SOFTWARE

sgcocaller and comapr: personalised haplotype assembly and comparative crossover map analysis using single gametes

Ruqian Lyu[1,2], Vanessa Tsui[3,4], Wayne Crismani[3,4], Cynthia Liu[1,2], Heejung Shim[2†] and Davis J. McCarthy[1,2*†]

*Correspondence: dmccarthy@svi.edu.au
[1]Bioinformatics and Cellular Genomics, St. Vincent's Institute of Medical Research, Melbourne, Australia
[2]Melbourne Integrative Genomics, Faculty of Science, The University of Melbourne, Melbourne, Australia
Full list of author information is available at the end of the article
†Heejung Shim and Davis J. McCarthy supervised this work

Abstract

Profiling gametes of an individual enables the construction of personalised haplotypes and meiotic crossover landscapes, now achievable at larger scale than ever through the availability of high-throughput single-cell sequencing technologies. However, haplotyping single gametes from high-throughput single-cell DNA sequencing datasets using existing methods requires intensive processing. Here we introduce an efficient software toolset using modern programming languages for the common tasks of haplotyping haploid gamete genomes and calling crossovers (*sgcocaller*), and constructing and visualising individualised crossover landscapes (*comapr*) from single gametes. With additional data pre-possessing, the tools can also be applied to bulk sequenced samples.

Keywords: single-cell DNA sequencing; haplotype assembly; gamete sequencing; meiotic crossover; genome analysis

Background

Meiosis is a process required during sexual reproduction which generates gametes—egg or sperm cells—that contain half of the chromosomes of the parent cell [1]. Chromosome segregation and meiotic crossovers create abundant genetic diversity in the offspring. Meiotic crossovers are also required for accurate chromosome segregation [1] and reduced crossover rates are linked to increased risk for trisomy 21 [2, 3]. Past studies have shown that meiotic crossover rates and distributions vary greatly among species, sexes and even individuals [4, 5, 6, 7, 8, 9]. Variation in genetic factors, such as *PRDM9*, changes the distribution of crossover hotpots in human and mouse [10, 11, 6, 4]. Crossover-regulating genes also limit overall crossover rates [12, 13, 14]. Populations of different demographic backgrounds show differences in recombination landscapes, for example hotspot locations differ in African-American genetic maps compared with Europeans and West Africans [15, 16]. It has also been shown that meiotic crossovers vary among sexes and crossover hotspots can be sex-specific [17, 8, 5].

Single-cell DNA sequencing of gametes collected from an individual can be used to construct personalised meiotic crossover distributions [9, 7]. The scalability of modern single-cell assays, especially droplet-based platforms, has made it possible to profile thousands of sperm cells per individual in one experiment [7]. Assaying

gametes from females is substantially more challenging at scale, but nevertheless technological developments will make large-scale single-gamete sequencing more common across sexes and organisms. The standard pre-processing pipeline of a single-cell DNA sequencing dataset generates the BAM (BinaryAlignment/Map) file that contains the mapped and cell-barcoded DNA reads from all single cells. Haplotyping these single-cell genomes with existing tools requires the steps of de-multiplexing the single-cell reads according to their cell barcodes into thousands of intermediate files before applying haplotyping methods [18]. While processing can be parallelised across cells, there are many opportunities to improve upon the use of methods developed for bulk DNA-seq data when analysing single-cell DNA-seq data by avoiding multiple reading and writing of millions of sequence reads. Single-gamete DNA sequencing data usually have low depth of coverage across the genome, especially when generated using high-throughput droplet-based single-cell sequencing protocols. However with a group of gametes sequenced, the single nucleotide polymorphism (SNP) linkage information in the gametes offers enough information for constructing chromosome-level phased haplotypes of the individual. Therefore, we here introduce *sgcocaller*, an efficient command-line toolkit to directly process the large single-cell DNA-sequencing alignment files produced by the current standard pre-processing pipelines for personalised haplotype construction and single-gamete crossover identification. *sgcocaller* is also applicable to individual gametes sequenced by bulk DNA sequencing methods with proper dataset preprocessing. We also introduce an associated Bioconductor/R package, *comapr*, for the construction, visualisation, and statistical analysis of crossover landscapes. Working in the Bioconductor ecosystem enables *comapr* to integrate seamlessly with other commonly used packages specialised in analysing biological datasets.

Results

Overview of *sgcocaller* and *comapr*

We have implemented a toolset for processing large-scale single-cell DNA sequence data from gametes with modules to construct the personalised haplotypes of the donor and to call crossovers for individual gametes. The toolset consists of two components, *sgcocaller* and *comapr* (see Availability of data and materials), for different tasks and have been implemented using designated programming languages in multiple release formats to suit the expected use-case scenarios (Fig. 1).

sgcocaller - single-gamete crossover caller

To suit the requirements of processing large alignment files containing DNA sequence read data for hundreds or thousands of individual gametes, *sgcocaller* has been designed as a command-line tool (CLT) implemented using the programming language Nim (https://nim-lang.org/). The software imports the *hts-nim* library [20] for fast processing of large DNA alignment files and variant calling files. It also imports the Rmath library (in programming language C from the R-project [21]) to use the well-defined distribution functions. To interface with the C-based library, a wrapper package (in Nim) was created and implemented as the *distributions* package that is openly accessible (see Availability of data and materials). In crossover-calling scenarios with known phased haplotypes such as F1 hybrid samples generated by crossing known reference strains of species, and cases in which

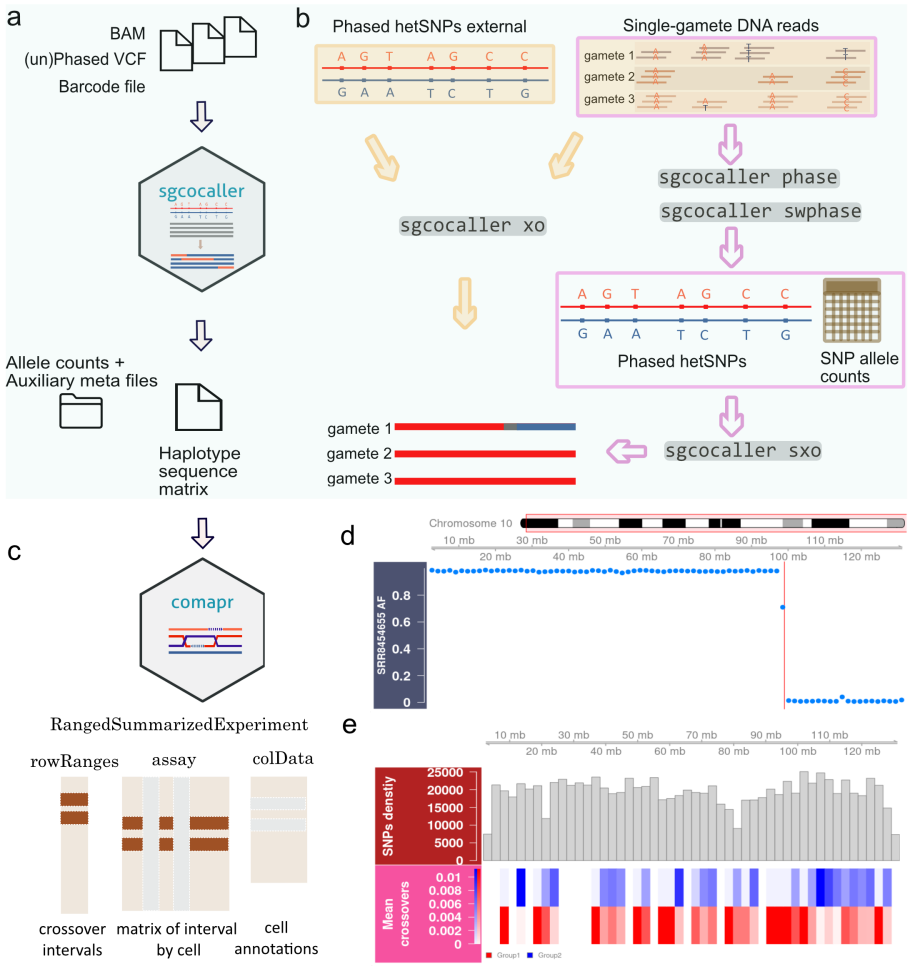

Figure 1 Overview of sgcocaller and comapr workflow a),c) File and data flows of *sgcocaller* and *comapr*. a) *sgcocaller* takes the aligned DNA reads from all gametes, the list of hetSNPs (phased or unphased) and the cell barcode list and produces the single-gamete haplotype sequence in sparse matrix. b) The diagram shows the main function of *sgcocaller* which is to resolve single gametes' haplotypes from DNA reads using phased hetSNPs obtained from external sources (left) or from applying *sgcocaller phase* (right) by the demonstrated two workflows. c) Output files from *sgcocaller* can be further processed and analysed by *comapr* that adopts the *RangedSummarizedExperiment* as the main data structure, which enables seamless integration with existing Bioconductor packages. d,e) Two example plots generated by *comapr*. d) Alternative allele frequencies plot for binned windows with crossover regions highlighted (vertical bar) from one selected gamete cell. e) The heatmap plot of crossover counts for two groups of mouse sperm cells with SNP density plot in histogram on top (both plots used the mouse sperm dataset [19]).

the haplotypes of donors are provided in an input VCF file, *sgcocaller xo*—the crossover calling module—can be applied directly. It requires three input files to call crossovers: the mapped BAM file with cell-barcoded DNA reads from all gametes, the VCF file of phased heterozygous SNP (hetSNP) markers and the list of cell barcodes (Fig. 1a). When the phased haplotype information is not available from external sources, *sgcocaller* also offers a phasing module, *sgcocaller phase*, that is based on the SNP linkage data inherent in read data from the individual gametes to produce the personalised whole-chromosome haplotypes. To call crossovers using outputs of *sgcocaller phase*, *sgcocaller sxo* is recommended instead for calling

crossovers as it uses the intermediate files generated by *sgcocaller phase* including the phased haplotype and the allele specific read count matrices as inputs, which avoids double handling of BAM/VCF files (Fig. 1). Our tool is engineered to operate directly on the output of common single-cell data processing pipelines such as CellRanger, STARSolo [22] and similar. With proper pre-processing, *sgcocaller* can also work with bulk-sequencing samples as demonstrated in the application example with the mouse single-sperm dataset (see Application of *sgcocaller* and *comapr* on public datasets).

comapr - crossover analysis for construction of genetic distance maps in R

The second software component, *comapr*, serves as a post-processing tool that includes functions for finding crossover positions, quantifying crossover rates across groups, and conducting comparative analyses after the sequences of haplotypes are inferred by *sgcocaller xo* (or *sgcocaller sxo*) for each chromosome in gametes. It is implemented as an open-source Bioconductor/R package, which offers easy integration with other packages in Bioconductor for analysing biological data and statistical testing. *comapr* includes functions to directly parse output files generated by *sgcocaller xo* with tunable parameters to systematically filter out potential false positive crossover calls and create structured data objects to represent the crossover information across all cells (Fig. 1c; see Methods). *comapr* provides quality checking functions and visualisations to understand the features of the underlying dataset and for choosing sensible filtering thresholds. It enables convenient plotting of summary plots such as number of crossovers per sample group, converting crossover rates to genetic distances in units of centiMorgans, and plotting genetic distances over chromosomes or the whole genome (see Application of *sgcocaller* and *comapr* on public datasets). *comapr* integrates with the genomic visualisation package Gviz [23] and easily generates alternative allele frequency plots with crossover intervals highlighted or with genomic feature tracks overlaid on top of identified crossover tracks (Fig. 1d). To facilitate easy statistical significance testing for comparisons of crossover rates in gamete groups, such as gametes collected from individuals with different genetic backgrounds or from different experimental groups, *comapr* implements two re-sampling based methods, bootstrapping and permutation testing (see Methods).

sgcocaller phase and *sgcocaller swphase*

Detecting crossovers in gametes' genomes requires the haplotypes of the donor's genome. When the donor's haplotypes are not available from other sources, *sgcocaller* offers a module *sgcocaller phase* that is able to produce the phased hetSNPs from unphased hetSNPs using the available single-gamete data (Fig. 1b). *sgcocaller phase* uses SNP markers' co-appearance information across all gametes to generate the chromosome-scale haplotype of the donor from single-gamete data, an idea that has also been applied in a previous study [24] (Fig. 2a). The phasing algorithm in *sgcocaller phase* first finds a template cell (a cell without crossovers) and uses the cell's genotype sequence as the template haplotype. Cells in the context of the manuscript are haploid gametes unless otherwise specified. It next fills in the missing SNPs from the template haplotype using SNP linkage information in other gametes

(see Methods). *sgcocaller phase* also generates auxiliary files for conveniently plotting diagnostic plots that help to inspect the quality of the inferred haplotypes and check switch errors (Fig. 2b). Since crossovers are low frequency events at each site across the gametes, switch errors in the inferred haplotypes can be identified from inspecting the diagnostic plot. When the plotted genotype sequences of all cells show existence of crossovers at the same position, it indicates a "crossover" or switch error has occurred in the inferred haplotype (Fig. 2b, left). Where switch errors occur we need to identify them and correct the inferred haplotypes in the template and other cells (Fig. 2b, right). We have included a switch error correction module *sgcocaller swphase* for finding and correcting the switch errors if any in the phased haplotype by *sgcocaller phase*. It produces switch scores for a selected list of SNPs with a higher risk of having switch errors and identifies the switch positions to generate the corrected haplotype (Fig. 2c; see Methods).

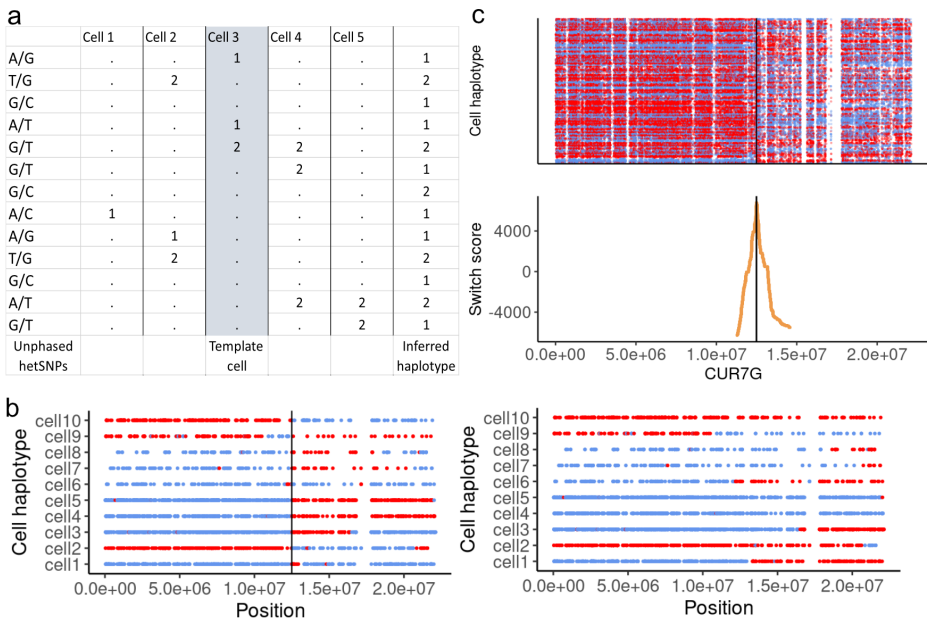

Figure 2 sgcocaller phase and sgcocaller swphase generate personalised haplotypes from single-gamete data a) Genotype matrix of an example of five gametes regarding the list of the hetSNPs. In *sgcocaller phase*, a cell is chosen as a template cell and its genotype sequence is used as the backbone for generating the inferred personal haplotype. The ideal template cell is a cell without crossovers and has sufficient SNP coverage along the chromosome. b) *sgcocaller phase* generates auxiliary files that can be used for plotting diagnostic plots, which can show whether a good template cell has been chosen and the inferred haplotype is correct. Genotype sequence of ten randomly selected cells are plotted and colored by whether the SNP's genotype matching the inferred template haplotype or its complimentary haplotype. Switching errors can be clearly identified where all cells have shown a crossover at the same position. Left plot shows the inferred haplotype has a switch error indicated by the black vertical line, while the plot on the right shows a correctly phased haplotype. Both plots were generated using the apricot gametes dataset [18]. c) *sgcocaller swphase* can identify the switch errors in inferred haplotype and correct them. Switch scores (generated using apricot dataset [18]) were calculated based on the genotypes of the flanking SNPs across all cells and switch errors were found as the identified peaks in switch scores.

We applied *sgcocaller phase* on an apricot pollen dataset generated with the 10X scCNV protocol (a droplet-based single-cell DNA sequencing protocol) [18]. The *sgcocaller phase* results demonstrate very high concordance with the haplotypes

from the published assembly from the same study (Fig. 3a,b). To evaluate performance of *sgcocaller phase*, the phasing accuracy was calculated using the fraction of concordant hetSNPs between the haplotype inferred from *sgcocaller phase* and the published haplotype. The hetSNPs were grouped into bins with 100 consecutive SNPs in each bin. The proportion of SNPs matching the published haplotype in 100-SNP bins concentrated at 1.0 across eight chromosomes indicating high phasing accuracy across the genome (Fig. 3a). When defining the alternative alleles as the alleles with lower total abundance in each bin and summarising the alternative allele reads frequencies (AAF) in 1,000-SNP bins, we found that using the haplotypes generated by *sgcocaller phase* resulted in more bins with AAF valued at zero than using the published haplotype (Fig. 3b,c). These results suggest *sgcocaller phase* generated haplotypes that are more concordant with the single-gamete read data than the published assembly.

sgcocaller xo

Two haplotypes (represented by the list of red or blue alleles on each chromosome in Fig. 1b) can be found in the diploid genomes of donors and the gametes from the donor inherit a combination of the two haplotypes (Fig. 1b). The crossover calling module *sgcocaller xo* detects the crossovers by finding the haplotype transitions in the haploid genomes of gametes. Gametes have haploid genomes, therefore theoretically there is only one type of allele that can be observed at each SNP position in each gamete. However, due to technical noise such as mapping artefacts, there can be a substantial number of SNP sites with two types of alleles found but with biased ratio towards the true allele type in the genome. To suppress the potential noise in the dataset, *sgcocaller xo* implements a two-state Hidden Markov Model (HMM; see Methods) and adopts binomial distributions for modelling the emission probabilities of the observed allele read counts (Fig. 4a).

The two hidden states in the HMM represent the haplotype origins of DNA segments (represented by allele types of a list of SNPs) in the gametes' genomes. Transitioning from one state to another between two SNP markers corresponds to a crossover detected and the transition probability is set to be dependent on the two SNP markers' base pair distances (physical distances) [19]. The transition probability is programmed as an configurable option (*cmPmb*) in *sgcocaller xo*. The relationship between observed allele counts and the underlying hidden states are modelled by the two binomial distributions whose success rates are also user-configurable options (*–thetaREF, –thetaALT*) in *sgcocaller xo* (Fig. 4a and Methods). The two states (named as REF and ALT) match with the alleles (bases) in the REF and ALT fields in the input VCF. The binomial distributions in the HMM model the ALT allele read counts at each SNP site and thus the –thetaREF value is expected to be small (e.g., 0.1) while –thetaALT value is expected to be higher (e.g., 0.9).

Lastly, the Viterbi algorithm [27] is applied to infer the most probable hidden state sequence for the list of hetSNPs for each chromosome in gametes. As currently designed, *sgcocaller* is intended to work on data of gametes collected from diploid donors. In addition to applying the HMM to the allele counts and using the Viterbi algorithm for inferring the hidden state sequences, we also included calculation of a crossover confidence score, which is the log-likelihood ratio of the Viterbi state

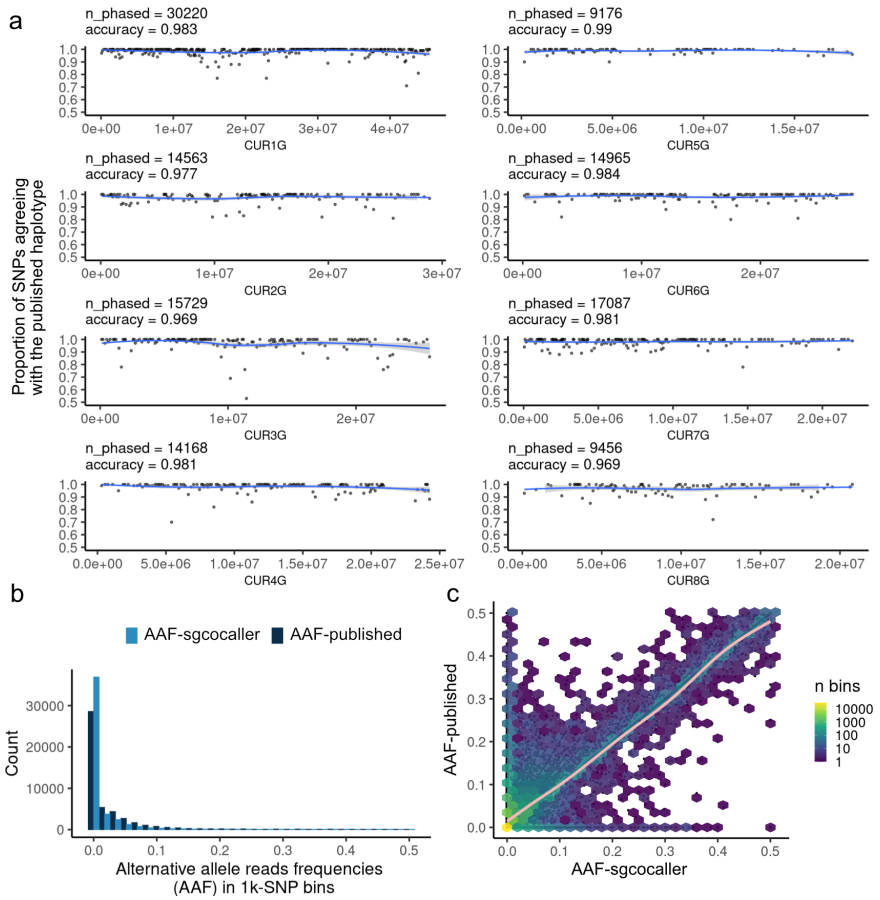

Figure 3 Comparisons between haplotype inferred by sgcocaller phase versus the published haplotype using apricot dataset [18] a) Proportion of SNPs agreeing with the published haplotype in 100-SNP bins across eight chromosomes from the using the apricot dataset [18]. The number of phased SNPs (*n_phased*) by *sgcocaller phase* and phasing accuracies (*accuracy*) were printed on top of each panel. Smoothed curves were fitted by using "loess" method from R [25, 21] . **b)** The histogram of alternative allele reads frequencies (AAF) for bins of 1,000 SNPs were plotted from all chromosomes colored by results using the haplotype generated by *sgcocaller phase* (AAF-sgcocaller) or the published haplotype (AAF-published). More bins with AAF valued at zero when using the haplotype by *sgcocaller phase* suggested that the haplotype from *sgcocaller phase* was more concordant with the single-gamete data. **c)** The two alternative read frequencies (AAF-sgcocaller and AAF-published) for each bin were plotted in scatter plot with AAF-sgcocaller on the x axis and AAF-published on the y axis. Hexagons were plotted overlaying the scatter plot with color scale indicating counts of bins covered by the hexagon area. Fitted curve that aligned with the diagonal line was plotted using the method "gam" from ggplot2 [25, 26, 21]. Alternative allele in these plots are always the alleles with fewer read counts in each bin.

segment (see Methods). The log-likelihood ratio of a segment is derived by finding the difference between the log-likelihood of the segment given the inferred state and the log-likelihood of the segment given the altered state (log-likelihood of the first path given the data minus the log-likelihood of the second path; Fig. 4b).

The outputs

Due to the sparsity of single-cell datasets, we have used sparse matrices in Matrix Market format as the output for *sgcocaller xo*, which saves disk space and enables efficient parsing in downstream analysis. *sgcocaller xo* generates sparse matrices of the allele counts, inferred haplotype states (output of the Viterbi algorithm) for

each SNP across each cell. Columns of these matrices correspond to the list of single gamete cell barcodes and rows correspond to the list of hetSNPs. A supplementary text file (*viSegInfo.txt*), which contains the summary features of inferred Viterbi segments (a list of consecutive SNP markers with the same inferred haplotype state) (Fig. 4b), is also provided and can be used for convenient post-processing such as filtering of false positive crossovers. The haplotypes of each SNP are inferred using the Viterbi algorithm [27] therefore the states/segments are also referred to as the Viterbi state/segments. Features including starting SNP position, ending SNP position, the number of SNPs supporting the segment, and the log-likelihood ratio of the Viterbi segment are recorded for each segment in the text file (*viSegInfo.txt*).

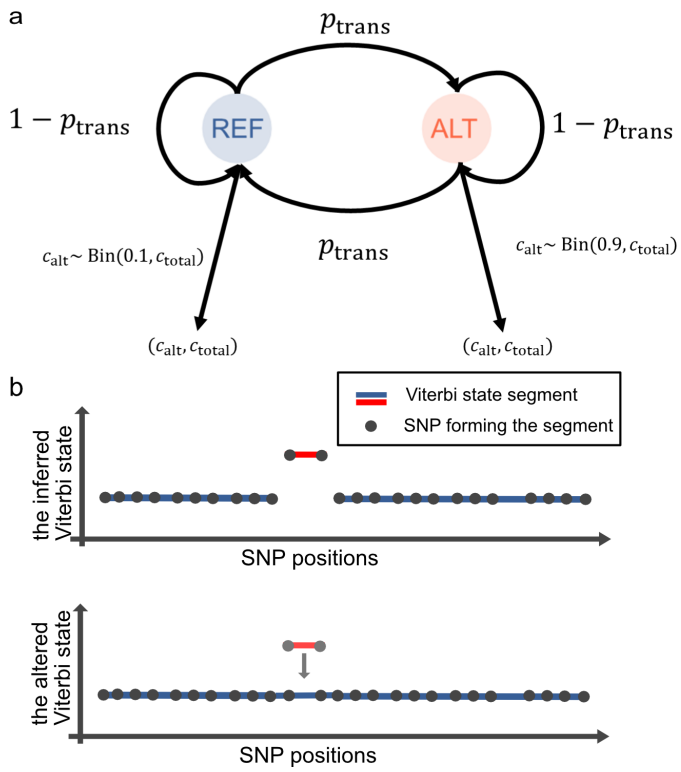

Figure 4 Hidden Markov model diagram and the Viterbi state segment a) Schematic diagram of the two-state (REF and ALT) Hidden Markov Model implemented in *sgcocaller xo*. The two hidden states represent the two haplotype states of hetSNPs with observation as allele read counts. The two possible alleles at each hetSNP site can be referred as REF or ALT arbitrarily. The two states (named as REL and ALT) match with the alleles (bases) in the REF and ALT fields in the input VCF. Binomial distributions have been adopted for modelling the relationship of observed allele counts and the underlying haplotype states of hetSNPs (emission model). p_{trans}, transition probability of transitioning to a different state (configurable option in *sgcocaller xo*). The alternative allele read count (c_{alt}) follows the binomial distribution with $p_{\text{alt}} = 0.9$ (configurable option) when hidden state is ALT and $p_{\text{ref}} = 0.1$ (configurable option) given hidden state as REF. c_{total}, the sum of reference allele read counts and alternative allele read counts. **b)** Schematic diagram of the Viterbi state segment with the inferred hidden states (output from *sgcocaller xo*) and the altered hidden states (artificially generated by reversing the inferred states of the SNPs in the underlying segment). The log ratios of the likelihoods of two conditions for each segment are reported in the output file of *sgcocaller xo*, which can be used as confidence scores of identified crossovers. Segments are colored by their states. Dots represent the SNPs forming the segments.

sgcocaller sxo

The *sgcocaller sxo* module supplements the crossover calling module *sgcocaller xo*. It runs the same core function as *sgcocaller xo* but uses the generated outputs (i.e., allele count matrices and the phased haplotype) from *sgcocaller phase* instead of the original BAM and VCF file, thus reducing unnecessary double handling of large DNA sequencing data.

Scope of *sgcocaller*

Although *sgcocaller* has been optimised to work on barcoded large-scale single-cell DNA sequencing data with outputs in formats of sparse matrices, it can also be applied on sequencing reads from one cell or bulk DNA sequencing samples for which the cell barcodes are not available with simple preparation.

For single-gamete DNA sequencing datasets generated using bulk-like protocols (individual sequencing libraries that generated separate sets of reads for gametes), *sgcocaller* can still be applied, and we have released open-access tools for preparing such data to make it compatible with the *sgcocaller—comapr* pipeline. We demonstrated examples of applying *sgcocaller* on such datasets [19, 28](see Application of *sgcocaller* and *comapr* on public datasets).

Application of *sgcocaller* and *comapr* on public datasets

Dataset of mouse sperm cells with modified format

We applied *sgcocaller* and *comapr* to haplotype sperm cells from a published DNA sequencing dataset of individual mouse sperm cells collected from an F1 hybrid mouse (C57BL/6J X CAST/EiJ) [19]. The raw sequencing dataset was downloaded from GEO (Gene Expression Omnibus) with accession GSE125326 (see Methods) and the workflow with detailed steps for pre-processing the raw reads and the execution of *sgcocaller—comapr* is publicly available (see Availability of data and materials). The crossovers called by *sgcocaller—comapr* (with a mean number of crossovers of 12 per sperm across autosomes) were highly consistent with the crossovers called from the original paper (Fig. 5). There were 7 cells (4%) that have been called with a different number of crossovers compared to the original paper (Fig. 5c). We demonstrate and provide reproducible code for running the functions to generate plots of the number of crossovers per sperm cell (Fig. 5a), genetic distances in the chosen size of chromosome bins along chromosomes (Fig. 5b), and the number of crossovers (COs) identified per chromosome (Fig. 5d) in the pubic repository (see Availability of data and materials).

To demonstrate the functions in *comapr* that test for differences in groups of cells, such as cells from different individuals, we divided the sperm cells into two groups randomly. The crossover counts and crossover rates of each group are calculated over the SNP intervals via function `countCOs` and genetic distances are derived by the `calGeneticDist` function that applies a user-selected mapping function such as the Kosambi mapping function [29]. Comparative plots can be generated for the two groups including genetic distances plots for binned intervals along the chromosome and the cumulative genetic distances along a chromosome or whole genome (Fig. 6a,b,c). Resampling-based testing functions, bootstrapping and permutation, have been called to assess the group differences statistically (Fig. 6d,e). The bootstrapping testing was conducted via `bootstrapDist` function. The bootstrap 95%

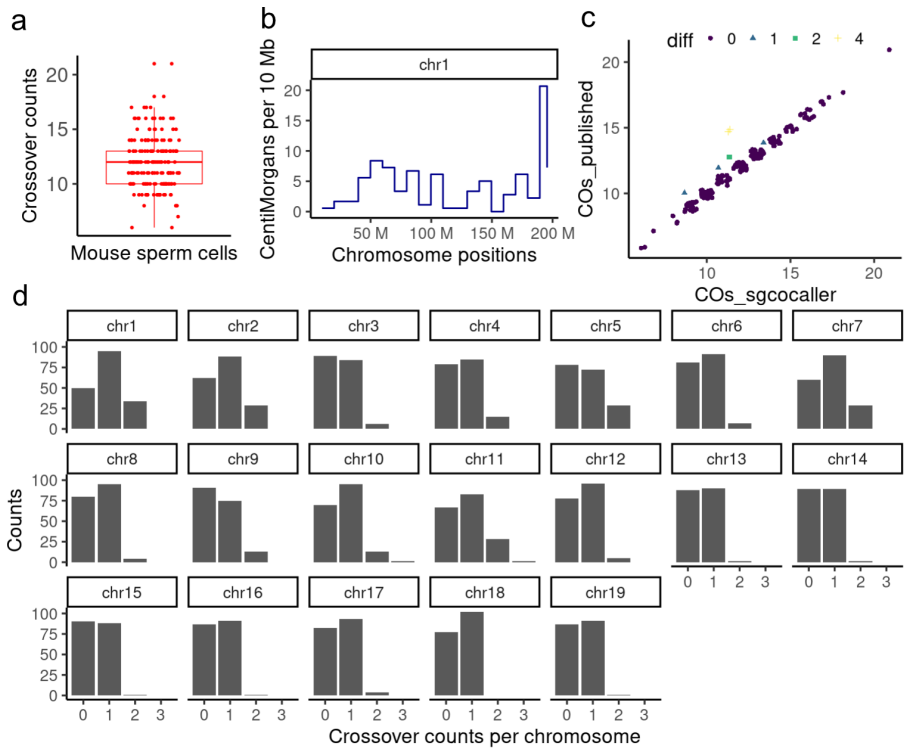

Figure 5 Application of sgcocaller and comapr on mouse sperm dataset [19] *comapr* enables easy plotting of crossover summary statistics. **a)** The distribution of crossover counts per sperm (n = 173 sperm cells). **b)** The genetic distances (in centiMorgans, which were derived from observed crossover rates by applying the mapping function Kosambi [29] with implemented function in *comapr*) plotted for chromosome regions along chromosome 1. It has shown a low to zero crossover rate around the centromere region (125M). **c)** The number of crossovers (COs) called per sperm by *sgcocaller* (x-axis) versus by the original paper (y-axis) are plotted; points are jittered and colored and shaped by the differences in number of crossovers by the two methods. Majority of the sperm cells have been called with the same number of crossovers by the two approaches. **d)** The frequency of crossovers counts for each chromosome are plotted for all sperm cells.

confidence interval of the group differences was calculated. The interval included zero, indicating that there was not enough evidence to support a difference in mean crossover numbers between the two groups at a significance level of 0.05, which was expected because the group labels of sperm cells were randomly assigned (Fig. 6d). The permutation testing via `permuteDist` returned the same conclusion with a permutation p-value > 0.05 (see Methods).

10X-scCNV sequencing of gametes from a diploid apricot tree

We next applied *sgcocaller* and *comapr* on another example dataset of single-cell sequenced haploid pollen cells from an apricot tree generated using the 10X single-cell CNV protocol [18]. The downloaded single-cell DNA reads from each apricot gamete (ENA:PRJEB37669) were re-aligned to the haplotype-resolved haploid genome (Currot) released from the same study. The hetSNP markers were identified by calling heterozygous variants from the pooled reads of all gametes *de novo* using `bcftools` and high-quality variants were kept [30, 31] (see Methods). The details of re-alignment, SNP marker identification and execution of *sgcocaller* can be accessed through our publicly available repository. The plots generated for the

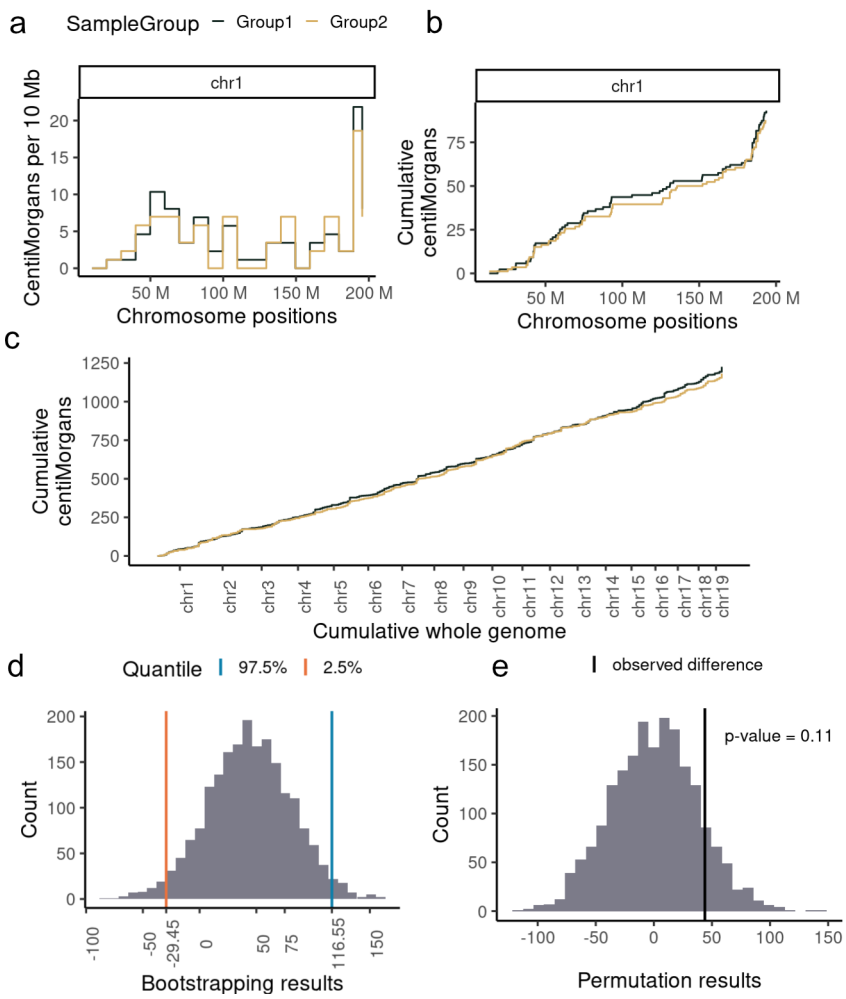

Figure 6 Demonstration of group comparisons plots generated by comapr using two groups (randomly assigned) of mouse sperm cells [19] a,b) The genetic distances are plotted for binned chromosome intervals (10 Mb) and cumulative genetic distances for two randomly assigned mouse sperm groups along chromosome 1. **c)** The cumulative genetic distances plot across all autosomes for the two sperm groups. **d)** The bootstrapping distribution of the difference in total genetic distance between the two groups with two vertical lines indicating the lower bound and the upper bound of 95% confidence interval. **e)** Permutation results of difference in total genetic distance between two sperm cell groups with observed group difference indicated by the vertical line and p-value labelled on the top.

manuscript can be accessed as reproducible markdown files as well as a an HTML report (see Availability of data and materials).

Comparing with the crossover profile constructed from the original study, *sgcocaller* and *comapr* have generated a highly concordant crossover profile for the analysed nuclei (Fig. 7 and Fig. S2). The number of crossovers identified per gamete (n = 329 gametes) were consistent with the published number of crossovers for these gametes by using both the published haplotype (sg.xo KnownHaps) and the *sgcocaller phase* generated haplotype (sg.sxo SgHaps) (Fig. 7a). The majority of the 2,632 chromosomes studied here were called with the same number of crossovers by both methods. Few chromosomes (4% by sg.xo KnownHaps; 5.6% by

sg.sxo SgHaps) exhibited differences in crossover counts (Fig. 7b). Out of the chromosomes that were called with different numbers of crossovers, 69% of them were called with fewer crossovers by sg.xo KnownHaps and 86% with fewer crossovers by sg.sxo SgHaps. Therefore, the discrepancy of results from the two methods resulted predominantly from *sgcocaller* being more conservative and calling fewer crossovers. Inspecting the crossover profile by chromosome regions has also demonstrated consistent genetic distances across the whole genome (Fig. 7c).

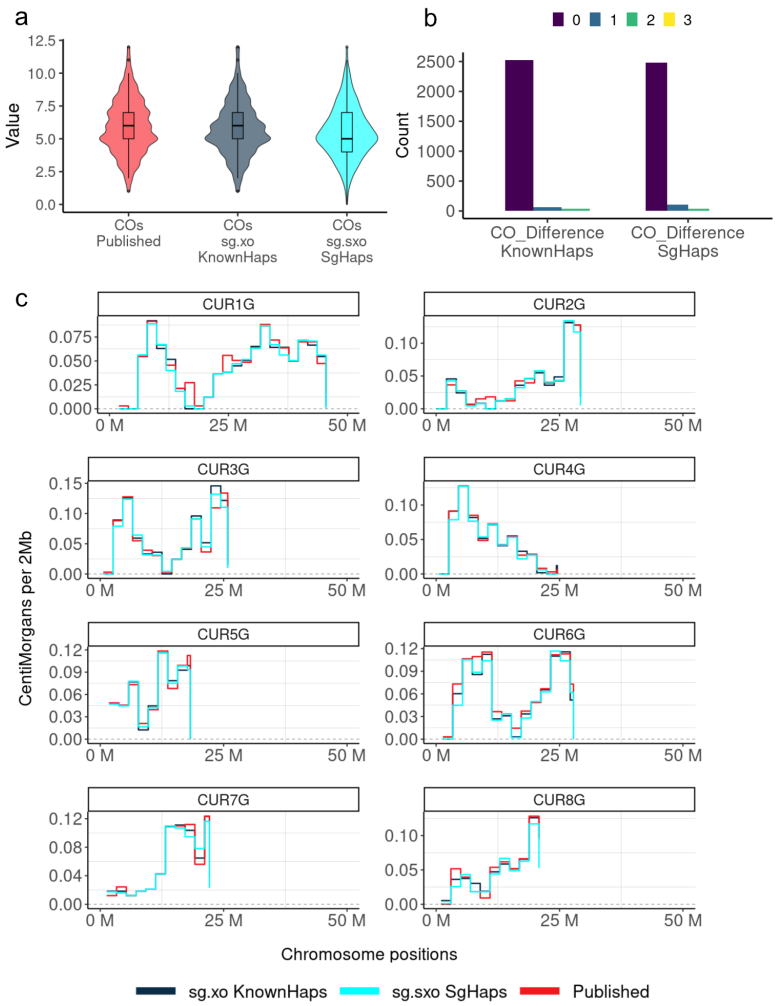

Figure 7 Comparisons between number of crossovers called from sgcocaller versus from the published study a) The distribution of the number of crossovers called per apricot gamete (n = 329 apricot gametes) from the published study [18] (COs Published), calling *sgcocaller xo* using the published haplotype (sg.xo KnownHaps) and calling *sgcocaller sxo* with *sgcocaller phase* generated haplotype (sg.sxo SgHaps). **b)** The frequency of the difference in crossover counts with published crossover results per chromosome by *sgcocaller* using either the published haplotype (CO_Difference KnownHaps) or using *sgcocaller phase* generated haplotype (CO_Difference SgHaps). **c)** The genetic distances in every 2 Mb chromosome bins are calculated using crossover rates estimated from three approaches that show high concordance along regions of the genome.

It is worth noting that not all cells available in the published study were analysed and included in the application results of *sgcocaller* due to the cell filtering step included in *comapr* which filtered out cells with excessive numbers of crossovers

called and cells with poor SNP coverage. Chromosomes with excessive numbers of crossovers called are likely due to library preparation artefacts or abnormal chromosome segregation that results in implausibly many heterozygous SNPs in single gametes (Fig. S3). *comapr* offers a systematic way of filtering out these cases and can be applied to all gametes.

Comparing to other methods

Many studies apply hidden Markov model based approaches for haplotype construction and crossover identification using sequencing or genotyping datasets [7, 19, 32, 33, 34, 35]. The majority of them, however, used customized or in-house scripts for crossover calling [19, 32, 7]. Only a handful of them published reusable software tools or pipelines [34, 36], but none of them apply in the same scenarios as *sgcocaller* and *comapr*, which work directly on large-scale single-cell DNA sequencing datasets. The Hapi method, implemented in an R package, was proposed to construct chromosome-scale haplotypes using data from a small number of single gametes [36]. Given a genotype matrix as input, it has functionalities overlapping with *sgcocaller phase* [36]. We therefore compared the phasing performance of Hapi and *sgcocaller phase—swphase* on data of 11 human sperm cells (the same dataset used in the performance evaluation of Hapi in the original paper [36]) and the 10X scCNV generated apricot gamete dataset [18] (see Methods). To measure uncertainty in method performance, we constructed dataset replicates by dividing the original sets of gametes into ten portions and each time leaving one portion (10%) of the gametes out. Since there were only 11 cells in total in the human sperm cell dataset, 11 dataset replicates were constructed (n = 10 cells each). Ten distinct datasets were constructed from the apricot gametes (n = 330 gametes each). We have run Hapi and *sgcocaller phase—swphase* on the constructed datasets on both the human sperm cell and apricot gametes. The phasing accuracy, measured by calculating the fraction of SNPs agreeing with the ground-truth haplotype sequence, and the number of phased SNPs by both approaches were compared (Fig. 8a,b).

sgcocaller phase—swphase achieved the same completeness as Hapi and achieved comparable accuracies for all sperm combinations (Fig. 8a). Hapi was applied with the same settings for all chromosomes and datasets in which 4 chromosomes triggered errors (out of 242 chromosomes tested across all datasets) and were excluded from the comparison. The performance results on the apricot dataset indicated *sgcocaller phase—swphase* generated haplotypes that were generally concordant with the published haploid genome assembly, whereas Hapi did not perform well with fewer hetSNPs phased and low phasing accuracies across eight chromosomes (Fig. 8b).

We further compared the differences between the two datasets, which potentially explained the discrepant performance of Hapi. The comparison showed that the human sperm sequencing dataset has much higher SNP coverage per cell and cell coverage per SNP compared to the apricot dataset (Fig. 8c and Fig. S1). While Hapi and *sgcocaller phase* both perform well on relatively high-coverage data, *sgcocaller phase* comprehensively outperforms Hapi on low-coverage data common in high-throughput single-cell DNA sequencing studies.

We also applied *sgcocaller phase* and Hapi on the mouse sperm dataset (n = 194 sperm cells) and measured running times for both methods. The individual

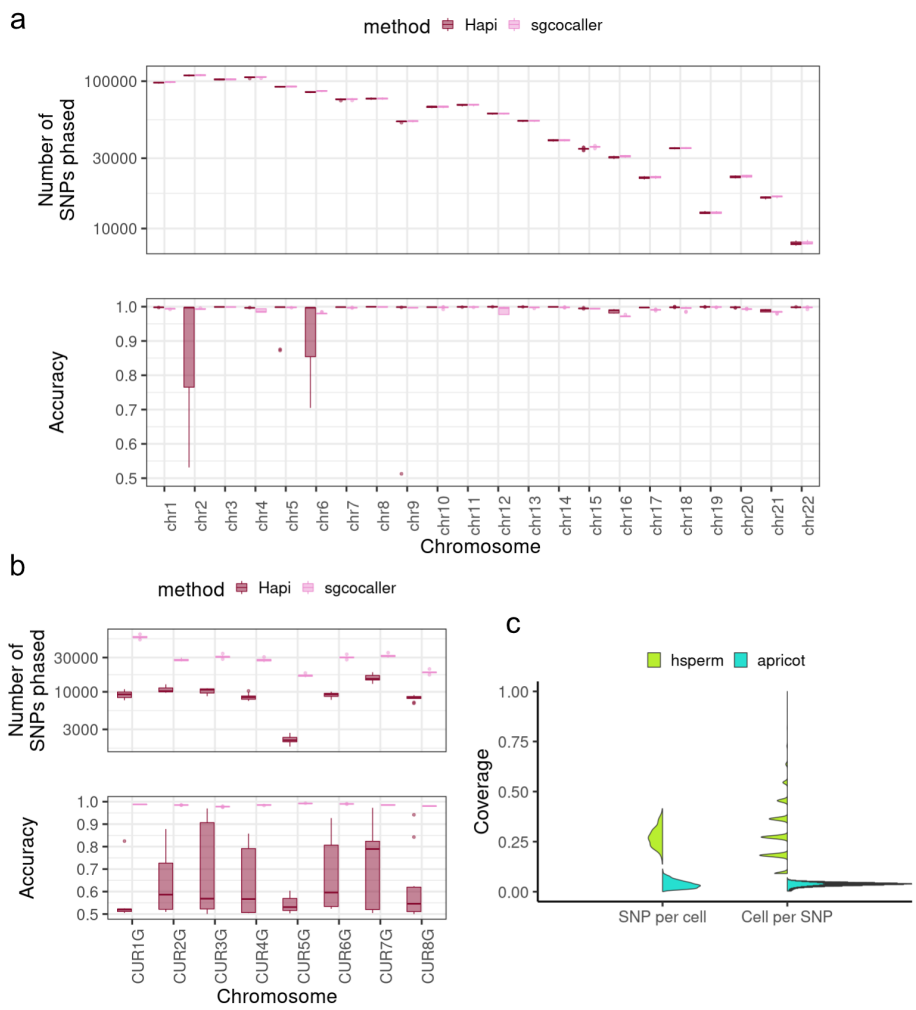

Figure 8 Phasing module performance comparison with existing method The phasing performance of *sgcocaller* and Hapi on the human sperm sequencing dataset (n=11 sperm cells) and apricot single-cell DNA sequencing dataset (n=367 apricot gametes) were compared. **a)** The number of phased SNPs and phasing accuracy from running two methods (*sgcocaller* and Hapi) on 11 constructed datasets using 11 human sperm cells [28] are plotted in boxplots. The number of points per boxplot is 11 when all datasets were successfully phased. Hapi failed to return values for 4 chromosomes (out of $11 \times 22 = 242$ in total), which were not included in the plots. **b)** Same with a) but with results from running on apricot datasets. Number of points per boxplot is 10. **c)** The characteristics of two datasets are compared, which indicates the human sperm (hsperm) dataset is of higher coverage. Distributions of the SNPs' coverages per cell and cells' coverages per SNP show that both metrics are much higher in the human sperm sequencing dataset.

mouse sperm cells were sequenced using bulk sequencing techniques, thus the mouse sperm dataset was of high read coverage comparable to the human sperm dataset but with more cells. While the running time of Hapi and *sgcocaller phase* on the apricot and human sperm dataset was similar (data not shown), we observed that *sgcocaller phase* was able to generate the chromosome-scale haplotypes significantly faster than Hapi. Considering *sgcocaller phase* processes an extra step of parsing the DNA read file (BAM file) and variant file (VCF file), it makes *sgcocaller phase* more advantageous in terms of speed. Examining diagnostic plots generated from *sgcocaller phase* can help with identifying switch errors so that *sgcocaller swphase*

can be run selectively. Nevertheless, *swphase* was applied to all chromosomes and the total running times were still many-fold shorter than Hapi (Fig. S4).

Scalability of *sgcocaller*

We tested the scalability of different modules in *sgcocaller* with input files of varying sizes. Scalability testing was conducted by applying *sgcocaller* on chromosome 1 from the mouse sperm dataset. We measured the running time and memory usage of different modules by varying either the number of SNPs or the number of DNA reads available. When varying the number of DNA reads to process, the same input VCF which had 4.6×10^5 SNPs was used, while the same BAM file with 1.1×10^8 reads was used when varying the number of SNPs. Running time and memory usage for all three modules scale linearly with the number of SNPs (nSNPs) to process. *sgcocaller sxo* runs significantly faster than *sgcocaller xo* as expected since it parses the allele counts matrices from *sgcocaller phase* instead of the input BAM and VCF file. Running time and memory usage of the three modules do not strictly increase with more DNA reads (nReads) to process, and plateau after nReads reaches about 320 million (Fig.9). This performance is potentially due to the coverage rate of the available SNPs increasing as the number of DNA reads increases. However, when DNA reads are sufficient to cover all the available SNPs, increasing number of reads does not immediately increase running time or memory usage. Measurements were generated from running *sgcocaller* on a Linux server with `RedHat7.9(Maipo)` and with `--thread 4` for decompressing BAM input. The measurements of time and memory usage for different inputs were obtained using the "benchmark" function from the workflow manager `Snakemake` [37].

Discussion

We have introduced a toolkit that consists of a command-line tool and a Bioconductor/R package for processing large-scale single-gamete DNA read datasets for individualised haplotype construction, crossover identification, and crossover landscape analysis. Personal haplotype construction is important in population genetics and clinical genetics for interpreting rare and disease-implicated variants [38, 39]. Genomes of haploid gametes provide information of "long-range" haplotype blocks of the diploid donors and can be revealed via standard short-read sequencing. Single-gamete data generated using low-depth short-read sequencing are sufficient for reconstructing the two personal haplotypes by aggregating linkage information in a group of gametes. Taking advantage of the single-gamete data, *sgcocaller phase* is able to construct personal haplotypes with standard short-read sequencing methods. Comparisons with other phasing methods have shown that *sgcocaller phase* offers better performance on single-cell sequenced gametes in terms of accuracy and efficiency.

The application demonstration of *sgcocaller—comapr* on public datasets indicates that *sgcocaller—comapr* are able to produce the same results as other methods but with greater convenience and computational efficiency for large datasets. Although *sgcocaller—comapr* are optimised to work with cell-barcoded DNA reads, we also demonstrated and showed examples of how our tools can be applied on bulk-sequenced individual gametes. Using the modern programming language Nim

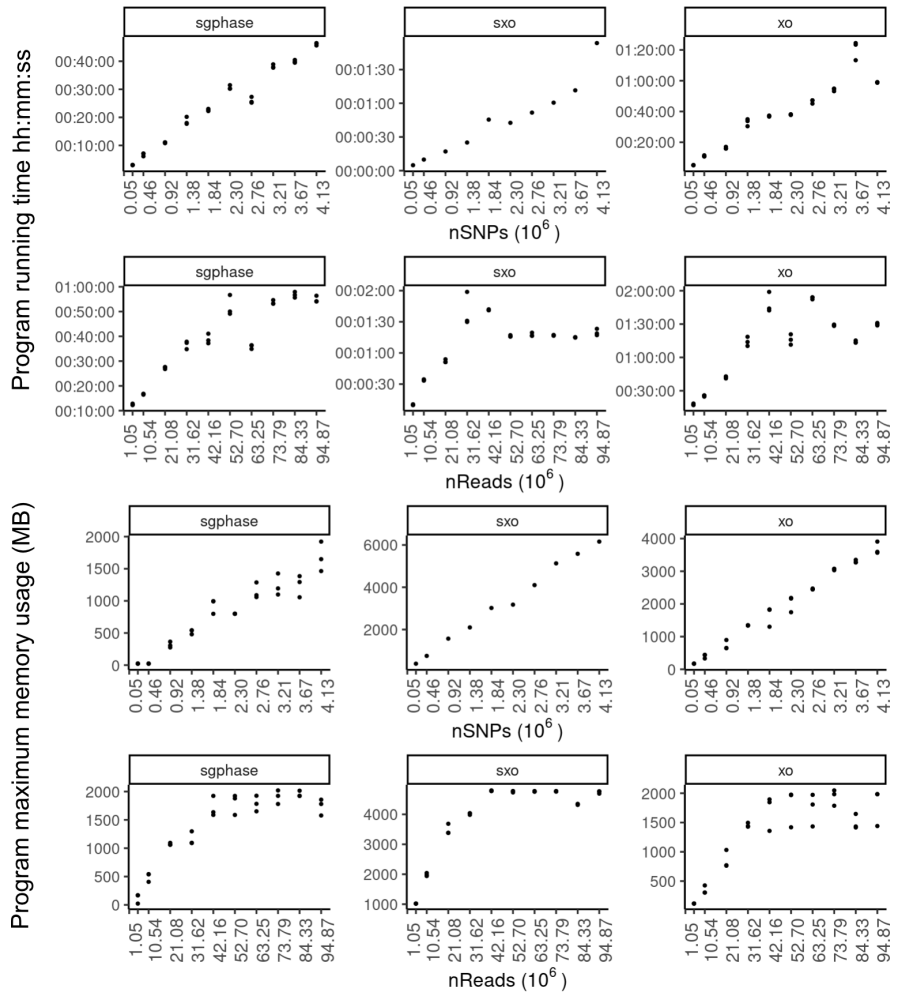

Figure 9 Module running time and memory usage by sgcocaller with varying input sizes. Top two panels show the running time in format of hour:minute:seconds of different modules with varying number of SNPs (nSNPs) and number of DNA reads (nReads) to process. X-axis labels are in the scale of 10^6. Bottom two panels show the memory usage with the reported 'max_rss' in mega bytes (MB), which is maximum amount of memory occupied by a process at any time that is held in main memory (RAM), for different modules. Each measurement has been repeated three times by using the 'benchmark' function from `Snakemake` [37].

and building on top of C-based libraries, *sgcocaller* processes the large genomic datasets efficiently. The multiple release formats of *sgcocaller* enable it to be more accessible to the community.

Abnormal meiotic crossovers are often associated with infertility [40]. Therefore, characterising individual meiotic crossover profiles and studying crossovers as a phenotype using single-cell techniques has application in reproduction clinics. Mechanisms and factors affecting meiotic crossovers are active research topics. Using single-gamete methods for generating and comparing crossover profiles of individuals with different conditions is more efficient and does not require recruiting a large number of samples.

With the advancement of single-cell DNA sequencing technologies that generate larger-scale and higher-quality datasets, our software serves as a complete toolkit as

the first step to inspect the single gamete data of an individual for understanding individual-level variations in meiotic crossovers [9].

Due to the current design of the methods, *sgcocaller*—*comapr* are limited to the analysis of gametes from diploid organisms. Currently, *sgcocaller* only supports using multiple threads for decompressing the input BAM file and recommends users to run each chromosome in parallel to reduce computational time. Future work in the development of *sgcocaller* might be to update the program to include the multi-threading feature by chromosomes internally and to support analyses for polyploid organisms.

Conclusion

Available in multiple release formats (nimble package, static binary, docker image), *sgcocaller* fills the gap of an efficient tool with simple installation for constructing personalised haplotypes and calling crossovers in single gametes from single-cell and bulk DNA sequenced gametes. The availability of the companion Bioconductor/R package *comapr* enables the downstream analysis of crossovers in the cells and among individuals and integrates well with existing R packages for various visualisation and exploratory analysis tasks.

Methods

Phasing from single gametes

Haploid genomes of the pool of gametes collected from an individual are the results of meiosis and meiotic crossovers. Based on the known mechanisms of meiosis and crossovers, with respect to each chromosome half of the gametes produced in meiosis will not have crossovers and thus they represent the original (maternal/paternal) haplotypes of the diploid donor. Thus, for a given chromosome, we expect the rest of gametes to have crossovers (i.e., the chromosomes are formed by the combined parental haplotypes of the donor). The crossovers are low frequency events across chromosomes and crossover positions are sparse. The SNP linkages in small chromosome regions across all haploid gametes are therefore reliable for reconstructing the donor's haplotypes.

Single gamete genotype matrix

To phase the two haplotypes of each chromosome for the diploid donor from gametes, the first step is finding the list of unphased hetSNPs (heterozygous SNP loci that differ between the maternal and paternal homologous chromosomes) by standard variant calling tools such as bcftools using pooled DNA reads from gametes. Only biallelic SNPs are considered in this step, therefore finding one haplotype also implicitly resolves the second haplotype by switching all alleles to its alternative at each hetSNP. In other words, the two haplotypes are bitwise complementary to each other. With the hetSNPs known, the DNA reads from each gamete are parsed and summarised into a genotype matrix for each chromosome with values of 1 or 2 corresponding to matching with the REF allele and the ALT allele at each unphased hetSNP. ALT allele read frequency (AF) is used for genotyping each SNP in each gamete and AF ≤ 0.3 is genotyped as 1 (REF) while SNPs with AF ≥ 0.7 are genotyped as 2 (ALT). Filtering options are available for excluding low mapping/base quality reads, low coverage SNPs per cell with the options *–minCellDP* and *–minTotalDP*.

Template cell identification

sgcocaller phase first attempts to identify a gamete cell without crossovers as a template cell whose sequence of genotypes are used as the initial inferred haplotype. An ideal template cell for phasing each chromosome is a cell without crossovers and having sufficient SNP coverage regarding this chromosome. To find a cell without crossovers, *sgcocaller phase* finds pairs of gametes with lowest genotype dissimilarity. It is based on the idea that when two gamete chromosomes have no crossovers, their genotype sequences will be the same (at least very similar). *sgcocaller phase* includes $maxDissim$ as a user supplied option that controls how similar ($1-maxDissim$) it requires for two cells to be considered as template cells. *sgcocaller phase* finds a maximum of three cell pairs as potential template cells. When multiple template cell pairs are available, the cell with the largest SNP coverage is chosen as the template cell.

There are rare cases when two gametes have crossovers at exactly the same positions, which also leads to high genotype similarity in the gametes. Such cases can be revealed by diagnostic plots (Fig. 2b) with the simple R script provided and then corrected by *sgcocaller swphase*. The template cell to use can also be chosen by the user via the $templateCell$ option.

Infer missing SNPs in the haplotype template

Upon forming a haplotype template, *sgcocaller phase* increases the completeness of the haplotype template by inferring the genotype of the missing SNPs (that is, SNPs with no read coverage) from the template using other gametes in which the SNP is available (that is, have read coverage). A haplotype template represents two actual haplotype (allele) sequences (h, h') that are bitwise complementary with respect to the REF and ALT alleles defined for the hetSNPs used; one haplotype is thus derived by changing all alleles from the complementary haplotype to their alternatives. Intuitively, we want to use linkage information from other gametes to "fill in the gaps" in the template haplotype by finding gametes with coverage at the missing SNP site. Loosely, if a gamete with coverage at the missing SNP has the same haplotype as the template, then we assume that the template should have the same allele at the missing SNP as the gamete with coverage. If the gamete with coverage has the complementary haplotype, then the missing SNP in the template should have the alternative allele.

More precisely, to infer missing SNPs' genotypes in the haplotype template, we need to find the linkage type of the SNP to the haplotype template. Only two types of linkages are possible due to the existence of two possible alleles (e.g., 1 or 2) for a given SNP. The first linkage type ("type 1") refers to gametes with the template haplotype and allele 1 at the missing SNP and gametes with the haplotype complementary to the template haplotype and allele 2 at the missing SNP; that is, the missing SNP should have allele 1 in the template (type 1:$\{(1, h), (2, h')\}$). The second linkage type ("type 2") is the inverse, such that the template should have allele 2 at the missing SNP (type 2:$\{(1, h'), (2, h)\}$). To determine the linkage type of the SNP in supporting gametes (that is, those with read coverage of the SNP), the nearby SNPs' genotype sequence in each supporting gamete (specifically, the 10 closest SNPs with read coverage in both the template gamete and supporting

gamete) is first compared with the template haplotype at the matching positions to define the haplotype (template or complementary) of the supporting gamete. With the haplotype of the supporting gamete determined, the linkage type supported by the gamete immediately follows. The posterior probabilities of linkage types of a missing SNP are calculated by looking at the number of gametes supporting each type of SNP linkage. Assuming a genotype error rate of 0.1 and that the two linkage types are equally likely to happen, the posterior probability of each linkage type of a missing SNP can be calculated:

$$t_1\text{: \{type 1 linkage counts\}} \qquad t_2\text{: \{type 2 linkage counts\}}$$

$$\begin{aligned}
p(\text{type } 1|t_1,t_2) &= \frac{p(t_1,t_2|\text{type } 1)p(\text{type } 1)}{p(t_1,t_2|\text{type } 1)p(\text{type } 1)+p(t_1,t_2|\text{type } 2)p(\text{type } 2)} \\
&= \frac{p(t_1,t_2|\text{type } 1)}{p(t_1,t_2|\text{type } 1)+p(t_1,t_2|\text{type } 2)} \\
&= \frac{0.9^{t_1}0.1^{t_2}}{0.9^{t_1}0.1^{t_2}+0.1^{t_1}0.9^{t_2}} \\
p(\text{type } 2|t_1,t_2) &= \frac{0.9^{t_2}0.1^{t_1}}{0.9^{t_1}0.1^{t_2}+0.1^{t_1}0.9^{t_2}}.
\end{aligned}$$

We use a threshold cut-off (default as 0.99 and it can be changed via option $posteriorProbMin$) for determining whether a missing SNP can be inferred. Missing SNPs with posterior probabilities over the threshold are inferred by the suggested linkage type. Applying this approach genome-wide, we can make maximal use of read coverage across all gametes to maximise the completeness of the template haplotype. After inferring missing SNPs from the template haplotype, the step of inferring SNPs is then performed against all hetSNPs in the template haplotype to correct any genotyping errors in the template haplotype.

Switch error correction

When an ideal template cell is not used for phasing in the previous step, the chosen template cell may have crossovers leading to switching errors in the inferred haplotype (Fig 2b). We implement an additional module, *sgcocaller swphase*, that is able to detect the switch errors and generate the corrected haplotype. To save unnecessary computing, *sgcocaller swphase* calculates the switch scores only for identified SNP bins whose positions have high risk of having switch errors. High risk SNP bins are found by firstly grouping all hetSNPs into bins of 2,000 consecutive SNPs with a moving step of 200 SNPs (both are changeable via options when running *sgcocaller swphase*). The proportion of gametes having crossovers are calculated for each bin. A SNP bin is labelled as a high risk bin when the proportion of gametes having crossovers is above 0.5. It is based on the idea that when the majority of gametes have crossovers, it indicates a "crossover" in the template haplotype. The crossover identification in this step is fast as it simply compares the dissimilarity of gametes' genotype sequences with the inferred haplotype sequence for the SNPs in each bin. A default threshold value of 0.0099 is set for the dissimilarity to decide whether a crossover has happened in the gamete or not.

To construct the switch score (formed using a concept for splitting blocks similar to that from a previous haplotype construction method [41]), which represents how likely a switch error has happened at SNP i in the inferred haplotype, the switched haplotype is first constructed. Let H_l^i represents the haplotype sequence to the left N bases of SNP i, and H_r^i represents the haplotype sequence to right N bases of SNP i. The current haplotype around SNP i is $H^i = \{H_l^i, H_r^i\}$. The switched haplotype is a new sequence of $H_{sw}^i = \{H_l^i, H_r^{i\prime}\}$, where $H_r^{i\prime}$ is the bitwise complementary sequence of H_r^i. The switch score for each SNP i is calculated using the log-ratio of the probability of observing the genotype sequences in all gametes given the switched haplotype with the probability of observing the genotype sequences in all gametes given the non-switched haplotype. Assuming the occurrence of a switch error or not at any site is random, that is the prior of having switch or no switch at a SNP site equals 0.5, the switch score is the log-ratio of the posterior probabilities of the switched haplotype and the not-switched haplotype. When calculating the probability of observing each gamete's genotype sequence given the haplotype (H^i or H_{sw}^i), only the local (N) SNPs are considered (controlled by the *lookBeyondSNPs* option with default set to 20).

The haplotype H^i is a sequence of alleles and also implicitly defines the second haplotype $H^{i\prime}$ which can be derived by switching all alleles to their complementary alleles (in the called genotypes of the hetSNPs). The probability of observing all gametes' genotype sequences at the local (N) SNPs around SNP i given the local haplotype $(H^i, H^{i\prime})$ is,

$$p(G^i \mid H^i, H^{i\prime}) = \prod_{j=1}^{m} p(G_j^i \mid H^i, H^{i\prime}),$$

where G^i represents local genotypes around SNP i of all (m) gametes, and G_j^i represents local genotypes at SNP i from gamete j. For each gamete j, the probability of its genotype sequence given $(H^i, H^{i\prime})$ is

$$p(G_j^i \mid H^i, H^{i\prime}) = \frac{p(G_j^i \mid H^i) + p(G_j^i \mid H^{i\prime})}{2}.$$

Similarly for the switched haplotype H_{sw}^i, the probability of observing the local genotypes of all gametes given $(H_{sw}^i, H_{sw}^{i\prime})$,

$$p(G^i \mid H_{sw}^i, H_{sw}^{i\prime}) = \prod_{j=1}^{m} p(G_j^i \mid H_{sw}^i, H_{sw}^{i\prime}),$$

where

$$p(G_j^i \mid H_{sw}^i, H_{sw}^{i\prime}) = \frac{p(G_j^i \mid H_{sw}^i) + p(G_j^i \mid H_{sw}^{i\prime})}{2}.$$

In addition, probability of observing a gamete's genotype sequence given a haplotype sequence, $p(G_j \mid h)$, is calculated as (assuming genotype error rate 0.1 and

using d as the number of different bases between the allele sequence in G_j and the allele sequence in haplotype allele sequence h):

$$p(G_j \mid h) = 0.1^d \times 0.9^{(K-d)},$$

where K is the number of co-existing SNPs in two allele sequences under comparisons. The switch score for SNP i is thus

$$S_i = \log \frac{p(G^i \mid H^i_{sw}, H^{i\prime}_{sw})}{p(G^i \mid H^i, H^{i\prime})}.$$

Upon calculating switch scores for a sequence of SNP positions, the switching point is identified as the peak of a stretch of positive switch scores (Fig. 2c). The minimum threshold for identifying switch points is set via the option *minSwitchScore*, which can vary depending on features of the dataset including the number of cells available. The template haplotype is then corrected by flipping all SNP alleles to their complementary alleles after an identified switch point, thus generating the corrected haplotype with the switch error removed.

Crossover detection

Crossovers can be detected by finding haplotype shifts in the gametes' haploid genomes. With DNA reads from each cell mapped, the haplotypes of SNP markers are inferred by looking at the alleles carried by the DNA reads mapped to genomic positions of these SNP markers. However, with technical artefacts (from sequencing and mapping), it is expected to observe some proportion of conflicting alleles from the underlying haplotype (Fig. 1). To reconstruct the haplotype structure of the genome from the mapped DNA reads while accounting for technical noise including mapping errors for crossover identification, *sgcocaller xo* applies a Hidden Markov model (HMM) with a binomial emission model (Fig. 4).

The Hidden Markov Model The data in this model are the allele-specific read counts across the list of hetSNP sites for each chromosome, whereas the underlying haplotype of a SNP site on the chromosome is a hidden variable to be inferred. In the case of gametes, which have haploid genomes, there are two possible hidden states for each SNP corresponding to the two haplotypes of the parent. The two possible alleles at each hetSNP site can be referred as REF or ALT arbitrarily. The REF or ALT state for each SNP also aligns with the same REF or ALT allele in the provided VCF input file. At each SNP site i, the two hidden states: $s^i = \text{alt}$ corresponds to ALT haplotype while $s^i = \text{ref}$ corresponds to REF haplotype. The emission probabilities are modelled by two binomial distributions:

$$c^i_{\text{total}} = c^i_{\text{alt}} + c^i_{\text{ref}} \ ,$$

$$c^i_{\text{alt}} | s^i = \text{alt} \sim \text{Bin}(c^i_{\text{total}}, p_{\text{alt}}) \ ,$$

$$c^i_{\text{alt}} | s^i = \text{ref} \sim \text{Bin}(c^i_{\text{total}}, p_{\text{ref}}) \ ,$$

where c^i_{alt} and c^i_{ref} denote the alternative allele read count and reference allele read count at SNP i, respectively, c^i_{total} denotes the total read count at SNP i, and p_{alt} and p_{ref} are configurable parameters when running *sgcocaller xo* and denotes the success rates in the two binomial distributions respectively. The transition probabilities (p^i_{trans}) are modelled dependent on markers' base pair distances [19] with the default of average 0.1 centiMorgan per 1Mb (1 million base pairs) which can be changed via the *cmPmb* option. Lastly, the initial probabilities for the two hidden states are both set to be 0.5, making the assumption that they are equally likely to happen.

With this HMM, the commonly used dynamic programming method, the Viterbi algorithm [27], is implemented in *sgcocaller xo* to solve for the most probable hidden state sequence for each chromosome in each cell.

Measuring support for detected crossovers

We use a quantitative measure, the log-likelihood ratio (logllRatio), for measuring the amount of support from data for detected crossovers. We define a (Viterbi) state segment as a consecutive list of SNPs with the same state. We also use the term "inferred state" to refer to the state inferred through applying the Viterbi algorithm, whereas we use the term "altered state" of a SNP to refer to the state obtained by altering its inferred state to its opposite. We measure the support in the data for the detected crossover using the log-likelihood ratio for the segment with the inferred state, relative to the altered state.

Specifically, the logllRatio is calculated by taking the log-likelihood of the data given the current inferred state minus the log-likelihood of the data given the altered state (Fig. 4b). For a state segment that spans SNP m to k, assuming the state segment has been inferred with state REF,

$$\text{Logll}_{\text{inferred}} = \log(t_l) + \sum_{i=m}^{k} \log f(c^i_{\text{alt}}; c^i_{\text{total}}, p_{\text{ref}}) + \log(t_r),$$

where t_l and t_r denote the transition probabilities from SNP $m-1$ to m and from SNP k to $k+1$, respectively, and f is the binomial probability mass function given by

$$f(x; c, p) = \binom{c}{x} x^p (c-x)^{1-p}.$$

The log-likelihood of the data given the altered state is thus

$$\text{Logll}_{\text{altered}} = \log(1-t_l) + \sum_{i=m}^{k} \log f(c^i_{\text{alt}}; c^i_{\text{total}}, p_{\text{alt}}) + \log(1-t_r).$$

The logllRatio is derived by

$$\text{Logll}_{\text{inferred}} - \text{Logll}_{\text{altered}}.$$

comapr - implementation

Data structure The well-structured 'RangedSummarizedExperiment' class defined in the Bioconductor R package 'SummarizedExperiment' is used as the main data structure in *comapr* for storing the SNP interval regions and the number of called crossovers per cell (Fig. 1) [42]. The data slot 'rowRanges' is used for storing the SNP intervals, whereas the cell-level information is stored in the 'colData' slot. Using 'RangedSummarizedExperiment' also enables *comapr* to conveniently integrate with the various genomic coordinate plotting functions in the Gviz package [43].

Resampling-based functions for testing differences in crossover profiles To test for differences in the number of crossovers between any two groups of cells, re-sampling methods, permutation and bootstrapping tests [44, 45, 46], have been implemented in *comapr*. The `permuteDist` function performs permutation:

1. Record the observed difference in total genetic distances between the two groups of cells, d_{obs}.
2. Take the group labels vector and permute the group labels by randomly assigning the labels across cells and calculate the new difference with the newly-generated permuted grouping.
3. Repeat step 2 for B times (e.g., $B = 1,000$).
4. Calculate the permutation p-value by using the `permp` function from the `statmod` package [47] that calculates the appropriate p-values for the permutation test when permutations are sampled with replacement, avoiding the common pitfalls of under-estimated p-values [44, 47].

The steps for generating the bootstrapping confidence intervals of genetic distance differences in two groups (A and B) of cells are implemented in the `bootstrapDist` function:

1. Randomly draw n cells with replacement from cells in group A where n is the group size of A and calculate the total genetic distance d_1 with the cells drawn.
2. Randomly draw m cells with replacement from cells in group B where m is the group size of B and calculate the total genetic distance with the cells drawn d_2.
3. Calculate difference in total genetic distances by $d_1 - d_2$.
4. Repeat step 1-3 for B times (e.g., $B = 1,000$).
5. Calculate bootstrap confidence intervals for the sampling distribution of the difference in genetic distance using the `quantile` functions at a desired level from R [21].

Whether or not the acquired confidence intervals contain zero can be use to decide whether or not the difference is statistically significant at a desired level.

Preprocessing public datasets

Mouse sperm dataset

Raw fastq files of 217 mouse sperm cells were available from accession GSE125326 (Gene Expression Omnibus) [19], fastq files of 194 mouse sperm cells were downloaded. The downloading process of the rest 23 cells failed and they were not analysed. We applied `fastp-v0.20.1` [48] on the raw reads for filtering out low quality reads and adapter trimming, before applying `minimap2-2.7_x64-linux` [49]

for aligning the reads to the mouse reference genome mm10. The mapped reads were further processed by `GATK MarkDuplicates` and `GATK AddOrReplaceReadGroup` from the `GATK-v4.2` pipeline [50]. Read sorting and indexing were performed using samtools-v1.10 [31, 51]. A customised open-access tool `appendCB` (see Availability of data and materials) was applied to add barcode sequences to each sperm's DNA reads with tag CB using its SRR sequence. The DNA reads in each sperm sample were sub-sampled to retain half of the reads so that the dataset would mimic a droplet-based single-cell DNA sequencing data. The barcode-tagged single-sperm BAM files were then merged and indexed using `samtools` [31].

We followed the main steps described in the original study for finding heterozygous SNPs in the mouse donor's genome [19]. We first called variants *de novo* on the bulk sperm sample 'SRR8454653' (sequenced DNA reads of pooled multiple sperm) using `GATK-HaplotypeCaller`. Only the hetSNPs with `MQ>50 AND DP>10 AND DP<80` were kept. Since the mouse donor was an F1 hybrid of (C57BL/6J X CAST/EiJ), the list of reference hetSNPs was downloaded from the dbSNP database (`CAST_EiJ.mgp.v5.snps.dbSNP142.vcf.gz` and `C57BL_6NJ.mgp.v5.snps.dbSNP142.vcf.gz`) from the Mouse Genome Project [52]. The called variants in sample SRR8454653 were further filtered to only keep the positions which were called as homozygous alternative (`GT==1/1`) in `CAST_EiJ.mgp.v5.snps.dbSNP142.vcf.gz` and not overlapping with variants in `C57BL_6NJ.mgp.v5.snps.dbSNP142.vcf.gz`. Scripts are publicly available (see Availability of data and materials).

10X scCNV apricot gametes

Two experiments were conducted to sequence the apricot gametes in the original published study [18]. The pre-aligned BAM files (of two experiments) were downloaded from European Nucleotide Archive (ENA) under accession number "PRJEB37669". The downloaded pre-aligned BAM files were converted to fastq reads with `samtools-v1.10` while keeping the cell barcode tags to read sequence names. Reads in fastq files were then mapped to the published haploid genome "Currot" using `minimap2-2.7_x64-linux`. The identification of hetSNP markers was performed by running `bcftools-v1.10` on the pooled DNA reads from the two experiments. The identified hetSNPs were filtered by `QUAL > 200 & FORMAT/DP < 280 & FORMAT/DP > 120 & FORMAT/GT=='0/1' & (FORMAT/AD[0:1])/(FORMAT/DP) > 0.38 & (FORMAT/AD[0:1])/(FORMAT/DP) < 0.62` following steps detailed in the original study [18]. Gametes from the two experiments were merged in the analysis and gametes with barcode collisions were removed. Scripts are publicly available (see Availability of data and materials).

Phasing performance comparison

Performance of *sgcocaller phase* was compared to Hapi [36] on the 11 human sperm dataset [28] and on the apricot gametes generated by the 10X scCNV protocol [18].

11 human sperm dataset[28]

11 datasets were constructed by leaving one sperm out from the 11 available sperm cells in the human sperm dataset [28]. The list of unphased hetSNPs

and phased results (used as ground truth) were downloaded from the supplementary dataset shared in the published study [28]. The parameter settings of each method were kept consistent when applying the two methods on the 11 datasets. Hapi was run following the tutorial example and the allowNA parameter in imputing missing genotypes function was set to 3. *sgcocaller phase* was run using options: `--threads 4 --barcodeTag CB --minDP 2 --maxDP 50 --maxTotalDP 200 --minTotalDP 6 --maxDissim 0.099--minSNPdepth 1 --maxExpand 1000 --posteriorProbMin 0.9` for each chromosome in each dataset. *sgcocaller swphase* was called for all chromosomes with `--lookBeyondSnps 150 --minSwitchScore 580` and `--minPositiveSwitchScores 100`.

10X scCNV sequenced apricot gametes

For the apricot dataset [18], 10 datasets were constructed by leaving out 10% of the gametes each time. The haploid genome assembly"Currot" published previously was used as the haplotype ground truth. The alleles in the list of called hetSNPs was swapped for every other position to create the list of unphased hetSNPs. Due to the sparsity of the dataset, the `allowNA` argument in the `hapiImupte` function of Hapi was raised to 350 to avoid triggering running errors. *sgcocaller phase* was applied with options `--threads 4 --barcodeTag CB --minDP 2 --maxDP 10 --maxTotalDP 80 --minTotalDP 6 --minSNPdepth 1 --posteriorProbMin 0.99` and *sgcocaller swphase* was called with default settings.

Calling crossovers

Mouse sperm dataset

Crossover results for mouse sperm data were obtained from calling *sgcocaller xo* on the prepared hetSNPs and BAM file for 194 sperm cells as described before (see Methods). *sgcocaller xo* was called with options: `--maxTotalDP 450 --maxDP 10 --thetaREF 0.1 --thetaALT 0.9`. The final crossover intervals were identified using *comapr* with detailed code available in a public GitLab repository (see Availability of data and materials) with following filtering settings.

Segment level filters:

1. minSNP=30, the segment that results in one/two crossovers has to have more than 30 SNPs of support.
2. minlogllRatio=150, the segment that results in one/two crossovers has to have logllRatio larger than 150.
3. bpDist=10^5, the segment that results in one/two crossovers has to have base pair distances larger than 10^5.

Cell level filters:

1. maxRawCO=55, the maximum number of raw crossovers allowed on a chromosome (excessive number of crossovers indicate abnormal chromosomes. Cells with any chromosome having more crossovers than the *maxRawCO* are be excluded.
2. minCellSNP=200, there has to be more than 200 SNPs detected within a cell per chromosome, otherwise this cell is removed.

Apricot data with known haplotype
sgcocaller xo was applied on 367 apricot gametes to identify crossovers using the hetSNPs called (see Methods) with option: `--thetaREF 0.3 --thetaALT 0.7 --maxTotalDP 280 --maxDP 30 --minTotalDP 20 --minDP 2 --cmPmb 1e-12`. The following fitering thresholds were applied when parsing and interpreting crossover intervals from *xo*'s results: minSNP=10, minlogllRatio=5, maxRawCO = 5, minCellSNP=200, bpDist=10^6. `bpDist` was set to 10^5 for chromosome 1 and scaled by relative chromosome size for the rest of the chromosomes.

Apricot data with sgcocaller phased haplotype
sgcocaller sxo was applied on sgcocaller phased haplotypes and used the generated output files from the previous step, thus avoiding redundant processing including parsing DNA alignment files. The following options were applied `--minTotalDP 5 --minDP 1 --thetaREF 0.1 --thetaALT 0.9 --cmPmb 1e-9 --maxDP 6 --maxTotalDP 50`. The following fitering thresholds were applied when parsing and interpreting crossover intervals from *sxo*'s results: minSNP = 5, minCellSNP = 80, maxRawCO = 5, minlogllRatio = 0, bpDist=10^6. `bpDist` was set to 10^5 for chromosome 1 and scaled by relative chromosome size for the rest of the chromosomes.

Availability of data and materials

Source code of the latest version of *sgcocaller* is publicly available at a GitLab repository and *comapr* at a GitHub repository under a MIT license. *comapr* is also available as a Bioconductor package. The datasets used in this study are public datasets with raw sequencing data downloaded from GEO with accession GSE125326 and ENA with project ID PRJEB37669. The analysis steps including preprocessing and scripts for generating the figures are openly accessible at GitLab repositories and workflowr [53] web pages demonstrating the full workflow of applying *sgcocaller—comapr* are available. The source code and analysis repositories included in this study are listed below:

- sgcocaller: https://gitlab.svi.edu.au/biocellgen-public/sgcocaller
- comapr: https://github.com/ruqianl/comapr and https://bioconductor.org/packages/comapr
- Mouse sperm data analysis: https://gitlab.svi.edu.au/biocellgen-public/hinch-single-sperm-DNA-seq-processing
- Apricot gamete analysis: https://gitlab.svi.edu.au/biocellgen-public/calling-crossover-from-single-gamete-sequencing-of-apricot
- appendCB: https://github.com/ruqianl/appendCB
- distributions: https://github.com/ruqianl/distributions

Author details
[1]Bioinformatics and Cellular Genomics, St. Vincent's Institute of Medical Research, Melbourne, Australia. [2]Melbourne Integrative Genomics, Faculty of Science, The University of Melbourne, Melbourne, Australia. [3]DNA Repair and Recombination Laboratory, St Vincent's Institute of Medical Research, Melbourne, Australia. [4]The Faculty of Medicine, Dentistry and Health Science, The University of Melbourne, Melbourne, Australia.

References

1. Hunter, N.: Meiotic recombination: The essence of heredity. Cold Spring Harb. Perspect. Biol. **7**(12) (2015)
2. Warren, A.C., Chakravarti, A., Wong, C., Slaugenhaupt, S.A., Halloran, S.L., Watkins, P.C., Metaxotou, C., Antonarakis, S.E.: **Evidence for reduced recombination on the nondisjoined chromosomes 21 in down syndrome**. *Science* **237**(4815), 652–654 (1987). doi:10.1126/science.2955519

3. Sherman, S.L., Takaesu, N., Freeman, S.B., Grantham, M., Phillips, C., Blackston, R.D., Jacobs, P.A., Cockwell, A.E., Freeman, V., Uchida, I., Mikkelsen, M., Kurnit, D.M., Buraczynska, M., Keats, B.J.B., Hassold, T.J.: **Trisomy 21: Association between reduced recombination and nondisjunction**. *American Journal of Human Genetics* **49**(3), 608–620 (1991)
4. Berg, I.L., Neumann, R., Sarbajna, S., Odenthal-Hesse, L., Butler, N.J., Jeffreys, A.J.: **Variants of the protein PRDM9 differentially regulate a set of human meiotic recombination hotspots highly active in african populations**. *Proc. Natl. Acad. Sci. U. S. A.* **108**(30), 12378–12383 (2011)
5. Kong, A., Thorleifsson, G., Gudbjartsson, D.F., Masson, G., Sigurdsson, A., Jonasdottir, A., Walters, G.B., Jonasdottir, A., Gylfason, A., Kristinsson, K.T., Gudjonsson, S.A., Frigge, M.L., Helgason, A., Thorsteinsdottir, U., Stefansson, K.: **Fine-scale recombination rate differences between sexes, populations and individuals**. *Nature* **467**(7319), 1099–1103 (2010)
6. Parvanov, E.D., Petkov, P.M., Paigen, K.: **Prdm9 controls activation of mammalian recombination hotspots**. *Science* **327**(5967), 835–835 (2010)
7. Bell, A.D., Mello, C.J., Nemesh, J., Brumbaugh, S.A., Wysoker, A., McCarroll, S.A.: **Insights into variation in meiosis from 31,228 human sperm genomes**. *Nature* **583**(7815), 259–264 (2020)
8. Bhérer, C., Campbell, C.L., Auton, A.: **Refined genetic maps reveal sexual dimorphism in human meiotic recombination at multiple scales**. *Nat. Commun.* **8**, 14994 (2017)
9. Lyu, R., Tsui, V., McCarthy, D.J., Crismani, W.: **Personalized genome structure via single gamete sequencing**. *Genome Biol.* **22**(1), 112 (2021)
10. Myers, S., Bowden, R., Tumian, A., Bontrop, R.E., Freeman, C., MacFie, T.S., McVean, G., Donnelly, P.: **Drive against hotspot motifs in primates implicates the PRDM9 gene in meiotic recombination**. *Science* **327**(5967), 876–879 (2010)
11. Baudat, F., Buard, J., Grey, C., Fledel-Alon, A., Ober, C., Przeworski, M., Coop, G., de Massy, B.: **PRDM9 is a major determinant of meiotic recombination hotspots in humans and mice**. *Science* **327**(5967), 836–840 (2010)
12. Girard, C., Chelysheva, L., Choinard, S., Froger, N., Macaisne, N., Lehmemdi, A., Mazel, J., Crismani, W., Mercier, R.: **AAA-ATPase FIDGETIN-LIKE 1 and Helicase FANCM Antagonize Meiotic Crossovers by Distinct Mechanisms**. *PLOS Genetics* **11**(7), 1005369 (2015). doi:10.1371/journal.pgen.1005369
13. Séguéla-Arnaud, M., Crismani, W., Larchevêque, C., Mazel, J., Froger, N., Choinard, S., Lemhemdi, A., Macaisne, N., Van Leene, J., Gevaert, K., De Jaeger, G., Chelysheva, L., Mercier, R.: **Multiple mechanisms limit meiotic crossovers: TOP3α and two BLM homologs antagonize crossovers in parallel to FANCM**. *Proceedings of the National Academy of Sciences* **112**(15), 4713–4718 (2015). doi:10.1073/pnas.1423107112
14. Crismani, W., Girard, C., Froger, N., Pradillo, M., Santos, J.L.L., Chelysheva, L., Copenhaver, G.P., Horlow, C., Mercier, R.: **FANCM Limits Meiotic Crossovers**. *Science* **336**(6088), 1588–1590 (2012). doi:10.1126/science.1220381
15. Hinch, A.G., Tandon, A., Patterson, N., Song, Y., Rohland, N., Palmer, C.D., Chen, G.K., Wang, K., Buxbaum, S.G., Akylbekova, E.L., Aldrich, M.C., Ambrosone, C.B., Amos, C., Bandera, E.V., Berndt, S.I., Bernstein, L., Blot, W.J., Bock, C.H., Boerwinkle, E., Cai, Q., Caporaso, N., Casey, G., Cupples, L.A., Deming, S.L., Diver, W.R., Divers, J., Fornage, M., Gillanders, E.M., Glessner, J., Harris, C.C., Hu, J.J., Ingles, S.A., Isaacs, W., John, E.M., Kao, W.H.L., Keating, B., Kittles, R.A., Kolonel, L.N., Larkin, E., Le Marchand, L., McNeill, L.H., Millikan, R.C., Murphy, A., Musani, S., Neslund-Dudas, C., Nyante, S., Papanicolaou, G.J., Press, M.F., Psaty, B.M., Reiner, A.P., Rich, S.S., Rodriguez-Gil, J.L., Rotter, J.I., Rybicki, B.A., Schwartz, A.G., Signorello, L.B., Spitz, M., Strom, S.S., Thun, M.J., Tucker, M.A., Wang, Z., Wiencke, J.K., Witte, J.S., Wrensch, M., Wu, X., Yamamura, Y., Zanetti, K.A., Zheng, W., Ziegler, R.G., Zhu, X., Redline, S., Hirschhorn, J.N., Henderson, B.E., Taylor, H.A. Jr, Price, A.L., Hakonarson, H., Chanock, S.J., Haiman, C.A., Wilson, J.G., Reich, D., Myers, S.R.: **The landscape of recombination in african americans**. *Nature* **476**(7359), 170–175 (2011)
16. Spence, J.P., Song, Y.S.: **Inference and analysis of population-specific fine-scale recombination maps across 26 diverse human populations**. *Sci Adv* **5**(10), 9206 (2019)
17. Broman, K.W., Murray, J.C., Sheffield, V.C., White, R.L., Weber, J.L.: **Comprehensive human genetic maps: individual and sex-specific variation in recombination**. *Am. J. Hum. Genet.* **63**(3), 861–869 (1998)
18. Campoy, J.A., Sun, H., Goel, M., Jiao, W.-B., Folz-Donahue, K., Wang, N., Rubio, M., Liu, C., Kukat, C., Ruiz, D., Huettel, B., Schneeberger, K.: **Gamete binning: chromosome-level and haplotype-resolved genome assembly enabled by high-throughput single-cell sequencing of gamete genomes**. *Genome Biol.* **21**(1), 306 (2020)
19. Hinch, A.G., Zhang, G., Becker, P.W., Moralli, D., Hinch, R., Davies, B., Bowden, R., Donnelly, P.: Factors influencing meiotic recombination revealed by whole-genome sequencing of single sperm. Science **363**(6433) (2019)
20. Pedersen, B.S., Quinlan, A.R.: **hts-nim: scripting high-performance genomic analyses**. *Bioinformatics* **34**(19), 3387–3389 (2018)
21. R Core Team: R: A Language and Environment for Statistical Computing, Vienna, Austria (2020). https://www.{R-project}.org
22. Kaminow, B., Yunusov, D., Dobin, A.: STARsolo: accurate, fast and versatile mapping/quantification of single-cell and single-nucleus RNA-seq data (2021)
23. Hahne, F., Ivanek, R.: Statistical Genomics: Methods and Protocols. Springer, New York, NY (2016). doi:10.1007/978-1-4939-3578-9_16. http://dx.doi.org/10.1007/978-1-4939-3578-9_16
24. Lu, S., Zong, C., Fan, W., Yang, M., Li, J., Chapman, A.R., Zhu, P., Hu, X., Xu, L., Yan, L., Bai, F., Qiao, J., Tang, F., Li, R., Xie, X.S.: **Probing meiotic recombination and aneuploidy of single sperm cells by whole-genome sequencing**. *Science* **338**(6114), 1627–1630 (2012)
25. Wickham, H.: *Ggplot2: Elegant Graphics for Data Analysis*. Springer, New York, NY (2009)
26. Wood, S.N.: **Fast stable restricted maximum likelihood and marginal likelihood estimation of semiparametric generalized linear models**. *Journal of the Royal Statistical Society (B)* **73**(1), 3–36 (2011)
27. Viterbi, A.: **Error bounds for convolutional codes and an asymptotically optimum decoding algorithm**. *IEEE*

Trans. Inf. Theory **13**(2), 260–269 (1967)
28. Kirkness, E.F., Grindberg, R.V., Yee-Greenbaum, J., Marshall, C.R., Scherer, S.W., Lasken, R.S., Venter, J.C.: **Sequencing of isolated sperm cells for direct haplotyping of a human genome**. *Genome Res.* **23**(5), 826–832 (2013)
29. Kosambi, D.D.: **THE ESTIMATION OF MAP DISTANCES FROM RECOMBINATION VALUES**. *Ann. Eugen.* **12**(1), 172–175 (1943)
30. Li, H.: **A statistical framework for SNP calling, mutation discovery, association mapping and population genetical parameter estimation from sequencing data**. *Bioinformatics* **27**(21), 2987–2993 (2011)
31. Li, H., Handsaker, B., Wysoker, A., Fennell, T., Ruan, J., Homer, N., Marth, G., Abecasis, G., Durbin, R., 1000 Genome Project Data Processing Subgroup: **The sequence Alignment/Map format and SAMtools**. *Bioinformatics* **25**(16), 2078–2079 (2009)
32. Zhou, Y., Shen, B., Jiang, J., Padhi, A., Park, K.-E., Oswalt, A., Sattler, C.G., Telugu, B.P., Chen, H., Cole, J.B., Liu, G.E., Ma, L.: **Construction of PRDM9 allele-specific recombination maps in cattle using large-scale pedigree analysis and genome-wide single sperm genomics**. *DNA Res.* **25**(2), 183–194 (2018)
33. Roach, J.C., Glusman, G., Hubley, R., Montsaroff, S.Z., Holloway, A.K., Mauldin, D.E., Srivastava, D., Garg, V., Pollard, K.S., Galas, D.J., Hood, L., Smit, A.F.A.: **Chromosomal haplotypes by genetic phasing of human families**. *Am. J. Hum. Genet.* **89**(3), 382–397 (2011)
34. Rowan, B.A., Patel, V., Weigel, D., Schneeberger, K.: **Rapid and inexpensive whole-genome genotyping-by-sequencing for crossover localization and fine-scale genetic mapping**. *G3* **5**(3), 385–398 (2015)
35. Hou, Y., Fan, W., Yan, L., Li, R., Lian, Y., Huang, J., Li, J., Xu, L., Tang, F., Xie, X.S., Qiao, J.: **Genome analyses of single human oocytes**. *Cell* **155**(7), 1492–1506 (2013)
36. Li, R., Qu, H., Chen, J., Wang, S., Chater, J.M., Zhang, L., Wei, J., Zhang, Y.-M., Xu, C., Zhong, W.-D., Zhu, J., Lu, J., Feng, Y., Chen, W., Ma, R., Ferrante, S.P., Roose, M.L., Jia, Z.: **Inference of Chromosome-Length haplotypes using genomic data of three or a few more single gametes**. *Mol. Biol. Evol.* **37**(12), 3684–3698 (2020)
37. Mölder, F., Jablonski, K.P., Letcher, B., Hall, M.B., Tomkins-Tinch, C.H., Sochat, V., Forster, J., Lee, S., Twardziok, S.O., Kanitz, A., Wilm, A., Holtgrewe, M., Rahmann, S., Nahnsen, S., Köster, J.: **Sustainable data analysis with snakemake**. *F1000Res.* **10**(33), 33 (2021)
38. Tewhey, R., Bansal, V., Torkamani, A., Topol, E.J., Schork, N.J.: **The importance of phase information for human genomics**. *Nature reviews. Genetics* **12**(3), 215–223 (2011). doi:10.1038/nrg2950
39. Snyder, M.W., Adey, A., Kitzman, J.O., Shendure, J.: **Haplotype-resolved genome sequencing: experimental methods and applications**. *Nature reviews. Genetics* **16**(6), 344–358 (2015). doi:10.1038/nrg3903
40. Ferguson, K.A., Wong, E.C., Chow, V., Nigro, M., Ma, S.: **Abnormal meiotic recombination in infertile men and its association with sperm aneuploidy**. *Hum. Mol. Genet.* **16**(23), 2870–2879 (2007)
41. Edge, P., Bafna, V., Bansal, V.: **HapCUT2: robust and accurate haplotype assembly for diverse sequencing technologies**. *Genome Res.* **27**(5), 801–812 (2017)
42. Morgan, M., Obenchain, V., Hester, J., Pagès, H.: SummarizedExperiment: SummarizedExperiment Container. (2020). R package version 1.17.5
43. Hahne, F., Ivanek, R.: **Visualizing genomic data using gviz and bioconductor**. In: Mathé, E., Davis, S. (eds.) *Statistical Genomics: Methods and Protocols*, pp. 335–351. Springer, New York, NY (2016)
44. Phipson, B., Smyth, G.K.: **Permutation p-values should never be zero: calculating exact p-values when permutations are randomly drawn**. *Stat. Appl. Genet. Mol. Biol.* **9**, 39 (2010)
45. Efron, B., Tibshirani, R.J.: An introduction to the bootstrap chapman & hall. New York **436** (1993)
46. Martin, M.A.: An introduction to bootstrap methods with applications to R by M.R. chernick and R.A. LaBudde. Aust. N. Z. J. Stat. **54**(2) (2012)
47. Phipson, B., Smyth, G.K.: **Permutation p-values should never be zero: calculating exact p-values when permutations are randomly drawn**. *Statistical Applications in Genetics and Molecular Biology* **9**(1), 39 (2010)
48. Chen, S., Zhou, Y., Chen, Y., Gu, J.: **fastp: an ultra-fast all-in-one FASTQ preprocessor**. *Bioinformatics* **34**(17), 884–890 (2018)
49. Li, H.: **Minimap2: pairwise alignment for nucleotide sequences**. *Bioinformatics* **34**(18), 3094–3100 (2018)
50. McKenna, A., Hanna, M., Banks, E., Sivachenko, A., Cibulskis, K., Kernytsky, A., Garimella, K., Altshuler, D., Gabriel, S., Daly, M., DePristo, M.A.: **The genome analysis toolkit: a MapReduce framework for analyzing next-generation DNA sequencing data**. *Genome Res.* **20**(9), 1297–1303 (2010)
51. Danecek, P., Bonfield, J.K., Liddle, J., Marshall, J., Ohan, V., Pollard, M.O., Whitwham, A., Keane, T., McCarthy, S.A., Davies, R.M., Li, H.: Twelve years of SAMtools and BCFtools (2020). 2012.10295
52. Keane, T.M., Goodstadt, L., Danecek, P., White, M.A., Wong, K., Yalcin, B., Heger, A., Agam, A., Slater, G., Goodson, M., Furlotte, N.A., Eskin, E., Nellåker, C., Whitley, H., Cleak, J., Janowitz, D., Hernandez-Pliego, P., Edwards, A., Belgard, T.G., Oliver, P.L., McIntyre, R.E., Bhomra, A., Nicod, J., Gan, X., Yuan, W., van der Weyden, L., Steward, C.A., Bala, S., Stalker, J., Mott, R., Durbin, R., Jackson, I.J., Czechanski, A., Guerra-Assunção, J.A., Donahue, L.R., Reinholdt, L.G., Payseur, B.A., Ponting, C.P., Birney, E., Flint, J., Adams, D.J.: **Mouse genomic variation and its effect on phenotypes and gene regulation**. *Nature* **477**(7364), 289–294 (2011)
53. Blischak, J.D., Carbonetto, P., Stephens, M.: Creating and sharing reproducible research code the workflowr way [version 1; peer review: 3 approved]. F1000Research **8**(1749) (2019). doi:10.12688/f1000research.20843.1

Ethics approval and consent to participate
Not applicable.

Competing interests
The authors declare that they have no competing interests.

Acknowledgements
Thanks to Brendan Hill for reviewing the source code of sgcocaller. The double Holliday junction in the *comapr* Hex Logo was made with affinity designer and Biorender.

Authors' contributions
R.L developed the software and wrote the manuscript. V.T and W.C generated testing datasets for prototyping the software and reviewed the manuscript. W.C and C.L contributed ideas in designing tools' functions and C.L reviewed comapr. H.S and D.J.M supervised this project.

Funding
W.C and D.J.M receive funding from the Australian National Health and Medical Research Council (GNT1129757, GNT1112681, GNT1185387, GNT1195595). D.J.M is further supported by the Baker Foundation, and by Paul Holyoake and Marg Downey through a gift to St Vincent's Institute of Medical Research. R.L and V.T are recipients of a Research Training Program Scholarship from the Australian Commonwealth Government and the University of Melbourne and SVI Foundation Top-Up Scholarship from St Vincent's Institute. R.L receives a Xing Lei PhD Top-up Scholarship in Mathematics and Statistics. V.T receives a St Vincent's Institute Top-Up scholarship.

Additional Files
Additional file 1 — SupplementaryFigures.pdf
Additional file descriptions text (including details of how to view the file, if it is in a non-standard format or the file extension). This might refer to a multi-page table or a figure.

Additional file 2 — Sample additional file title
Additional file descriptions text.

Supplementary figures

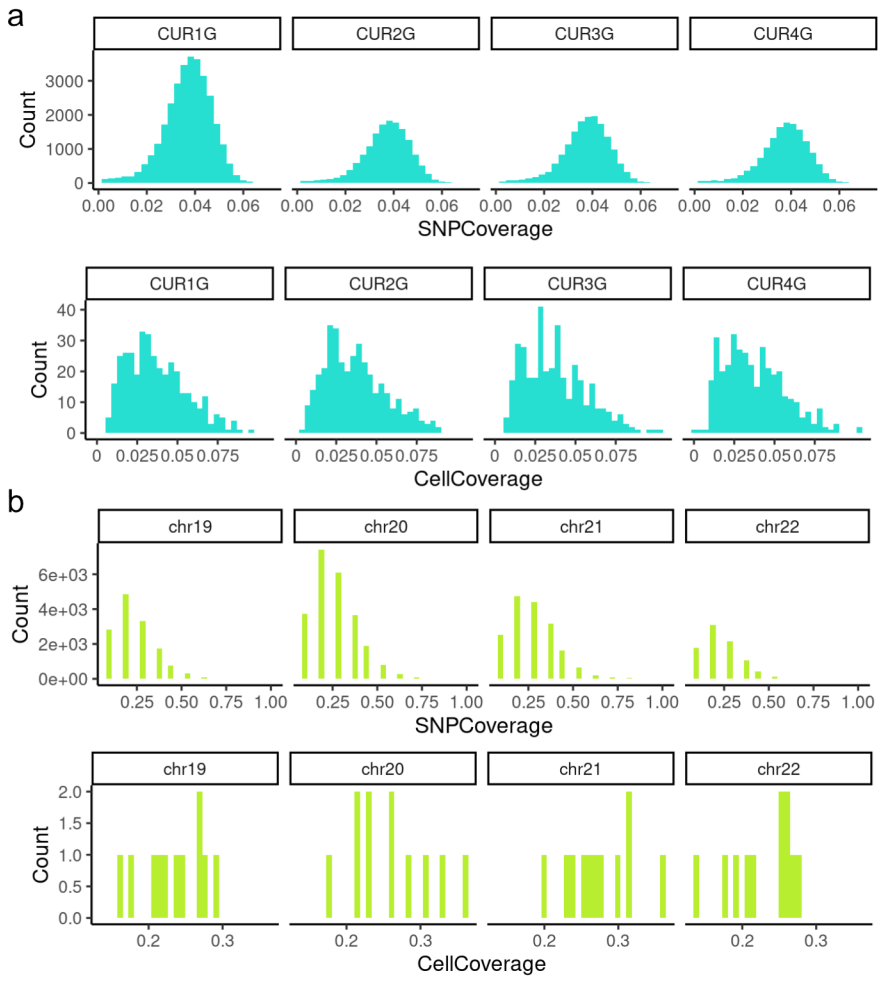

Figure S1 Dataset comparison of human sperm data with apricot pollen data a) SNPs' coverage per cell and cells' coverage per SNP are plotted for apricot pollen data [18]. b) Same plots but for human sperm sequencing data [28]

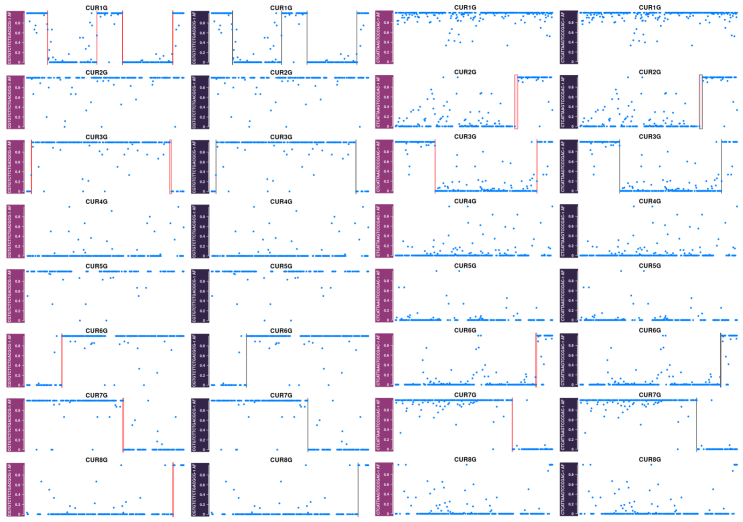

Figure S2 Crossovers called by sgcocaller (left column) and from published study (right column) for two randomly selected apricot pollen

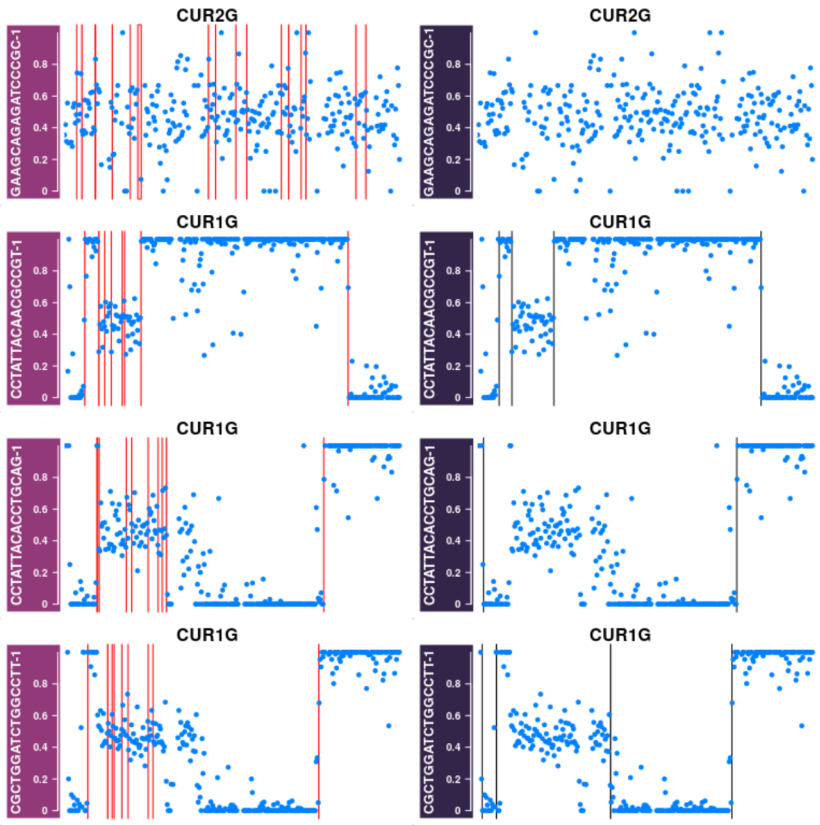

Figure S3 Examples of chromosomes with excessive crossovers called by sgcocaller xo and filtered in analysis (left column). Right column plots the crossover results released by previous study [18]

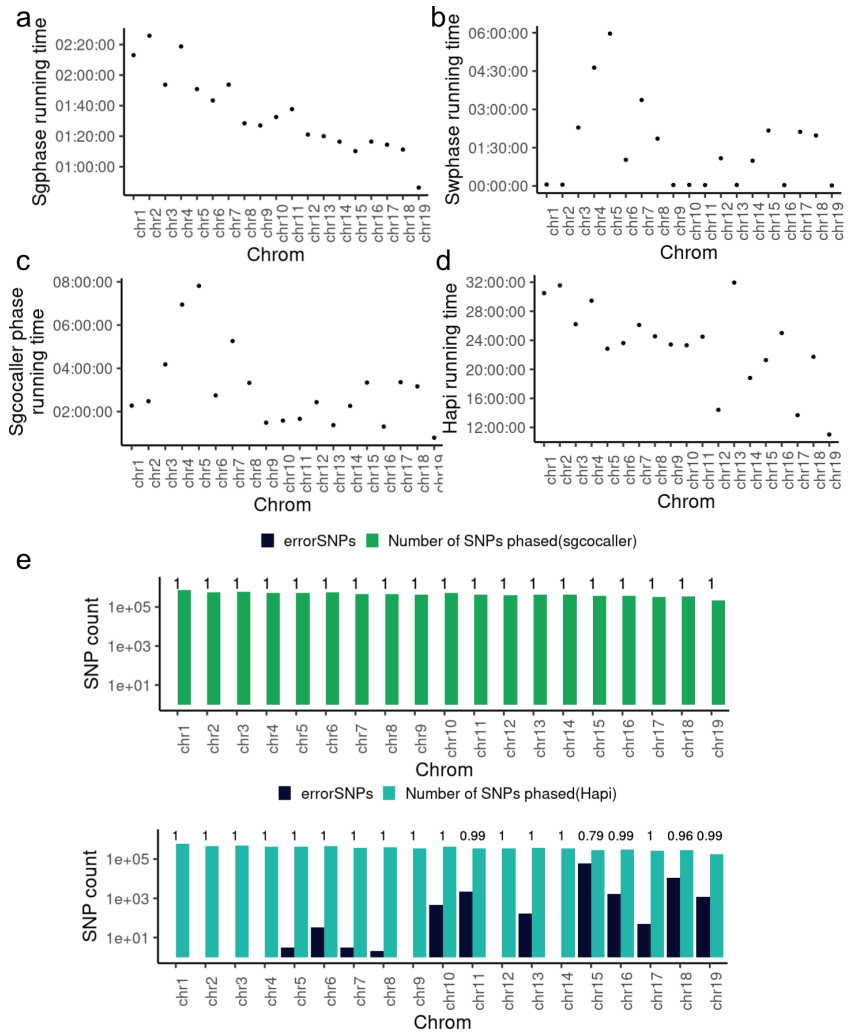

Figure S4 Phasing running time and accuracy comparison by sgcocaller and Hapi on mouse sperm dataset [19]a), Running time for each chromosome taken by *sgcocaller phase*.b), Running time for each chromosome taken by *sgcocaller swphase*. c), Running time for each chromosome taken by *sgcocaller phase* plus *sgcocaller swphase*. d), Running time for each chromosome taken by Hapi. e) Phasing accuracies for each chromosomes by two methods on mouse sperm dataset (n=194 sperm cells) [19]

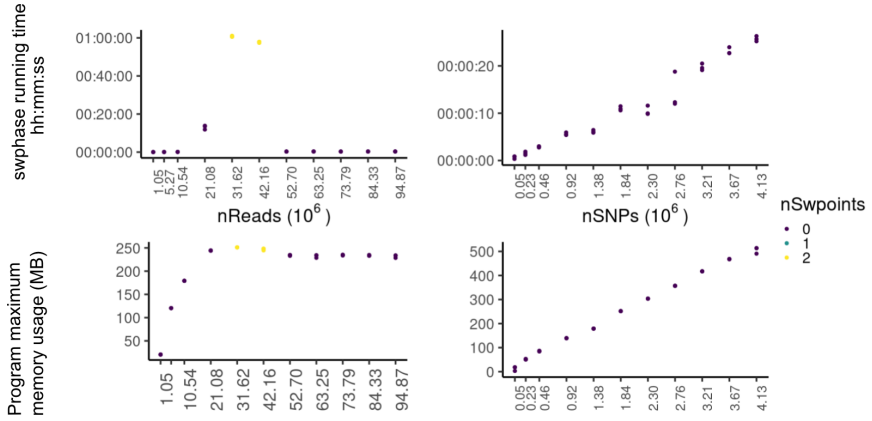

Figure S5 The scalability of sgcocaller swphase Measurements were generated from running *sgcocaller swphase* on a linux server with `RedHat7.9(Maipo)`.The measurements of time and memory usage for different inputs were obtained using the "benchmark" function from the workflow manager `Snakemake` [37]. Time was measured in seconds and memory usage in megabyte (MB). max_rss (resident set size) is the maximum amount of memory occupied by a process at any time that is held in main memory (RAM).'nSwitchpoints' is the number of switch points identified.