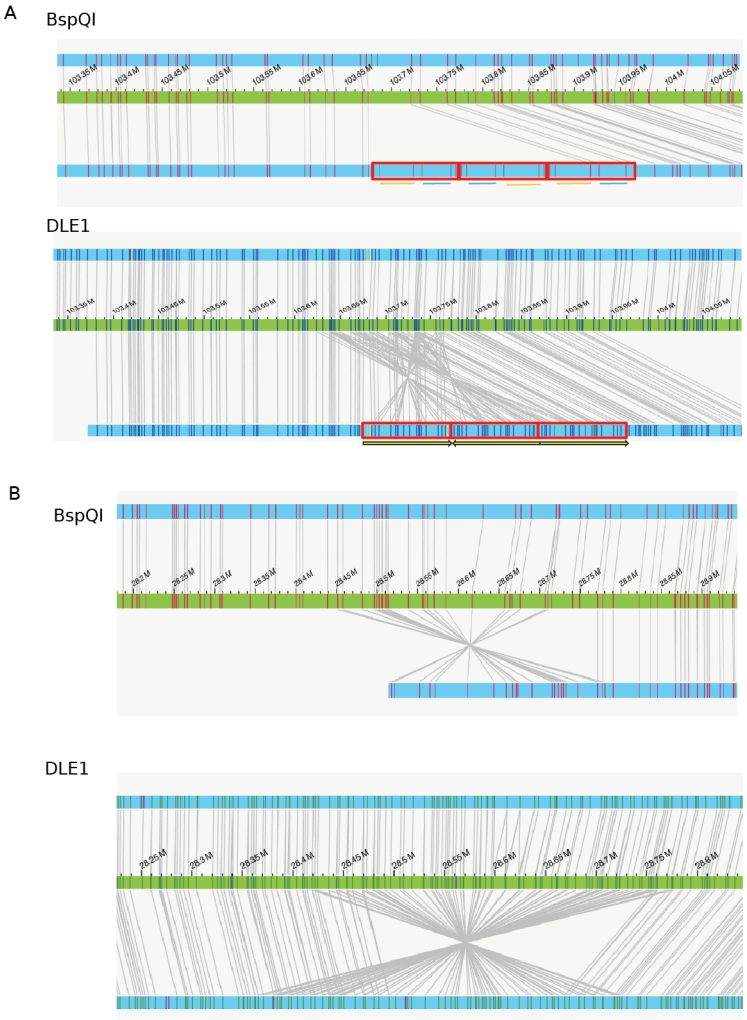

Figure S10: Optical maps from other labeling chemistries support SV calls made from optical maps on Fig. 6C,D. (A) Optical maps from the same region as Fig. 6C from BspQI and DLE1 chemistries. (B) Optical maps from the same region as Fig. 6D from BspQI and BssSI chemistries

7 Genome browser

We have developed a public website accompanying this paper, available at `wi38.research.calicolabs.com`.

This site provides an overview of the work, a list of data files available for download, and an interactive genome browser. The genome browser provides two ways to visualize and explore our results:

1. Phased scaffolds aligned to the human reference genome (GRCh38), with corresponding phased variants called between our assembly and the reference

2. Phased scaffolds aligned to the unphased hybrid assembly, with the corresponding phased variants called from sequencing data (10x and Pac Bio) aligned to the assembly

Files are also available for download directly from Google Cloud Storage at the following bucket: `gs://calico-wi38-assembly-public`.

8 Supplementary Tables

Table S1: Polishing summary

Table S2: Genes that contain frameshifts after polishing

Table S3: Summary of correction of intermediate-length misassemblies

Table S4: Unaligned sequences

Table S5: Statistics of sequencing libraries from 1 chrom/well sorting

Table S6: Statistics of sequencing libraries from 5 chrom/well sorting

Table S7: Statistics of SNVs relative to GRCh38

Table S8: SV calls relative to GRCh38

Table S9: Heterozygous SVs in WI-38 genome

Table S10: SRA submission part 1 - PacBio reads

Table S11: SRA submission part 2 - 10x long reads

Table S12: SRA submission part 3 - sorted chromosomes data

9 References

References

Assmus, J., J. Kleffe, A. O. Schmitt, and G. A. Brockmann, 2013 Equivalent Indels - Ambiguous Functional Classes and Redundancy in Databases. PLoS ONE **8**: 1–9.

Chaisson, M. J. P., R. K. Wilson, and E. E. Eichler, 2015 Genetic variation and the de novo assembly of human genomes. Nature Reviews Genetics **16**: 627–640.

Chin, C.-S., P. Peluso, F. J. Sedlazeck, M. Nattestad, G. T. Concepcion, *et al.*, 2016 Phased Diploid Genome Assembly with Single Molecule Real- Time Sequencing. Nature Methods **13**: 1050–1054.

Formenti, G., M. Chiara, L. Poveda, K. J. Francoijs, A. Bonisoli-Alquati, *et al.*, 2018 SMRT long reads and direct label and stain optical maps allow the generation of a high-quality genome assembly for the european barn swallow (Hirundo rustica rustica). GigaScience **8**: 1–9.

Gordon, D., J. Huddleston, M. J. P. Chaisson, C. M. Hill, Z. N. Kronenberg, *et al.*, 2016 Long-read sequence assemblyof the Gorilla Genome. Science **352**: aae0344.

Li, H., 2018 Minimap2: Pairwise alignment for nucleotide sequences. Bioinformatics **34**: 3094–3100.

Marçais, G., A. L. Delcher, A. M. Phillippy, R. Coston, S. L. Salzberg, *et al.*, 2018 MUMmer4: A fast and versatile genome alignment system. PLoS Computational Biology **14**: 1–14.

pbsv, 2018 PacBio structural variant (SV) calling and analysis tools. `https://github.com/PacificBiosciences/pbsv`.

Sedlazeck, F. J., H. Lee, C. A. Darby, and M. C. Schatz, 2018 Piercing the dark matter: bioinformatics of long-range sequencing and mapping. Nature Reviews Genetics : 329–346.

548 Simão, F. A., R. M. Waterhouse, P. Ioannidis, E. V. Kriventseva, and E. M. Zdob-
549 nov, 2015 BUSCO: Assessing genome assembly and annotation completeness with single-copy
550 orthologs. Bioinformatics **31**: 3210–3212.

551 Wenger, A. M., P. Peluso, W. J. Rowell, P.-C. Chang, R. J. Hall, *et al.*, 2019 Accurate
552 circular consensus long-read sequencing improves variant detection and assembly of a human
553 genome. Nature Biotechnology **37**: 1155–1162.

554 Wu, T. D., and C. K. Watanabe, 2005 GMAP: a genomic mapping and alignment program for
555 mRNA and EST sequences. Bioinformatics **21**: 1859–1875.