Multi-Sample Approaches and Applications for Structural Variant Detection

Dissertation

zur Erlangung des Grades eines
Doktors der Naturwissenschaften
(Dr. rer. nat.)

am Fachbereich Mathematik und Informatik
der Freien Universität Berlin

vorgelegt von
Sebastian Niehus

Berlin 2022

Erstgutachter: Prof. Dr. Knut Reinert
Zweitgutachterin: Prof. Dr. Birte Kehr

Tag der Disputation: 19.07.2022

Declaration of Authorship

Name: Niehus
First name: Sebastian

I declare to the Freie Universität Berlin that I have completed the submitted dissertation independently and without the use of sources and aids other than those indicated. The present thesis is free of plagiarism. I have marked as such all statements that are taken literally or in content from other writings. This dissertation has not been submitted in the same or similar form in any previous doctoral procedure.

I agree to have my thesis examined by a plagiarism examination software.

Date: 23.05.2022 Signature:

Abstract

In recent years, advances in the field of sequencing technologies have enabled the field of population-scale sequencing studies. These studies aim to sequence and analyze a large set of individuals from one or multiple populations, with the aim of gaining insight into underlying genetic structure, similarities and differences. Collections of genetic variation and possible connections to various disease are some of the products of this area of research. The potential of population studies is widely considered to be huge and many more endeavors of this kind are expected in the near future. This opportunity comes with a big challenge because many computational tools that are used for the analysis of sequencing data were not designed for cohorts of this size and may suffer from limited scalability. It is therefore vital that the computational tools required for the analysis of population-scale data keep up with the quickly growing amounts of data.

This thesis contributes to the field of population-scale genetics in the development and application of a novel approach for structural variant detection. It has explicitly been designed with the large amounts of population-scale sequencing data in mind. The presented approach is capable of analyzing tens of thousands of whole-genome short-read sequencing samples jointly. This joint analysis is driven by a tailored joint likelihood ratio model that integrates information from many genomes. The efficient approach does not only save computational resources but also allows to combine the data across all samples to make sensitive and specific predictions about the presence and genotypes of structural variation present within the analyzed population. This thesis demonstrates that this approach and the computational tool *PopDel* that implements it compare favorably to current state-of-the-art structural variant callers that have been used in previous population-scale studies. Extensive benchmarks on simulated and real world sequencing data are provided to show the performance of the presented approach. Further, a first finding of medical relevance that directly stems from the application of *PopDel* on the genomes of almost 50,000 Icelanders is presented.

This thesis therefore provides a novel tool and new ideas to further push the boundaries of the analysis of massive amounts of next generation sequencing data and to deepen our understanding of structural variation and their implications for human health.

Zusammenfassung

Die Fortschritte im Bereich der Sequenziertechnologien gestatteten die Verbreitung von Sequenzierstudien auf Populationsniveau. Diese Studien zielen darauf ab, eine große Zahl an Individuen aus einer oder mehreren Populationen zu sequenzieren und zu analysieren, um Einblicke in zugrundeliegende genetische Strukturen, Gemeinsamkeiten sowie Unterschiede zu gewinnen. Sammlungen genetischer Varianten und deren mögliche Verbindungen zu einer Vielzahl an Krankheiten sind nur eines der Ergebnisse dieses Forschungsfeldes. Das Potential der Populationsstudien wird weithin als groß angesehen und viele weitere Anstrengungen dieser Art können in naher Zukunft erwartet werden. Dies stellt eine große Gelegenheit dar, die jedoch auch eine große Herausforderung mit sich bringt, da viele Computerprogramme, welche für die Analyse für Sequenzierdaten verwendet werden, nicht für Kohorten in der Größenordnung von Populationen entworfen wurden und folglich einen Mangel an Skalierbarkeit auf große Datensätze aufweisen können. Daher ist es für die Zukunft wichtig, dass die für die Analyse auf Populationsniveau notwendigen Computerprogramme mit den rasant wachsenden Datenmengen mithalten können.

Die vorliegende Dissertation leistet einen Beitrag zum Feld der Populationsgenetik durch die Entwicklung und Implementierung eines neues Ansatzes zum Auffinden von Strukturvarianten, der explizit im Hinblick auf die riesigen Mengen an Sequenzierdaten der Populationsstudien entworfen wurde. Dieser Ansatz ermöglicht es, zehntausende Genome aus *whole-genome short-read sequencing* Experimenten gemeinsam zu analysieren. Im Zentrum dieser Analyse steht ein Likelihood-Quotienten-Modell, welches Informationen aus vielen Genomene integriert. Der präsentierte Ansatz schont nicht nur Rechenressourcen, sondern ermöglich es zudem die Daten aller analysierten Genome zu kombinieren, um sensitive wie zuverlässige Aussagen über das Vorhandensein von Strutkturvarianten und deren Genotypen in der analysierten Population zu treffen. Diese Dissertation zeigt, dass dieser Ansatz, welcher in Form des Computerprogramms *PopDel* implementiert wurde, im Vergleich mit anderen dem Stand der Technik entsprechenden Programmen zum Auffinden von Strukturvarianten, die bereits in Studien auf Populationsniveau Anwendung fanden, gut abschneidet. Ausführliche Benchmarks auf simulierten wie auch realen Sequenzdaten belegen die Leistungsfähigkeit des dargelegten Ansatzes. Des Weiteren wird ein erster Fund von medizinischer Relevanz, welcher direkt durch die Anwendung von *PopDel* auf den Genomen von nahezu 50,000 Isländern zustande kam, präsentiert.

Folglich bietet diese Dissertation ein neuartiges Computerprogramm sowie Ideen, um die derzeitigen Grenzen der Analyse enormer Mengen an *next generation sequencing* Daten zu überwinden und unsere Einsicht sowie unser Verständnis der Strukturvarianten inlusive ihrer Bedeutung für die menschliche Gesundheit zu erweitern.

Acknowledgements

With these lines I want to take the opportunity to thank everyone who supported me during my time as an undergraduate student and PhD student.

I want to thank my first advisor Prof. Dr. Knut Reinert for mentoring me during my time as an undergraduate student and for pointing me at the open position in the newly established research group of Prof. Dr. Birte Kehr, my second advisor of this thesis.

I want to thank Prof. Dr. Birte Kehr for introducing me to the fascinating field of structural variation and population-scale genetics and giving me the possibility to work in this exciting field. Her availability, guidance and constructive feedback were always helpful and highly appreciated. Her ideas substantially contributed to shaping PopDel into what it is now.

Many thanks go to my co-authors Hákon Jónsson, Janina Schönberger, Eythór Björnsson, Doruk Beyter, Hannes Eggertsson, Patrick Sulem, Kári Stefánsson and Bjarni V. Halldórsson. I owe special thanks to Bjarni for his contributions to the ideas implemented in PopDel, his help during my stay at deCODE genetics in Iceland and his work in applying PopDel on the Icelandic genomes.

I also want to thank all past and current members of Kehr lab, for the productive discussions regarding their or my own research and for being great colleagues in general.

The Core Unit Bioinformatics of the Berlin Institute of Health deserves special thanks for the administration of the BIH HPC cluster, which enabled many of the computations I performed in the course of this thesis.

Lastly, I want to thank my parents and family for always encouraging me and my curiosity and always supporting my choices. I want to thank my wife Theresa for her support and for proofreading the drafts of my manuscript, especially those sections that were written later in the day. Those sections were a very practical demonstration of SVs, because they often enough contained duplicated or missing words and translocated sentence parts. I thank her for her good accuracy in detecting those variants in my manuscript.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Variation of the (human) genome drives phenotypic differences between individuals and populations. In 2015, the 1000 Genomes Project (G1k) established a list of more than 88 million variants from 2,504 human genomes[17]. Genetic variation can be divided into different types: Single nucleotide variants (SNVs), insertions and deletions (indels) of genetic sequence affecting up to 50 base pairs (bp) and structural variants (SVs), typically defined as non-substitution variants affecting more than 50 bp[37]. While SVs seem to be much rarer in the human genome when compared to SNVs and indels[17;37;82], they are playing a key role in the development of our genome and have an important influence on healthy as well as deleterious phenotypic traits and various disease[17;74]. Due to the size and complexity of SVs, their phenotypic influence is often stronger than that of the smaller SNVs and indels[54], making a thorough understanding of SVs and their associated mechanisms vital for the understanding of human development and health.

This endeavor is complicated by the size and complex nature of SVs, which is why the current knowledge of SVs is far behind that of SNVs[54]. New methods are needed for the detection and genotyping of SVs for closing this gap. This thesis contributes to the solution of the SV detection problem by presenting a new approach, PopDel, that was developed to draw information from thousands to tens of thousands of genomes simultaneously to enable a robust detection of SVs in population-scale data, making it especially suited for increasing amounts of sequencing data and the growing number of population-scale sequencing projects[16;37].

1.1 The Human Genome and its Function

Humans, like all living organisms, are to a very big portion defined by their genomes. The genome is the collection of genetic information about an organism and is encoded in *deoxyribonucleic acid* molecules, the DNA. A single DNA molecule has the structure of a double helix. Phosphates and deoxyribose make up the backbone and nucleobases encode for genetic information[2;86]. The nucleobases making up the DNA are adenine, guanine, thymine and cytosine, often encoded as single letters A,G,T and C. One building block of DNA consisting of the backbone and one nucleobase is called *nucleotide*. As the nucleobases of the two DNA helix strands are always paired via hydrogen bonds, they are often referred to as *base pairs* (bp). Under normal conditions, A is always paired with T and C is paired with G. The consecutive sequence of A,T,G and C in the DNA is referred to as *genetic sequence* and is commonly visualized as a sequence of these letters.

The sequence of the human genome comprises 3.2 billion bp. It is organized into 22 *autosomes* and 2 *gonosomes*[2]. Autosomes and gonosomes are *chromosomes*, single molecules of DNA packed into a dense organizational unit. In the healthy human genome, the autosomes are present in two copies each, whereas the gonosomes are present as two copies of the X chromosome for typical females and as one copy of the X chromosome accompanied by one copy of the Y chromosome for typical males. One copy of the chromosomes is inherited from the father and one copy from the mother. The presence of two copies of each autosome makes the human genome *diploid*. The two copies of a chromosome are not identical, but can differ in multiple positions. The alternative states of a gene (or any other locus) are called *alleles*[2]. Consequently, in the diploid human genome each locus is represented by two alleles.

The sections of the genome that encode for functional agents of the organism, e.g. proteins, are called *genes*[2]. A gene can be divided into *introns* and *exons*. Introns are the parts of a gene that do not encode for the sequence of the gene product and exons are the coding parts.

Almost all cells of the human body carry a complete and identical copy of the individual's genome in their *nucleus*, the core compartment of the cell. During a process called *protein biosynthesis*, the genes that are relevant for the function of the cell are "read" by a complex called *ribonucleic acid* (RNA) *polymerase* and transcribed into so called *pre-messenger RNA* (pre-mRNA). Like DNA, RNA is a polymer consisting of a sequence of nucleotides[2]. The main differences are that RNA contains the nucleobase uracil (U) instead of thymine, and that RNA does not contain deoxyribose but plain ribose. This makes RNA less stable than DNA. Further, it is typically present in single stranded form. mRNA is a piece of RNA that encodes for a protein. pre-mRNA still contains the introns of a gene and by removing them in a process called *splicing*, the mRNA is created.

The mRNA is transported outside of the nucleus and serves as a blueprint for a protein[2]. The *ribosomes* connect to the mRNA and translate its sequence into a sequence of *amino acids*, the building blocks of proteins. The translation scheme for this process follows a set genetic code, where each amino acid is encoded by one or more *codons*. One codon consists of the combination of three mRNA bases.

Following the translation, the amino acid polymers fold into proteins and additional post translational modifications are performed. After this, the proteins can commence their designated functions in the organism[2].

This process demonstrates how the genome encodes for the shape and function of all parts of the human body. Variation in the genetic sequence of the genome can therefore have a substantial impact on how the whole organism looks and operates, i.e. the phenotype. The effect of genetic variation also depends on the type of cell it first occurs in. Especially, one has to distinguish between variation in somaticand germline cells. Somatic cells are cells that make up the organisms and are not directly responsible for promoting genetic information to the offspring[2]. Germline cells are cells that later develop into eggs or sperms and whose primary task is the promotion of genetic information to the offspring[2]. If variation occurs in somatic cells, for example due to environmental influences like radiation or errors during the replication process of the cell, this change is typically local and can lead to the death of the cell. In case all control mechanisms of the body fail and more and more genes are affected, this can ultimately lead to the formation of tumors[2]. If variation occurs in germline cells it can be given to the offspring. Here it does not only affect a single cell, but the whole development of the organism and consequentially its appearance and function[2]. While germline variation has many mechanisms and is necessary for the evolution of the species, it can also lead to disease, disability and prenatal death of the developing organism. When speaking of *variation* in the context of this thesis, this always refers to *germline variation* if not stated otherwise.

Another important factor for the effect of a variant is its *zygosity*. If a variant is only present on one chromosomal copy, it is called *heterozygous*. If it is present on both copies of a chromosome, it is called *homozygous*. Depending on the location and type of variant, homozygous variants can have a stronger impact than heterozygous variants. The *genotype* of an individual describes the collection of alleles in the individuals genome. It therefore also describes the zygosity of variants in the genome. When speaking of a specific variant in an individual, the genotype or zygosity of the variant are often encoded as 0/0 (two reference alleles - non carrier), 0/1 (one reference allele and one variant allele - heterozygous carrier), or 1/1 (two variant alleles - homozygous carrier). The genotype that describes the sequence of a single chromosome is called *haploid genotype* or just *haplotype*. Variants from the same haplotype are often inherited together from a parent to its offspring unless a *recombination,* between the two homologous chromosomes of the parent occurs.

Given how our genome and its variation defines us and our lives, it is easy to see why it has been in the focus of research for many years and, considering how much there is still to learn about it, will continue to be so.

1.2 Variation of DNA Sequence: Small and Structural Variation

Variation of the genome is always defined as difference between one or more sequenced genomes and a defined reference genome, often only called the *reference*. Throughout this thesis, the expression *sample* will be used to refer to a sequenced DNA sample that has been taken from one distinct individual, whose genome is also referred to as *sample genome*. For most human studies of the last years this has been the *Genome Reference Consortium Human Build 37*[15] (GRCh37), also called *hg19*, released in 2009. The most recent release *GRCh38*[78], also called *hg38*, which improved over GRCh37 by resolving about 1,000 different shortcomings, like the closing of gaps and the correction of coordinates[78], has seen increased application since its release in 2013. Both GRCh37 and 38 are not the genomes of a single donor, but an admixture of multiple donors with varying proportions of contribution.

Variation of the genomic sequence can be categorized into different types:

Single Nucleotide Variants (SNVs) are variants that alter a single base of the genomic sequence. Given the reference DNA sequence $r =$ 'ACT', and the sample sequence from the same genomic position $s =$ 'AAT', then the change from C to A in s is an SNV. An example schematic for a single base pair substitution is shown in Figure 1.1.

Small Insertions and Deletions (indels) are short (typically defined as affecting less than 50 bp) insertions or deletions of sequence. Given the reference DNA sequence $r =$ 'ACGT' and the sample sequence from the same genomic position $s_d =$ 'AT'. The lack of the bases G and T in s_d is classified as a small deletion. If instead $s_i =$ 'ACCGT' the additional C would show an insertion. In this example, it is not possible to judge if the insertion occurred before or after the originally present C in the sequence. The same ambiguities can be observed for deletions as well, making it often hard to give the true coordinates of an indel, especially in the presence of repetitive sequence. Another important point about insertions is that they do not necessarily always represent sequence that has actually been inserted in the sample sequence, but rather indicate a deletion in the reference genome[41]. This is underlined by the presence of the insertions in genomes of chimpanzees and other primates. Examples for indels are given in Figure 1.1.

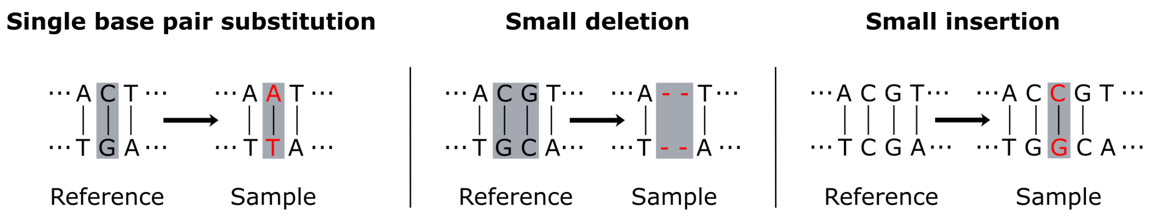
Single base pair substitution
Small deletion
Small insertion
···A C T··· ···A A T···
···T G A··· ···T T A···
···A C G T··· ···A - - T···
···T G C A··· ···T - - A···
···A C G T··· ···A C C G T···
···T C G A··· ···T G G C A···
Reference
Sample

Figure 1.1: Schematics of small variants. Vertical lines between complementary bases indicate the hydrogen bonds between the DNA strands. Changes when comparing the reference sequence to sample sequence are highlighted. Examples are given for single base pair substitutions (left), small deletions (center) and small insertions (right). Please note that the highlighting for the insertion could also be moved one column to the left because it cannot be determined if the insertion occurred before or after the second column.

Structural Variants (SVs) are variants that affect many (typically defined as 50 bp and more) bases of the DNA sequence. They can be further broken down into different types. For visualizations of some SV types, please see Figure 1.2.

Deletions are defined as described for indels above, with the only difference being their larger size.

Insertions, also being larger than their indel counterpart, can be further distinguished: If a piece of sequence is copied and inserted directly after (or before) the original sequence this is called a *tandem duplication*. If the copied piece is inserted somewhere else in the genome, it is called a *dispersed* or *interspersed* duplication. Duplications of the same piece of sequence can further be repeated more than once. If the inserted sequence has not been part of the genome so far, it is called a *non-reference sequence* (NRS) *insertion* .

Inversions describe that a piece of the DNA double strand has been inverted. For the nucleotide sequence this means that the affected piece of the sequence is replaced by its *reverse complement*. The reverse complement of a DNA sequence is gained by reversing the sequence and replacing each letter of the sequence with its complement as determined by the typical pairing of the nucleotides in the DNA. Given the reference DNA sequence $r = r_0, r_1, r_2, r_3 =$ 'ACTT', then the derived sequence gained by inverting substring r_1, r_2 would be $s =$ 'AAGT'.

Translocations are defined as a piece of sequence being moved from its original position to another position. If the new position is on the same chromosome as the original position, this is called an *intrachromosomal* translocation. If the sequence is translocated to another chromosome, it is called an *interchromosomal* translocation. Interchromosomal translocations can be *reciprocal*, meaning that one involved chromosome loses a piece of sequence to the other chromosome, but gains a piece of sequence from the other one instead, basically making this event an exchange. If a reciprocal translocation does not result in gain or loss of any sequence it is called *balanced*. A translocation is *unbalanced* if sequence is lost or gained in the process.

Deletions and insertions affect the number of copies of the DNA sequence an individual carries at the affected positions. They are therefore also called *copy number variants* (CNVs). Variants like inversions and balanced translocations that do not result in gain or loss of sequence are referred to as *copy number neutral* variants.

Complex SVs can be described by a combination of the mentioned canonical SV types. For example, an inversion can be accompanied by the deletion of a piece of flanking sequence, or a translocated piece of sequence can be inverted and duplicated before its insertion into the target chromosome, as shown in Figure 1.2.

There are multiple mechanisms and events that can result in the formation of SVs. They range from failures of DNA repair mechanisms, over replication errors induced by regions with low sequence complexity or high homology, to catastrophic events like the shattering and reassembly of whole chromosomes (chromothripsis)[11]. Some relevant formation mechanisms will be further presented in Section 2.1, together with more details on specific SV types.

Variation of the genomic sequence, both small and structural, does not always lead to phenotypic changes. The consequences of a variant heavily depend on its location in the genome and they type of the variant. If a variant only affects intronic sequence, it is less likely to have an effect than a variant affecting an exon, because the exons are later translated and transcribed into protein sequence. Variation may alter the codons that are translated into a protein sequence, causing other amino acids to be added to a protein or causing a premature stop of the translation process, called a *stop gain*.

The 1000 Genomes Project found that each individual differs in 4.1 to 5 million sites from the reference genome[17]. In their study, small variants (SNVs and indels) make up more than 99.9 % of the variants in a typical human genome. Nevertheless, SVs have been found to affect more bases due to their size. The G1k estimated that a human genome contains about 2,100 - 2,500 SVs that together affect roughly 20 million bases. Most of these SVs are large deletions (1000 variants) and insertions of *Alu elements* (951 variants), a family of transposable elements in the human genome. Other more recent studies that apply more advanced sequencing technologies, now frequently detect between 10,000 and 35,000 SVs per human genome[37], underlining the importance of SVs for human health.

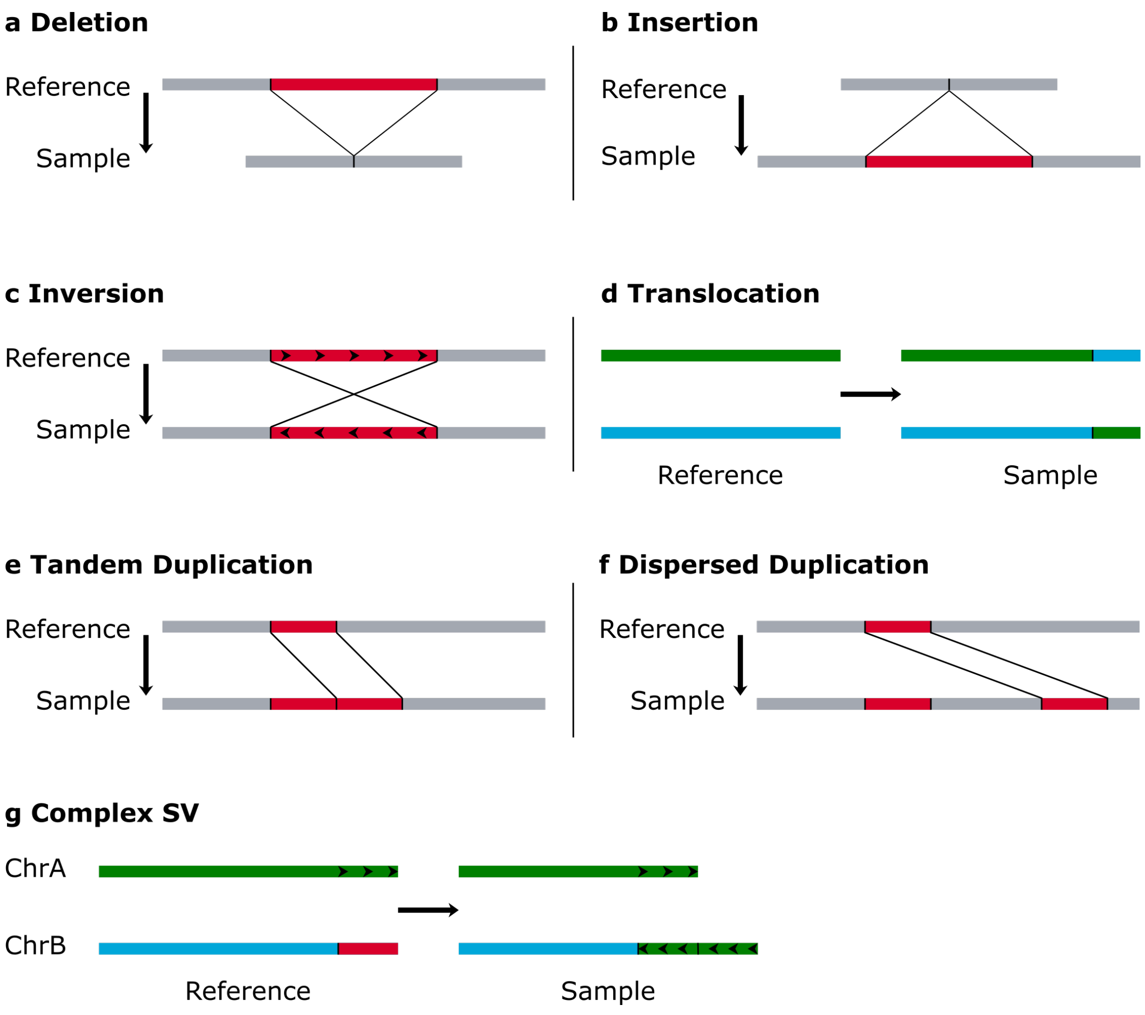

Figure 1.2: Examples for types of SVs. Colored blocks indicate pieces of sequence being modified from reference to sequenced genome. Arrows within the sequence indicate sequence direction. **a** Deletion, **b** Insertion, **c** Inversion, **d** Translocation, **e-f** Duplication, **g** a complex SV involving a piece of sequence being duplicated, inverted and translocated from one chromosome to another, while the target chromosome loses a piece of sequence.

1.3 Genome Sequencing

Genome sequencing is the basis for many modern methods for the analysis of genomes or the detection of genetic variation. Genome sequencing is the process of translating the chemical sequence of the DNA molecules into a human and machine readable format.

The cornerstone for DNA sequencing was laid with the invention of *Sanger sequencing* in the 1970s[76]. Sanger and his colleagues later received a Noble Prize in Chemistry for their work. Sanger sequencing works by taking a DNA fragment template and synthesizing a growing complementary DNA strand consisting of fluorescently (orignally: radioactively) labeled nucleotides. The sequencing is performed in four phases. In each phase another type of nucleotide (A,C,G or T) is labeled. By adding didesoxynukleosidtriphosphates (ddNTPs), the strand synthesis is stopped, resulting in fragments of different lengths. The fragments are sorted by their length via gel electrophoresis and the nucleotide sequence can be inferred from the radioactive labels.

The sequenced parts of the DNA fragments, or rather the sequence gained from them, are called *reads*. Also more recent techniques produce many short reads. Neither Sanger sequencing nor modern methods can process one human chromosome in one go.

The manual and slow process of Sanger sequencing was later automated and enabled the sequencing of the first human genome, that finished in the early 2000s.

Technical and computational advances in the 2000s brought *second generation sequencing*, also called *next generation sequencing* (NGS), which is still widely applied. NGS has a much higher throughput than Sanger sequencing and is mostly automated. One popular protocol for modern NGS is *Illumina paired-end sequencing.* In Illumina paired-end sequencing the DNA is fragmented and special adapters are attached to the fragments. The adapters allow the fragments to bind the *flow cell,* a device that carries the sequencing material in multiple lanes. The fragments are amplified in clusters in a process called *bridge amplification* with the aim of enhancing the later read out signal. The flow cell then undergoes multiple sequencing cycles where, similar to Sanger sequencing, single nucleotides are added to a newly synthesized DNA strand that is complementary to the original fragments. The bases used for the synthesis of the new DNA strands are labeled with fluorescence and after each addition phase the fluorescence signal is captured via a photo sensor. From this signal the added bases of the growing strand can be inferred. The process occurs in hundreds of millions of clusters on the flow cells in parallel, allowing for a high sequencing depth and speed. A special property of paired-end sequencing is that reads are generated starting from both ends of the DNA fragments. The reads typically have a size between 100 - 150 bp. Two reads from of the same fragment make up a read pair. The space between the outer ends of a read pair (without adapters) is called the *insert*. The length of the insert is called *insert size*. Depending on the protocol, a read pair does not necessarily cover the whole DNA fragment it originates from. The pairing of the reads allows for easier placement on the reference genome or assembly of the read pairs into larger contiguous sequences (contigs) because the two reads of a pair are expected to follow a certain distribution regarding their distance to each other. Read pairs also provide the information necessary for most SV detection protocols. Details on those signatures will be further elaborated in Section 2.1.

In the last years, a new generation of sequencing methods has emerged, called *third generation sequencing*, or *long-read sequencing* (LRS). While the exact techniques vary, they are capable of producing increasingly long reads. For example, *Oxford Nanopore Ultra-Long Read technology* can produce single reads that cover more than 1,500,000 bp, albeit their median length is much lower[60]. The increased read length comes at the cost of an increased error rate of the reads and lower sequence coverage. Further, LRS is still substantially more expensive then NGS, but as technology advances the costs are declining. Due to their size, long reads offer a substantial advantage in repeat rich regions of the genome. Also, it is often much easier to detect SVs from LRS data compared to NGS data. This is underlined by the much higher detection rate of SVs in studies involving LRS[16].

This thesis focuses on the currently wider applied NGS data and SV detection methods working with this type of data. Therefore, if not stated otherwise, subsequently mentioned read data is always assumed to be derived from NGS experiments.

1.4 Sequence Alignment and Read Mapping

A basic step in the analysis of NGS (and LRS) data after the sequencing process is the mapping of the reads to the reference genome[6]. The raw reads provide no information about their location on the reference genome, therefore a read mapper has to be applied to find the best location for all sequenced reads. Given the small alphabet size of the DNA, small length of the reads (about 150 bp) and repetitive nature of the human genome, this is a challenging task that does not always have unique solutions for all reads for all regions of the genome.

The problem of comparing a read sequence to the reference sequence can be seen as an instance of the *sequence alignment* problem. In a pairwise sequence alignment, the two compared sequences are typically visualized in rows. The columns of the alignment place identical positions of the sequences on top of each other. Depending on the application, different additional operations are allowed: *Gaps* may be inserted into the sequences to account for bases that are missing in one of the sequences, or bases that are not identical are allowed to align with each other which is called a *mismatch*. By maximizing the number of matching positions and minimizing the number of gaps and mismatches (more precisely: optimizing a score calculated from all of the above operations and further context), one can get a representation of how the two sequences compare to each other like demonstrated in Figure 1.3. There are multiple approaches to solving the alignment problem, each with their own applications. Dynamic programming algorithms like the *Needleman-Wunsch*[66] or *Smith-Waterman*[80] algorithm for example can be used to find an optimal alignment of two sequences. However, both algorithms' running times scale with the product of the sequence lengths and the number of sequences. This makes them too slow for mapping the hundreds of millions of short reads to the 3.2 billion bp long reference genome. Today this is solved by applying mostly heuristic approaches. Established approaches like *Bowtie*[51;52] or *BWA*[57] use suffix arrays for indexing the reference genome and quickly finding positions of portions of the reads. These *seeds* are then extended to get the final mapping positions.

Once the reads are mapped to the reference genome their alignment representations and the information on how the two reads of a pair are placed relatively to each other can be used to gain insight into differences between the reference and the sequenced genome, i.e. detect variation. This process will be described in the next section (1.5) more closely.

It is important to keep in mind that the resulting alignments of the mapping process can contain errors. Therefore, in 2008, Li et al. introduced the concept of *mapping quality* along with their read mapping software MAQ[59]. The mapping quality serves as a measure of confidence that a read truly originates from the assigned position. This can become an important information when later deciding to trust the alignment of a specific read when further downstream analyses like the detection of variants are performed.

Mapped reads of a sequenced genome are typically stored in the *sequence alignment/map* (SAM) format or its binary representation, the *binary alignment/map* (BAM) format.

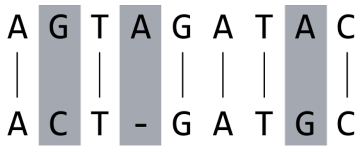

Figure 1.3: Example of a pairwise sequence alignment for the top sequence AGTAGATC and the bottom sequence ACTGATGC. Vertical lines indicates matches, while gray highlights mismatches and gaps that are further indicated by a dash in one of the sequences.

1.5 Basics of Variant Calling

Variant calling is the process of detecting differences between a sequenced genome and the reference genome. After the mapping of the NGS reads to the reference genome, each mapped read is represented as sequence alignment to the reference genome, thereby containing information on how the single positions of the read compare to the reference. If the genetic sequence of a sequenced genome differs from the sequence of the reference genome, for example due to the presence of an SNV, the mismatch will show in the alignments. Since reads can contain sequencing errors or the reads can be mapped incorrectly it is important that the examined position is not only covered by a single read but many[55]. The number of reads that cover on positions is called *coverage* or *read depth.* A typical average coverage for recent whole-genome sequencing (WGS) experiments is 30x coverage. The coverage also depends on the aim of the experiment[44;47]. In presence of real variation one expects to observe the deviation in the alignment for a substantial fraction of the reads that cover the examined position. Reads not containing the variant may originate from a non-variant haplotype in case of a heterozygous variant.

While SNVs can be detected from single short reads, SV detection requires additional signals that are encoded in the read pairs and their *orientation* relative to each other. NGS reads have an orientation on the DNA fragments they were derived from and therefore have an orientation or direction on the reference genome. This is due to the directionality of DNA molecules which are always synthesized from 5'-end to 3'-end. Some SVs can lead to the orientation of one or both reads of a pair being switched. SVs can further influence the distance between two reads of a pair that are mapped to the reference genome. A more detailed description of the signals that can be used to detect the different SV types is given in Section 2.1.

There are many ways one can detect small and structural variation from the mapped reads. Most modern methods involve a sophisticated statistical process that allows them to weight the probabilities that stem from the sequencing and alignment process against the probabilities of observing potential variants at different zygosity. An often applied approach is to calculate the posterior probability that an examined position is true variant in a Bayesian framework, like done by the well established tool *GATK* [63]:

$$p\left(G|D\right) = \frac{p\left(G\right)\prod_{b\in\mathcal{P}} p\left(b|G\right)}{p\left(D\right)} \tag{1.1}$$

with D representing the alignment data at the examined position and G indicating the Genotype. $p\left(G\right)$ is the prior probability of observing Genotype G, while b represents a single base from the *pileup* $\mathcal{P}$ at the inspected position. A pileup is the collection of all read bases covering the relevant position including information about their quality and how they are matching the reference. What priors are used and how the probabilities for the single bases and genotypes are calculated depends on the specific tool.

Genotyping describes the process of determining the genotype of one or more genomes for a set of variants. It is done by examining the reads in the region affected by the variant and comparing how many reads show evidence of the variant and how many reads contradict it, as show in Equation 1.1. If a variant is present in a homozygous state, virtually all reads are expected to show evidence of the variant (not considering sequencing and alignment errors). For heterozygous variants and under the assumption that the reads are sampled equally from both haplotypes one would expect ideally 50 % of the reads to show evidence of the variant. Of course, these are purely theoretical assumption, especially considering sequencing and alignment errors. Consequently, algorithms for genotyping have to apply various thresholds or apply statistical models for finding the most likely genotype. Genotyping is often performed by the variant caller, but special tools for genotyping of known variants in set of many genomes like *SVTyper*[14], or *GraphTyper*[22] also exist.

Another possibility for genotyping known variants in a genome are *k-mer* based approaches. A k-mer is a substring of length k. By dividing the read data into overlapping k-mers and comparing the counts of them to the k-mers one expects to observed in presence of absence of a specific variant, it is possible to infer the genotype of the variant in the sequenced genome without the need for performing the computationally intensive read mapping for the sequenced genome[79]. The tool *BayesTyper[79]* is one example for k-mer based genotyping.

Small variants can be detected and genotyped with very high reliability. A 2019 study by Chen et al.[12] reported F_1 scores (see Equation 4.6 in Subsection 4.1.2) consistently above 0.975 for all compared small variant calling pipelines (*GATK4*[63], *Strelka2[45]* and *Samtools-Varscan*[48;58]) on different data sets when evaluating SNV calls. For indels, the observed F_1 score was always above 0.85 for two of the three compared tools.

The detection of SVs can also be performed in a probabilistic framework. Many early SV callers were limited in their detection rates because they relied on only one type of signal[16]. By integrating multiple SV signals, modern SV callers like *Delly*[74], *Lumpy*[54] and *Manta*[13] can achieve higher accuracy.

Small and structural variants can also be detected in assembly approaches. The reads of the sequenced genome are assembled into longer contigs or whole genomes before comparing the assembly to the reference genome. This is especially useful for the detection of SVs[37] because it allows to detect the SVs directly from the alignment of the two genomes like one would do for small variants. A considerable drawback of this approach is that a full assembly requires very high coverage of the sequenced genome and a vast computational effort, making this approach unfit for general application. A locally limited assembly for the validation and characterization of a candidate SV is however often feasible and applied by some modern SV callers, e.g. Manta and GRIDSS[10].

Called variants are typically reported using the *variant call format* (VCF), which has been developed for the 1000 Genomes Project[18]. VCF is a tab separated format that describes each variant in a single line. It can further contain genome wise information regarding the presence of a variant in the last columns. Additional information tags can be defined in the header and reported in the INFO or FORMAT fields. An example of two variants represented in VCF is given in Figure 1.4. The first line describes a simple SNV in the 20th base of chromosome 1. The reference sequence contains a C as the 20th base of the sequence, while the variant genome contains an A. SAMPLE1 is heterozygous for this variant while SAMPLE2 does not carry the variant. The second lines shows how a SV can be described in VCF. Here, a deletion starts after position 132 on chromosome

21, which contains a G, and ends at position 232, effectively deleting the 100 bases in between. SAMPLE1 carries the SV on both haplotypes, making it a homozygous carrier, while SAMPLE2 is genotyped as a heterozygous carrier.

```
#CHROM   POS   ID   REF   ALT    QUAL   FILTER   INFO                FORMAT   SAMPLE1    SAMPLE2
1         20   .    C     T        42   PASS     .                   GT:GQ    0/1:141    0/0:255
21       132   .    G     <DEL>   .     PASS     SVTYPE=DEL;END=232  GT:GQ    1/1:255    0/1:100
```

Figure 1.4: Example of two variants represented in the variant call format. Lines starting with '#' indicate header lines. Each variant is described in the columns of one line: CHROM and POS hold the chromosome and position of the variant in the reference genome. The ID field can contain a unique identifier of the variant, for example a database accession number. The REF field contains the base(s) of the reference genome at the position of the variant and the ALT field contains the bases observed in the variant genome in presence of the variant. The QUAL field is used for reporting a confidence measure for the variant; higher QUAL values indicate a higher confidence. The FILTER field can contain information about filtered variants. The INFO field gives additional information, for example the variant type or the position where the variant ends. The FORMAT field is used for defining the fields of the subsequent columns: 'GT:GQ' indicates that the following columns, each containing information on one analyzed genome, contain the genotype before the colon and the genotype quality (a confidence measure for the correctness of the genotype) after the colon.

1.6 Population-Scale Genomics and Joint Variant Calling

NGS technologies have seen a steady decrease in costs and increased in throughput during the last years[37]. This development has made the whole-genome sequencing of increasingly large cohorts possible. Projects like *Genome of the Netherlands*[29] (GoNL), the *UK Biobank*[9] or endeavors by *deCODE genetics*[31], have sequenced thousands to hundreds of thousands of whole genomes of single populations and provided insight into demographics changes, rare alleles and genomic variation in general. Just at the end of 2021, the newest data of the UK Biobank has been released for researchers worldwide. It contains 5 PB of WGS data of 200,000 individuals across the United Kingdom, making it the largest single release of WGS data[84].

Population genomics are not limited to the analysis of single populations but often also aim at comparing the genomic makeup of two or more populations. For example, Mallick et al. sequenced 300 genomes from 142 diverse populations in the *Simons Genome Diversity Project* (SGDP)[62]. They provided insights into human ancestry and evolution and created a catalog of 34.4 million SNVs and 2.1 million indels.

Especially for the detection of rare variants with an allele frequency below 1 % in the general population, it is important to sequence many individuals because otherwise many rare alleles will not be carried by any of the sequenced individuals. The chances of observing variants of special interest can be enhanced by focusing the study on a cohort of individuals that are affected by a specific phenotype or disease, like done for asthma[61] or autism[89].

By using the catalogs of variants gained from such studies and correlating them to phenotype information in *genome-wide association studies* (GWAS), it is possible to learn what genes and variants are associated with diseases and possibly even find new therapeutic targets[89].

Joint variant calling describes the process of detecting variants from multiple genomes together. The variant calls are generated by integrating information across all genomes of the cohort and all genomes are genotyped for this set of jointly determined variants. This comes with the benefit of increased sensitivity. For example, assume two genomes that both lack sufficient evidence for the same variant such that no reliable variant call can be made when considering the genomes individually. When considering both genomes together, the combined evidence can be enough to be confident about the variant call at that position. Because of this, joint calling is a desirable approach for the analysis of population-scale data. It also eliminates the need to merge variant calls into single records after they have been made on a per genome basis, which can be ambiguous for SVs and also result in missing genotypes for genomes that were not genotyped at the respective location.

For small variation, approaches for the joint analysis of cohorts at population-scale have already been established. One well-known example is the *GATK HaplotypeCaller*[70]. Although its workflow does not constitute a joint calling in the classical sense because it still performs the initial detection of variants on a per genome basis, it allows the efficient joint genotyping for small variants in large cohorts. By analyzing small variants in all genomes of a cohort together, the accuracy can be increased. The approach of the GATK HaplotypeCaller is explained in more detail in Section 2.2.

Joint calling for the detection of SVs in hundreds of genomes has also been shown to work successfully[32;82]. For larger cohorts however, joint calling is still challenged by a lack of scalability and huge file input/output (I/O) loads[30]. This is why most pipelines rely on a sample-wise detection phase, followed by a merging of the variants and a sample-wise regenotyping step against the list of all merged variants[1;39;53]. The approaches of some established SV callers are presented in Section 2.3.

SVs are often detected with diverging positional estimates in different samples, making the merging process reliant on arbitrary size and distance thresholds[64;90]. A scalable joint calling approach would therefore not only provide an improved accuracy, but also a simpler pipeline that eliminates the uncertainty of the merging step.

1.7 Thesis Outline

This thesis contributes to the field of structural variant detection, joint calling and population-scale genomics in different aspects.

Firstly, it presents a new joint calling approach for SVs called *PopDel* (Population-wide Deletion calling). PopDel has been shown to scale to tens of thousands of human WGS samples and can reliably detect as well as genotype SVs. Originally developed for the detection and genotyping of deletions in population-scale data, its approach has been generalized and prototypes for inversions and tandem duplications have been implemented.

The thesis also introduces a new file format for the sparse representation of alignment data, the *PopD*el *read pair profile*. The PopDel profiles are binary representations of the most relevant alignment information for PopDel's downstream SV calling. The information is extracted from the original alignment files and are used by PopDel for the preprocessing of the data and reduction of the I/O load during the joint calling.

The benchmarks performed for the evaluation of PopDel is the third contribution of this thesis. It compares PopDel and the established SV callers Delly, GRIDSS, Lumpy and Manta on different simulated and real data sets with up to 1,000 simulated or 150 real samples to draw a comprehensive picture of the performance of the different approaches. The benchmark focuses on the scaling to many samples and underlines the benefits of joint calling. It further provides approaches as well as variant call sets for future research and benchmarking efforts.

A fourth point is the contribution to the detection of a novel deletion variant in the gene encoding for the low-density lipoprotein receptor[8]. The deletion was detected by applying PopDel on WGS data of 43,202 Icelanders. The shown correlation of the variant with a lifelong reduction of low-density cholesterol blood levels makes it a finding of high medical interest for future research concerning cardiovascular diseases. This finding was made in cooperation with deCODE genetics.

The remainder of the current chapter presents and discusses all publications that have arisen from this thesis or cooperation while working on topics related to this thesis. The aforementioned finding of the novel deletion in the low-density lipoprotein receptor gene is also presented here.

Chapter 2 introduces the technical background for the detection of SVs from NGS data. It further discusses the current methods for the joint calling of short and structural variants.

Chapter 3 describes the PopDel read pair profiles and the complete approach applied in PopDel for the joint detection and genotyping of SVs.

Chapter 4 presents the setup of the benchmarks and the results for all compared SV callers on the different data sets.

Finally, Chapter 5 concludes this thesis with a summary of all results and contributions, as well as a discussion of potentials and a future outlook.

1.8 Publications

The work on this thesis and in the field of structural variant calling has resulted in contributions and (co-)authorship in the following publications. A brief description of the publications' contents and my personal contributions will be presented in the subsequent subsections.

Bjornsson, E., Gunnarsdottir, K., Halldorsson, G. H., Sigurdsson, A., Arnadottir, G. A., Jonsson, H., Olafsdottir, E. F., **Niehus, S.**, Kehr, B., Sveinbjörnsson, G., et al. (2021). Lifelong reduction in LDL (low-density lipoprotein) cholesterol due to a gain-of-function mutation in LDLR. Circulation: Genomic and Precision Medicine, 14(1):e003029.[8]

Krannich, T., White, W. T. J., **Niehus, S.**, Holley, G., Halldorsson, B., and Kehr, B. (2022). Population-scale detection of non-reference sequence variants using colored de Bruijn graphs. Bioinformatics, 38(3):604-611.[50]

Niehus, S., Jónsson, H., Schönberger, J., Björnsson, E., Beyter, D., Eggertsson, H. P., Sulem, P., Stefánsson, K., Halldórsson, B. V., and Kehr, B. (2021). PopDel identifies medium-size deletions simultaneously in tens of thousands of genomes. Nature communications, 12(1):110.[67]

Sarwal, V., **Niehus, S.**, Ayyala, R., Chang, S., Lu, A., Darci-Maher, N., Littman, R., Chhugani, K., Soylev, A., Comarova, Z., et al. (2020). A comprehensive benchmarking of WGS-based structural variant callers. bioRxiv.[77]. Accepted for publication in Briefings in Bioinformatics.

Lifelong reduction in LDL (low-density lipoprotein) cholesterol due to a gain-of-function mutation in LDLR. The publication reports and discusses the discovery of the first known gain-of-function variant in the 3'-UTR of the low-density lipoprotein receptor (LDLR) gene causing a reduction of mean blood low-density lipoprotein (LDL) cholesterol levels. High levels of blood cholesterol have previously been linked to a variety of cardiovascular diseases, making the finding a variant of potential medical interest. The discovered deletion of 2.5 kb *(Del2.5)* could be linked to a significant (74 %) reduction of mean blood LDL cholesterol levels in carriers (Figure 1.5 C). Further analyses strongly suggest a causal relation. The deletion was first discovered in the WGS data of three closely related individuals (mother I.2, son II.4, grandson III.4 in Figure 1.5 B) in a cohort of 43,202 Icelanders by applying an early version of PopDel in batches of 10,000 individuals to specifically explore the genomic vicinity of the LDLR gene.

Since the authors of the study were interested in variants affecting the LDLR gene, first the PopDel read pair profiles (see Section 3.2.4) were created for the LDLR gene including 5000 bp of flanking sequence to both sides of the gene for all 43,202 WGS samples. Then *PopDel call* was applied on batches of 10,000 profiles, which at that time this was the upper limit for PopDel's joint calling. After processing each batch for the first time two carriers were detected in two of the batches. The carriers were added to the other batches as "seeds" to increase PopDel's sensitivity for *del2.5* (see Section 3.3.3 for details on the increased sensitivity). The calling was then repeated, yielding an additional carrier and thereby showing the benefit of the joint calling. The limit of 10,000 profiles no longer applies to the newest version of PopDel, making this batch-and-seeding procedure unnecessary. *Del2.5* was confirmed using polymerase chain reaction (PCR) and subsequent Sanger Sequencing. Following this discovery and the observation that all carriers are closely related, additional members of the family were WGS sequenced and phenotyped. This lead to the detection of the remaining four known carriers of *del2.5*. Further inspection of *del2.5* granted insight into the likely mechanism by which it leads to the over expression of LDL receptors, ultimately reducing the mean blood LDL levels: *Del2.5* increases the mRNA stability by removing known miRNA target sites in the 3'-UTR of the transcript. This increases the transcript's resistance to negative regulation. Consequentially, larger quantities of the LDLR mRNA are translated into proteins, causing a higher number of LDL receptors to be expressed on the cell surfaces. The increase in LDLR receptors increases the absorption of LDL from the blood (see Figure 1.6).

My specific contributions to this finding and publication were as follows: I implemented PopDel, which was at an early and non-publicly available stage at the time of the finding, in coordination with the other authors of the study. I provided the parameters for the runs of PopDel and assisted in the interpretation of the output, which was at that time not yet formatted in standard VCF. I further was involved in the wording of the manuscript parts concerning PopDel and the detection of *del2.5*.

This study is an especially vivid example of a good use-case for PopDel for various reasons: Firstly, none of the other applied tools for SV detection were capable of detecting *del2.5*. Additionally, PopDel's design allowed the efficient targeted exploration of the vicinity of the LDLR gene without the need for processing the whole BAM files of all 43,202 samples or subsetting them. The success of the batch-and-seed approach for detecting additional carriers of *del2.5* further served as a proof-of-concept of the benefits of the joint calling approach in real world scenario.

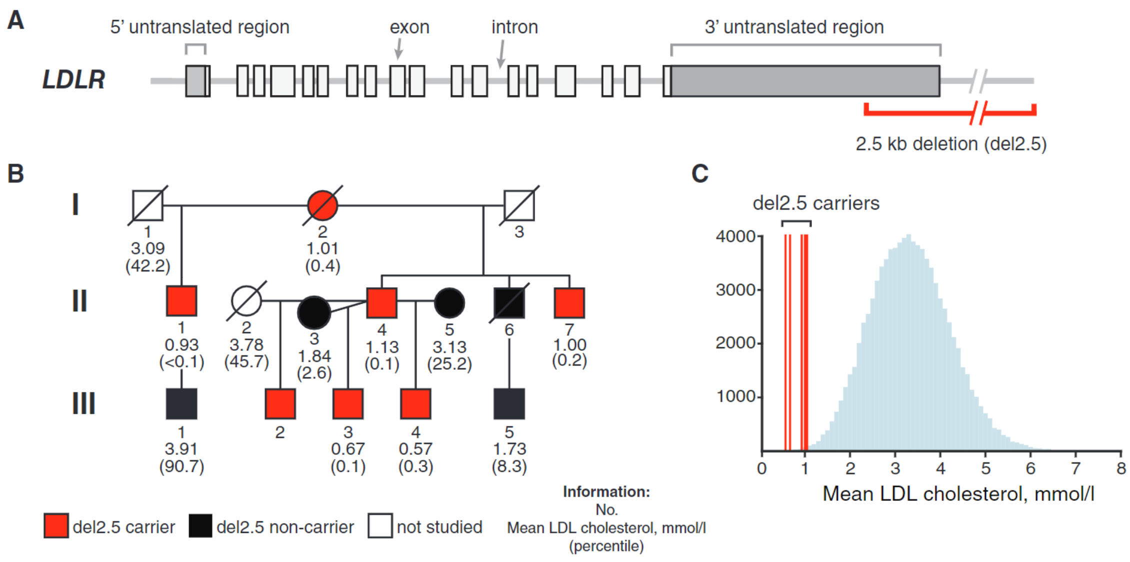

Figure 1.5: *Del2.5* causes low blood LDL cholesterol levels. **A**: Schematic of the genomic region of the LDLR gene and *del2.5*. ***B*:** Pedigree of the family that encompasses all detected carriers of del2.5. Genotyped individuals are marked as red (carriers) and black (non-carriers). The numbers below the individuals indicate the number in the respective generation, the mean LDL cholesterol level [mmol/l] and the age and sex adjusted percentile of the individual's LDL cholesterol level in the general Icelandic population. **C**: Distribution of mean LDL cholesterol levels measured in 101,857 Icelanders. Mean values for *del2.5* carriers are indicated by red lines. Figure taken from Bjornson et al. (2021)[8]

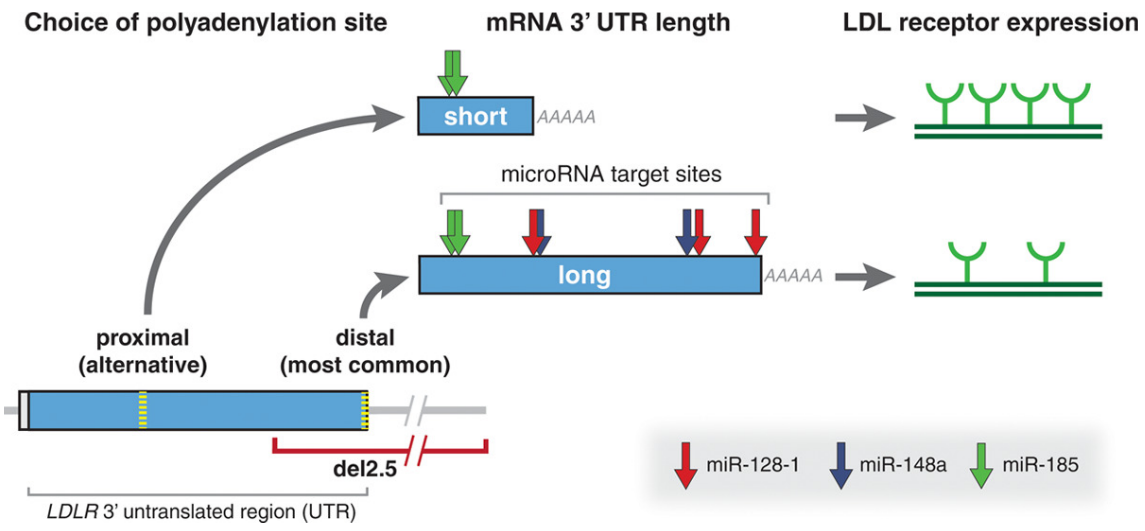

Figure 1.6: Proposed mechanism for the increased LDLR expression caused by *del2.5*. Figure taken from Bjornson et al. (2021)[8].

Population-scale detection of non-reference sequence variants using colored de Bruijn graphs. This publication presents *PopIns2,* a program for the detection and genotyping of non-reference sequences (NRS) in population-scale WGS data. PopIns2 is the successor of the tool PopIns[42] and introduces a new method for merging contig assemblies of unaligned read sequences from thousands of genomes by using colored de Bruin graphs (CDBG). As already discussed in 1.2 and 1.5, NRS, which typically manifest themselves as insertions with respect to the reference sequence, are an interesting subtype of SVs. Since their underlying sequence is missing from the reference NRS cannot be detected and resolved solely by the information of reads aligned to the reference. They rather require an assembly of the unaligned sequences, making the task more computationally demanding than the detection of other types of SVs. Therefore, most previously published tools, including PopIns, either excluded the detection of insertions from their analyses or exhibited limited scalability to bigger sample numbers.

PopIns2 uses the read sequences from the provided BAM files. It creates a sample wise contig assembly of those reads that do not (or poorly) align to the reference. In the subsequent merging step, the sample-wise contigs are merged into a set of unified NRSs. This is done by constructing a CDBG of the contigs. The colors of the vertices identify the samples. A path through the graph is generated by heuristically solving the *minimum path cover* problem. This new formulation of the problem and application of CDBGs not only improves the running time of the merging step compared to the original implementation of PopIns (50 minutes instead of 94 minutes for contigs generated from 150 WGS samples) but also drastically reduces the memory consumption by three orders of magnitude (342 MB instead of almost 100 GB for 150 WGS samples). Consequentially these improvements allow for many more samples to be processed together before reaching hardware restrictions. The final steps of the workflow, the placement of the unified NRSs on the reference genome and genotyping of the sequenced genomes for the candidate variants, underwent no algorithmic changes compared to the original implementation of PopIns.

To assess the performance of PopIns2 it was compared against PopIns and the NRS detection tool *Pamir*[40] on simulated and real human WGS data sets. On up to 100 simulated WGS samples, PopIns2 shows clear improvements over PopIns and its precision and recall are competitive with *Pamir's.* Benchmarking on real data was performed using the *Polaris HiSeqX Diversity Cohort* (BioProject accession PRJEB20654) (PDC) and the *Polaris HiSeqX Kids Cohort* (BioProject accession PRJEB25009) (PKC). The PDC comprises 150 WGS samples from three continental groups (50 African, 50 East Asian, 50 European samples), allowing to check for biologically meaningful NRS findings by performing a principal component analysis (PCA) on the NRS detected by PopIns2 and checking if the samples cluster by their known ethnicity (Figure 1.7). The steps' details for the processing of the variants and calculation of the PCA were the same as for the evaluation of PopDel and are described in Subsection 4.7.2. By expanding the samples of the PDC with the PKC, a set of 47 trio-families was be generated, enabling the assessment of the rate of estimated genotype combinations that violate Mendelian laws, and the computation of the rate at which NRS variants were transmitted from parents to their offspring. A detailed description of the metrics and their calculation is given in Subsection 4.1.3. As demonstrated in Figure 1.8, both implementations of PopIns2 produce largely consistent genotypes for the detected NRS variants and meaningful genotype likelihoods. Filtering by genotype likelihood can be used to achieve Mendelian inheritance error rates below 1%. The transmission rate is shown to be close to the expected value of 50%.

My specific contributions to the publication are the calculation of the PCA on the NRS variants detected by PopIns2 on the 150 samples of the PDC, as well as the calculation of the Mendelian inheritance error rates and transmission rates, including the filtering of variants not in *Hardy-Weinberg-Equilibrium* (HWE, see Equation 4.9 in Subsection 4.1.3), of the NRS variants detected in the 47 trios of the PKC. I further assisted in the interpretation of the results of those calculations and the wording of the manuscript parts concerning those analyses.

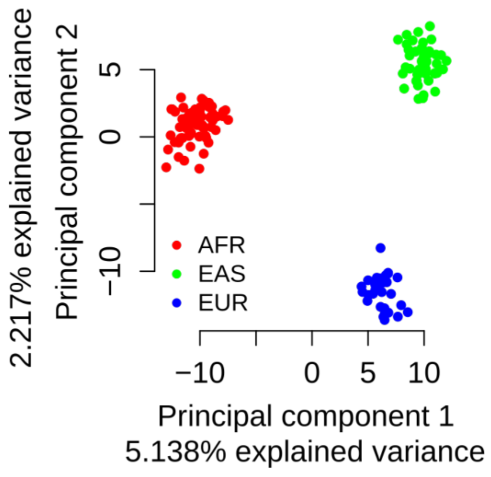

Figure 1.7: PCA on 1,787 NRS variants detected by PopIns2 on 150 WGS samples of the Polaris Diversity Cohort. Each data point indicates a sample, colored by its reported continental superpopulation (AFR, African; EAS, East Asian; EUR, European). Figure taken from Krannich et al. (2022)[50].

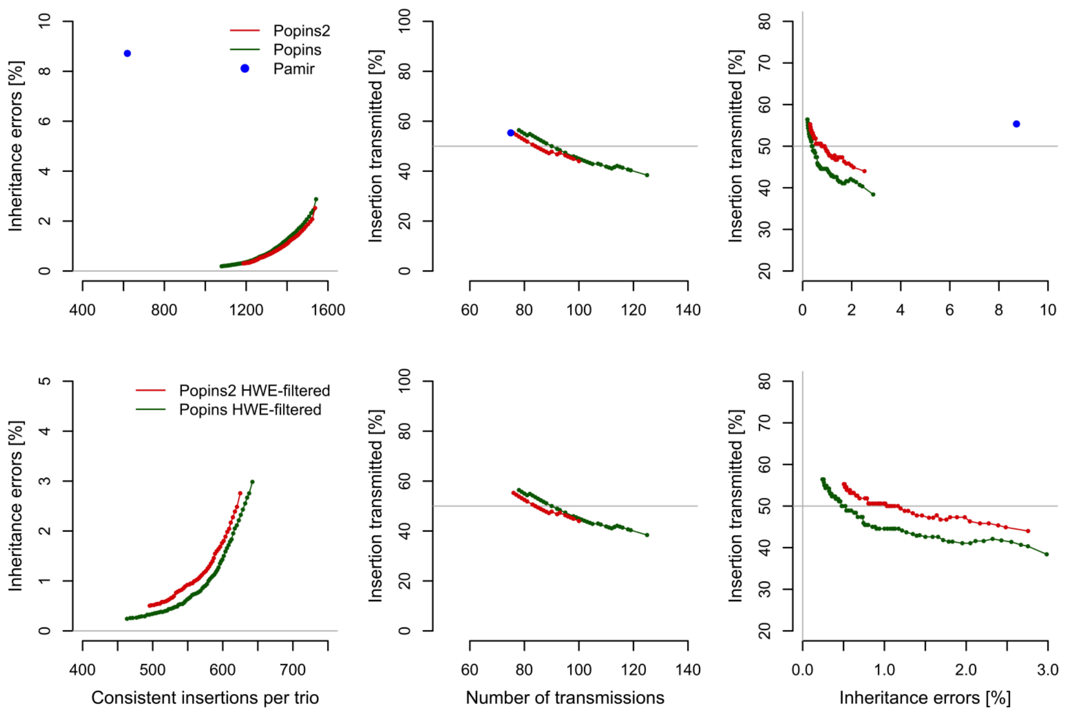

Figure 1.8: Mendelian inheritance error rate and transmission rate of NRSs found in 47 trios of the Polaris Diversity Cohort. Top: NRSs were not filtered by adherence to the Hardy-Weinberg-Equilibrium (HWE). Bottom: Calls were filtered for adherence to HWE (p-value 0.01). Grey lines indicate the theoretically ideal values. As Pamir does not include genotype likelihoods in its calls which were necessary for filtering the numbers of NRSs, only the median inheritance error rate or transmission rate across the 47 trios was included in the plots as a single data point. Since Pamir could not process all 47 trios jointly, no HWE filter was applied on its output. Figure taken from Krannich et al. (2022)[50].

PopDel identifies medium-size deletions simultaneously in tens of thousands of genomes. This publication presents the tool PopDel, details of the implemented methodology and an extensive benchmark on simulated as well as various real data sets to compare PopDel with other state-of-the-art SV callers. Since most contents of the publication are also presented and discussed in depth in Chapters 3 and 4 of this thesis, this publication will only be summarized briefly here. For further details, please consult the corresponding chapters. The publication contains the description of PopDel's algorithmic approach for the detection and genotyping of deletions and compares it with the SV callers Delly, GRIDSS, Lumpy and Manta on 1,000 simulated WGS samples of the human chromosome 21, 150 human WGS samples of the Polaris Diversity Cohort, 47 family trios of the Polaris Kids Cohort, the *Genome in a Bottle* sample HG002 plus its parents and the *Illumina Platinum Genome* HG001 (aka NA12878).

Evaluation on the simulated data included running time, memory consumption as well as precision and recall for increasing numbers of samples in the joint calling process and evaluation of the positional accuracy. Owing to the availability of high quality reference truth sets for the well studied genomes HG001 and HG002, precision and recall could also be calculated on real data and the agreement of the variant callers could be examined. For a discussion on the limitations of precision and recall on real data please see Subsection 4.4.4. The inclusion of the parental genomes of HG002 in a joint family calling showed the benefits of PopDel's joint calling approach by exhibiting increased sensitivity without loss of precision. The Polaris Cohorts were used to compare the quality of the estimated genotypes by assessing the rate at which they violated Mendelian laws and how many parental alleles were transmitted from the parents to their offspring. Further, the cohorts enabled the examination on how the detected variants reflect population structure known from literature. A *de novo* deletion detected by PopDel in one of the 47 Polaris trios is also presented and shows that PopDel is capable of detecting variants at a very low allele fraction or exclusive to one sample when analyzing many samples at once. The successful application of PopDel on WGS data of 49,962 Icelanders underlines its scalability to large cohorts.

As this work constitutes the central publication of this thesis, all parts of this paper were mainly conceived and authored by me.

A comprehensive benchmarking of WGS-based structural variant callers. This publication presents a deletion detection benchmark of 16 different SV callers. A preprint of the study has been published on bioRxiv. The study has been accepted for publication in the journal Briefings in Bioinformatics. The benchmark is based on a PCR confirmed set of deletions in seven inbred homozygous mouse strains and comparisons on the human Genome in a Bottle sample HG002 and the available high quality reference truth set. Please note that the benchmarking on human data is not included in the preprint. The number of detected variants per mouse strain is measured for each SV caller and performance metrics like precision and recall are calculated. The calls are generated for varying degrees of downsampling of the coverage of the BAM files to assess the tools' performance for both high and low coverage data. Further, a comparison of the estimated deletion lengths is performed to show to what degree the callers over- or underestimate the true size of a deletion. Besides the large number of compared SV callers, a strong point of the study is the use of real biological data for which all deletions are known and confirmed. This allows the assessment of the callers' precision and recall under more realistic conditions than the usual comparison on simulated data, which cannot fully reflect the properties of real sequencing data. A limitation of the approach is the lack of heterozygous variants and variants other than deletions in the benchmark. Further, the mouse genome is less repetitive than the human genome, reducing the difficulty of the variant detection task on mouse data.

My specific contributions to the publication consisted of setting up and performing the variant calls using the tools GRIDSS, Lumpy, Manta and PopDel on the mouse and human data. I assisted with the design of evaluation metrics, the selection of the human reference and evaluation of the variant calls on the human data. Further, I was involved in the wording of different parts of the manuscript.

Chapter 2

Background

This chapter aims at providing the necessary background about the detection of SVs from alignments of NGS data. Additionally, current methods for the joint analysis of small variants will be described to relate them to the approach applied by PopDel. A discussion of the current state of SV detection as performed by some SV callers other than PopDel will conclude this chapter.

2.1 Structural Variation in Detail

All SVs cause a disruption of the original sequence. Those points of disruption are called *breakpoints*. The positions where two previously separated substrings of a sequence form a new junction due to an SV are called *novel junctions.* Depending on the type of SV the number of breakpoints and novel junctions can vary.

SVs exhibit different signals or signatures in sequencing data around their breakpoints and the genomic regions in between. Different sequencing technologies provide different signatures, but this thesis focuses on the signatures that are exhibited by short read paired-end sequencing data. SV signatures from paired-end reads can be divided into three classes: *Read depth*, *discordant read pairs*, and *split read alignments*.

Read depth measures the number of reads that cover a specific genomic position or genomic window. SVs that cause a gain or loss of sequence, namely deletions, duplications and unbalanced translocations, also cause a change of the read depth. Those SVs can be summarized under the term *copy number variants* (CNVs). Even in absence of CNVs the read depth is naturally fluctuating. One factor that can cause a substantial fluctuations of the read depth is the *GC-content*. It describes the relative quantity of the bases G and C as compared to A and T in a piece of sequence. Simple corrections for the GC-bias exist and are necessary to detect deletions from read depth changes alone[7]. The read depth signal alone is only suitable for the detection of larger CNVs due to difficulties in the exact determination of the breakpoints[3].

Split read alignments can occur if not a pair but a single read of a pair is spanning the breakpoint of an SV, like displayed in Figure 2.1a for read pair s. One part of the read spanning the breakpoint cannot be mapped to the reference in one go. This part is therefore *soft-clipped*, meaning the affected positions will be marked as mismatches in the alignment, indicated by the light gray coloring in the figure. Some SV callers or mapping programs perform a split read alignment, which will try to find the best position for the soft clipped portion of the read. If the clipped portion is long enough and its sequence unique, it can be assigned the correct position. Split read alignments are typically exceptionally useful for the exact definition of the breakpoints of SVs.

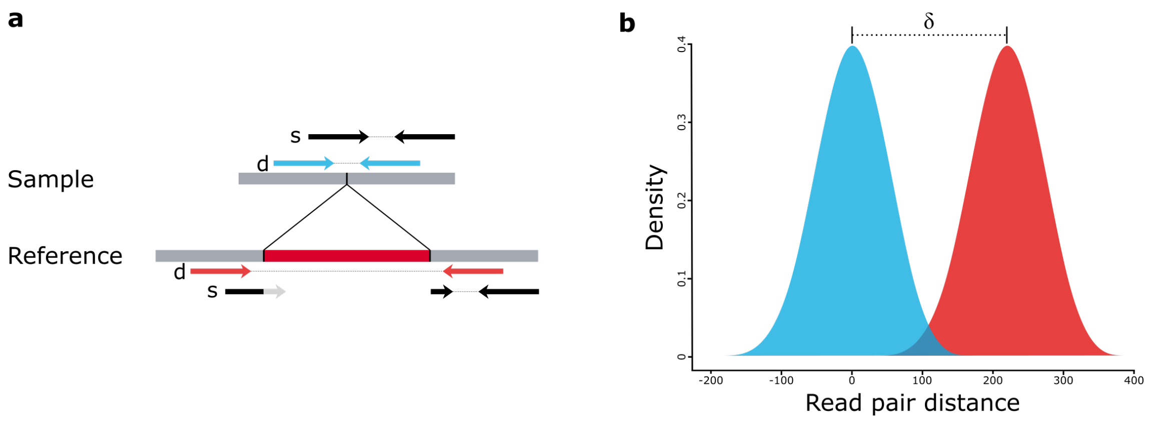

Figure 2.1: Read pair signatures of deletions. **a** A deletion (marked in red) causes an increase of the read pair distance for read pair d spanning the breakpoint. Further, a split read alignment for the first read in read pair s occurs. The read pair distance of read pair s is counted from the clipped end upstream of the deletion up to the 3'-end of the read that is located downstream of the deletion. Therefore, it is also increased. **b** Read pairs that span a deletion breakpoint follow a shifted distribution of read pair distances (red). The shift from the usual distribution (blue) corresponds to the deletion size δ.

Discordant read pairs are read pairs whose mapping distances or orientations do not match the expected values. The distance between the two reads in a pair is expected to follow a set distribution, depending on the distribution of the fragments selected during the preparation of the sequencing sample (see Figure 2.1b, blue curve). If a read pair overlaps an SV breakpoint, i.e. its first read is mapping to the left of an SV breakpoint and its second read is mapping to the right of the breakpoint, the distance between the read pairs will change depending on the size of the SV. Throughout this thesis, this *read pair distance* will be defined as the absolute distance in bp on the reference between the mapped 3'-ends of both reads in pair. It is closely related to the insert size (see Section 1.3), which also increases if a read pair overlaps an SV breakpoint. Neither read pair distance nor insert size are not defined for reads that map to different chromosomes. In soft clipped reads in forward orientation the rightmost unclipped base is considered as its 3'-end. Reads in reverse orientation with soft clipping are treated as if their 3'-end was located at their leftmost unclipped base. Read pair d in Figure 2.1a shows an example of an increased read pair distance.

A read pair is also discordant if the orientation of its reads in the reference genome is different from the expected forward-reverse orientation (FR). Duplications, inversions and some translocations cause a switch in the orientation of reads because they either influence the direction of sequence (inversions, some translocations) or the mapping order of the reads in a pair (duplications). Figure 2.5 shows an example for this process.

The frequency at which SVs occur in the human genome heavily depends on their type. Collins et al. created *gnomAD-SV*, a public database of SVs[16]. They report frequencies of different SV types found in 14,216 human genomes of different origin. A selection of the resulting distributions is shown in Figure 2.2. Generally, the frequencies of SVs declines with increasing size. Notable local peaks in the frequency distributions of CNVs are often caused by mobile elements, like *Alu* and *S*VA or *LINE1*. The frequencies reported in gnomAD-SV for the specific SV types will be given below in the respective subsections. The frequencies are to be interpreted as lower boundaries because it is to be expected that there are more SVs than those that are currently reliably detected.

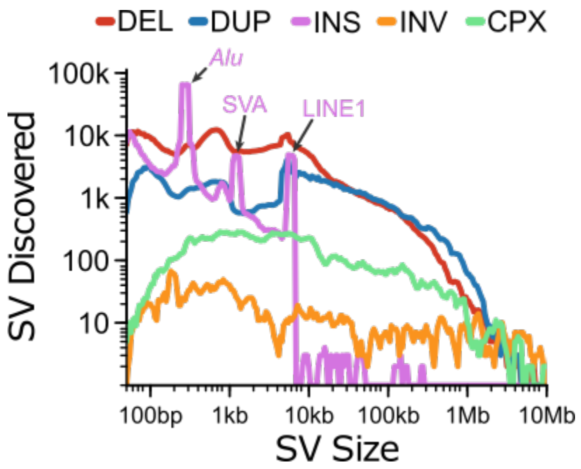

Figure 2.2: Frequencies of SVs types detected in 14,216 diverse samples in gnomAD-SV by SV size. DEL, deletion; DUP, duplication, INS, insertion; INV, inversion; CPX, complex SV. Labeled arrows indicate peaks associated with mobile element insertions. Figure adapted from Collins et al. (2019)[16].

Mechanisms of SV Formation There are different models that can explain the biological generation of SVs, some of which will be presented here.

One mechanism that often leads to *recurrent* SVs is *non-allelic homologous recombination* (NAHR)[11;25]. An SV is recurrent if it can be found in multiple unrelated individuals and therefore has not been inherited but emerged individually in different individuals[11]. NAHR can occur when long (above 10,000 bp) pieces of sequence are very similar to each other (about 95 % homology) and are relatively close by, typically between 50,000 to 1,000,000 bp. During mitosis or meiosis the two highly identical sequence pieces can align with each other instead of their correct haplotypes. This leads to a *crossing over* and causes a deletion of the sequence parts between the parts of highly identical sequence. NAHR between the strands of different chromosomes will further produce a duplication that accompanies the deletion[11]. A schematic of this process is shown in Figure 2.3.

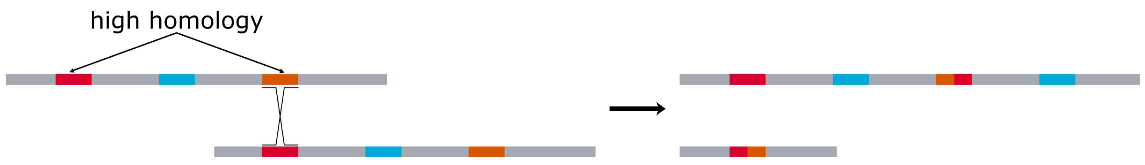

Figure 2.3: Schematic of an Non-allelic homologous recombination. The orange and red sequence parts share a high amount of homology, causing them to be aligned during mitosis or meiosis. The resulting crossing over causes sequence loss on one haplotype and sequence gain in the other haplotype.

Another mechanism is *non-homologous end joining* (NHEJ)[11;25]. NHEJ has been found as the cause of non-recurrent SVs, creating short insertions or deletions. NHEJ is a repair mechanism for double strand breaks of the DNA. When the two strand of the DNA double helix break at marginally shifted positions, the overhanging ends of the strands are removed and the new blunt ends of the double helix are jointed again to repair the break. This leads to the loss of the sequence of the overhanging ends. Alternatively, random nucleotides can be inserted at the position of the break[11]. Although NHEJ is the dominant repair mechanism for double strand breaks in human cells[2], it is not considered a major source of variation[27].

Microhomology-mediated break-induced replication (MMBIR) can occur when the replication fork of the DNA stalls or collapses during DNA synthesis[11;25]. The stalling can be caused by single strand breaks in the template DNA strand. Under ideal conditions, this would be covered by the *homologous recombination* mechanism, which repairs the break exactly[2]. But in the presence of microhomologies in the affected sequence the DNA polymerase can switch to another DNA strand and continue the replication with the wrong template[11;25]. This *template switching* can result in complex SVs of up to multiple megabases (millions of bases) in size[25]. The resulting SVs can include deletions, duplications, inversions and translocations[11;25;91]. Figure 2.4 shows an example of an MMBIR event leading to the formation of a complex SV.

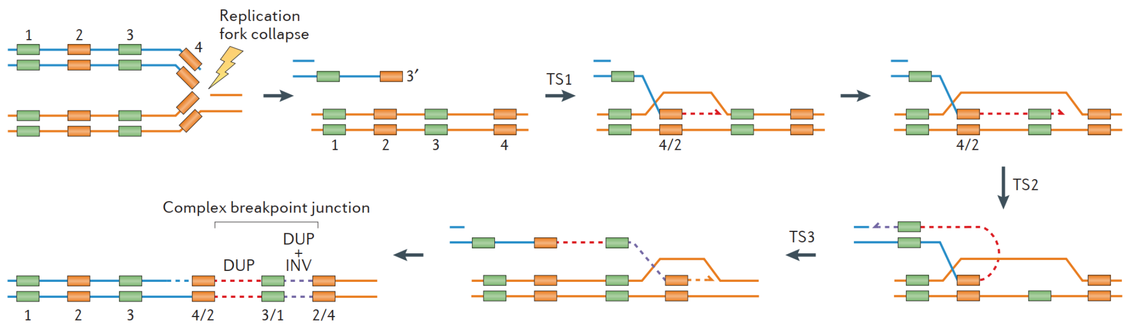

Figure 2.4: Complex SV caused by MMBIR. A single strand break causes the collapse of the replication fork. The nascent strand is trimmed back and the first template switch (TS) occurs when the homologous regions (indicated by the color of the segments) 4 of the nascent strand and 2 of the template strand are annealed. The polymerase continues the DNA synthesis until TS2 occurs: The newly synthesized block 3 of the nascent strand anneals to its duplicate on the nascent strand. This causes the synthesis of an inverted copy of the sequence upstream of block 3. During TS3 block 2 of the nascent strand anneals to block 4 of the template. The product of this process contains duplicated sequence between block 2 and 3 (including block 3), an inverted copy of the sequence after block 3 and an extra copy of block 4. Figure taken from Carvalho et al. (2016)[11].

2.1.1 Deletions

In the context of this thesis, a deletion is defined as a piece of consecutive sequence of at least 50 bp length that is missing from a sample sequence when compared to a reference sequence. Figure 1.2a shows a conceptual example of a deletion.

Deletions are the most commonly detected SVs in the human genome. The gnomAD-SV database reports a median of 3,505 deletions per human genome[16]. As shown in Figure 2.2, they follow the general trend of being rarer with increasing size. Local frequency peaks can be observed at 300 bp due to Alu insertions in the reference genome and at 6000 bp due to LINE1 insertions in the reference genome.

Deletions have been associated with a wide range of disorders in various studies. Examples are autism[89], rheumatoid arthritis[83] and muscular dystrophy[43].

Detection. One signal that can be used for the detection of (large) deletions is reduced read depth. Since the sequenced genome is lacking sequence on one or both haplotypes at the locus of the deletion the number of reads that are sampled from this position of the sample genome is reduced. This causes a reduced read depth at the affected location when mapping those reads to the reference.

Another deletion signature can be found in discordant paired end reads. A read pair overlapping the deletion breakpoint will exhibit a read pair distance that is increased by the size of the deletion, as shown in Figure 2.1a for read pair d. This causes a shift of the expected distribution of read pair distances by the deletion size (see Figure 2.1 b). The read pair distance is the main signal for the detection of deletions in PopDel.

If not a pair but a single read of a pair is spanning the breakpoint, like displayed in Figure 2.1a for read pair s, a split read mapping can occur. A split read mapping is a very clear signal for the presence of a deletion because it gives the exact size and position of the deletion, assuming the alignments of both part of the split read are correct. PopDel does not perform a split read alignment but uses the information of the soft clipped bases at the 3'-ends of the reads for the calculation of the exact breakpoints.

2.1.2 (Tandem) Duplications

In the context of this thesis a duplication is defined as a piece of consecutive sequence of at least 50 bp length that has been duplicated in the sample sequence when compared to a reference sequence. If the additional copy of the sequence directly follows the original sequence, this is called a *tandem* duplication. If the additional copy is inserted somewhere else in the genome, it is called a *dispersed* or *interspersed* duplication. The inserted copy of a duplication can also be detected as a insertion but does not consist of novel sequence. Figure 2.5 shows a conceptual example of tandem duplication and a dispersed duplication.

The gnomAD-SV database reports a median of 723 duplications per human genome[16]. They follow the general trend of being rarer with increasing length and have a local peak in frequency for 6000 bp due to LINE1 insertions (see Figure 2.2).

Duplications are associated with different disorders. For example, a duplication of the MeCP2 gene is causal for the *MeCP2 duplication syndrome*[73]. It is associated with weak muscle tone, autism and other neuropsychiatric disorders.

Detection. Like other CNVs, duplications can be detected by local changes of the read depth.

Paired end reads provide different signals suitable for the detection of tandem duplications. Read pairs overlapping the novel junction between the two copies of the sequence (read pair d in Figure 2.5a) will become discordant and exhibit a reverse-forward (RF) orientation when mapped to the reference. Further, their read pair distance will be increased depending on the length of the duplicated sequence. PopDel considers both the change in orientation and increased read pair distance when detecting duplications.

Split read mappings occur when a single read is overlapping the novel junction between the two sequence copies (read pair s in Figure 2.5a). As the primary alignment (here: its larger part) of the second reads follows the first read, the orientation of the read pair changes to RF. Further, the mapping positions of the split read alignment provide a sharp signal defining the breakpoints of the duplication. PopDel does not perform a split read alignment but uses the information of the soft clipped bases at the 3'-ends of the reads for the calculation of the exact breakpoint positions.

If the duplication does not occur in tandem, the paired end mappings can exhibit a signature similar to that of deletions. Pairs overlapping the first junction between the non-duplicated part of the sample sequence and the duplicated sequence, will show a read pair distance that is increased by the distance between the original and inserted sequence (read pair d in Figure 2.5b). If a read pair is overlapping the second junction of the duplicated and the non-duplicated sequence instead, the read pairs will further map to the reference sequence in RF orientation (read pair d' in Figure 2.5b).

Split read mappings can also occur for dispersed duplications, as shown for read pair s in Figure 2.5b. Here only the second junction of the duplicated sequence is sharply defined by the split read alignment. The first junction would require an additional read pair with one read overlapping the first junction.

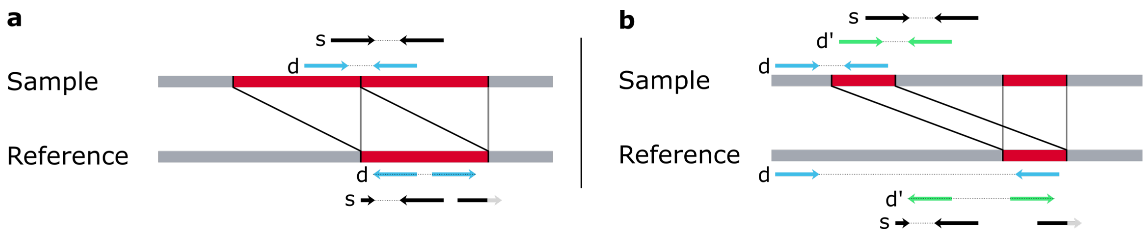

Figure 2.5: Read pair signatures of duplications.**a** A tandem duplication (marked in red) causes an increase in read pair distance and read pair orientation for read pair d. Further, a split read alignment occurs for the first read of read pair s. **b** An interspersed duplication causes an increased read pair distance for read pair d and an additional change in orientation for read pair d'. Further, a split read alignment occurs for the first read in read pair s.

2.1.3 Insertions

In the context of this thesis an insertion is defined as a piece of consecutive sequence of at least 50 bp length that is present in the sample sequence but absent from the reference sequence at that location. If the inserted sequence is not part of the whole reference, this is called a *non-reference sequence* (NRS) insertion or *novel sequence insertion*.

The gnomAD-SV database reports a median of 2,612 insertions per human genome[16]. As shown in Figure 2.2 many of those insertions occur as mobile element insertions. The majority of insertions are 6000 bp in length or shorter. The presence of many non-repetitive human NRS in primate genomes[41] suggest that those presumed insertions are actually sequences that have been deleted in the genomes that contributed to the reference genome.

Novel sequence insertions have been associated with coronary artery disease, myocardial infarction, variations in bone mineral density and breast cancer[41].

Detection. Novel sequence insertions are not detectable by read depth approaches. The reads that have been sampled from the novel inserted sequence do not map to the reference because the novel sequence is per definition not part of the reference.

Paired-end reads that overlap the junction of the inserted NRS become *one-end-anchored pairs* because only one of the reads can be mapped to the reference genome (see read pair o in Figure 2.6). Read pairs that are sampled completely from the inserted sequence do not map to the reference at all (read pair u in Figure 2.6).

A reads that overlaps one of the junctions can undergo clipping. This causes only the part of the read that has been sampled from the non inserted sequence to be mapped to the reference genome. The remaining part of the read will be clipped during the alignment, as shown for read pair c in Figure 2.6.

While the mere presence and genotype of an insertion can be assessed from above signatures, the determination of the actually inserted base pair sequence requires an assembly of the unmapped reads. The one-end-anchored reads can then be used to connect the newly assembled contigs and the reference sequence.

Due to the necessity of an assembly and the availability of PopIns[42;50], which already performs this task at population-scale, PopDel does not detect or genotype novel sequence insertions.

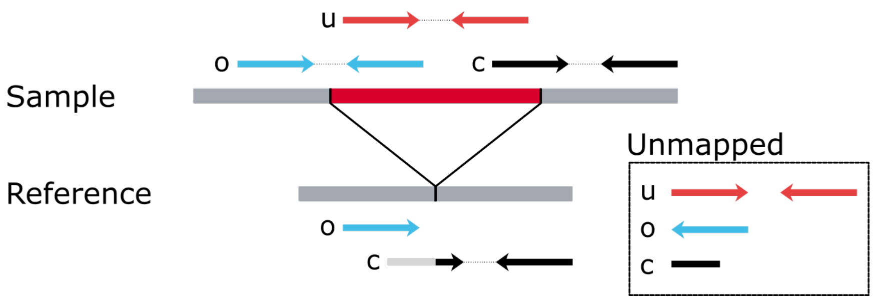

Figure 2.6: Read pair signatures of NRS insertions. A NRS insertion (marked in red) causes unmapped read pairs (read pair u) and one end anchored pairs (read pairs o and c). The upstream read of read pair c is further affected by clipping as its left end cannot be mapped to the reference.

2.1.4 Inversions

In the context of this thesis an inversion is defined as a piece of consecutive sequence of at least 50 bp length that is replaced by its reverse complement in the sample sequence when compared to a reference sequence. Figure 2.7 shows a conceptual example of an inversion.

Inversions are among the rarer SVs. The gnomAD-SV data base reports a median of 14 inversions per human genome[16]. In contrast to other SVs, the frequency of inversions is barely affected by their size (see Figure 2.2).

An inversion that disrupts the gene encoding for the blood-clotting protein Factor VIII has been been linked to sever hemophilia A[4].

Detection. Inversions are copy neutral: They do not cause any sequence to be gained or lost. Therefore, inversions cannot be detected by changes in the read depth alone, albeit local drops in read depth can sometimes be observed at the breakpoints of an inversion.

Read pairs that overlap one of the breakpoints of an inversion in the sample sequence become discordant when mapped to the sample genome. Depending on the overlapped breakpoint, the orientation of a pair changes either to forward-forward (FF; read pair d in Figure 2.7) or reverse-reverse (RR; read pair d' in Figure 2.7). This is the main signature PopDel uses for the detection of inversion. In addition to the change in read pair orientation, the read pair distance increases depending on the length of the inverted sequence and the location of the reads relative to the breakpoints. PopDel considers both the change in orientation and increase in read pair distance when detecting inversions.

Split reads can occur when one read of a pair is overlapping one of the inversion breakpoints as shown for read pair s in Figure 2.7. A correct split read alignment sharply defines two breakpoints of the inversion. PopDel does not perform a split read alignment but uses the information of the soft clipped bases at the 3'-ends of the reads for the calculation of the exact breakpoint positions.

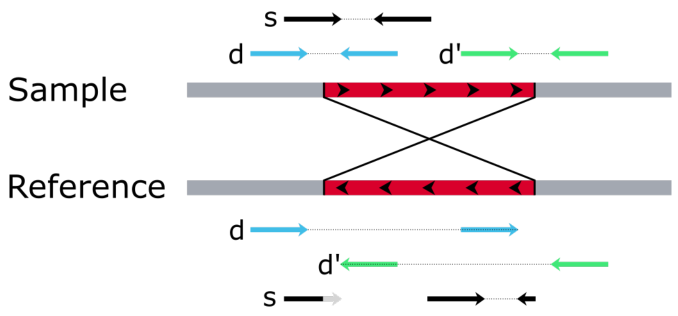

Figure 2.7: Read pair signatures of inversions. An inversion (marked in red, black arrows indicating sequence direction) causes an increase in read pair distance and a change in orientation for read pairs d and d'. Further, a split read alignment occurs for the first read of read pair s.

2.1.5 Translocations

In the context of this thesis an translocation is defined as a piece of consecutive sequence of at least 50 bp length that is moved from its original location (as defined by a reference genome) in the genome to another genomic location. If the new location is on the same chromosome as the original location, this is called an intrachromosomal translocation. If the sequence is translocated to another chromosome, this is called an interchromosomal translocation.

Translocations are called *reciprocal* or *balanced* if no sequence is gained or lost during the process, as shown in Figure 2.8. If sequence is gained or lost, it is called *unbalanced*. Unbalanced translocations count as CNVs.

Translocations are very rarely detected in healthy genomes. gnomeAD does not report any frequency for translocations in the 14,216 examined samples[16].

An example for disorders that can be caused by translocations is the Lynch syndrome. It can be caused by a translocation disrupting the MLH1 gene[33]. The MLH1 protein is involved in the DNA mismatch repair mechanisms and loss of function mutations in its sequence have been associated with various types of cancer.

Detection. Read depth approaches can detected the increased coverage caused by unbalanced translocations but will most likely classify the event as a duplication instead. This is because the change in read depth alone cannot be used to infer the translocation target site.

Read pairs overlapping one of the breakpoints of an interchromosomal translocation will have both of their reads mapped to different chromosomes. Additionally, depending on which parts of the chromosome(s) were translocated, their orientation will change from FR to one of RF, FF or RR. PopDel considers both the mappings to different chromosomes and change in orientation for detecting translocations. It does not detect intrachromosomal translocations.

Split reads can occur if a read is overlapping a breakpoint of a translocation. In this case both parts of the split read alignment will map to different chromosomes, sharply defining two of the breakpoints of the translocation. PopDel does not perform a split read alignment, but uses the information of the soft clipped bases at the 3'-ends of the reads for the calculation of the exact breakpoint positions.

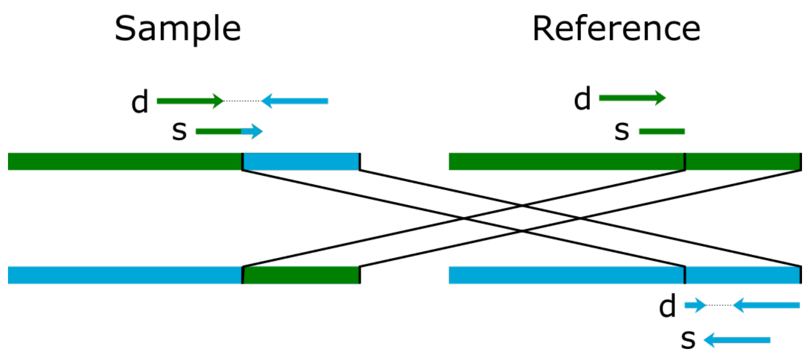

Figure 2.8: Read pair signatures of interchromosomal translocations. An interchromosomal translocation (indicated by the exchange of green and blue blocks) causes the two reads of read pair d to map to different chromosomes. Further, a split read alignment occurs for the first read of read pair s. Here both parts of the split read alignment map to different chromosomes.

2.2 Population Analysis for Small Variants

Joint analysis for small variants has already been established by the GATK *Haplotype-Caller*[70] (GATK HC). It has been shown to scale up to 90,000 exome sequencing samples without loss in accuracy.

GATK HC performs a haplotype assembly of all samples in the cohort. The locally assembled haplotypes are then compared against the reference to assess the presence of SNVs and indels. If one were to apply this assembly in a naive fashion by using the reads of all samples together, the approach would not scale to many samples due to the exponential increase in running time and memory consumption[70]. GATK HC solves this problem by applying a sample wise assembly in a module called the GATK *Reference Confidence Model* (GATK RCM) before calculating the genotypes using a paired Hidden Markov Model[21] (pair-HMM).

Reference Confidence Model. GATK RCM identifies regions of a genome where there is some evidence of variation[70]. To this end, it calculates an *active probability* for each position of the genome. This is done by considering the pile-up at each position and performing a simple genotyping that is based on alignment evidence of mismatches, indels and soft clipping. A Gaussian kernel is used to spread and combine the variant probabilities across neighboring positions. If the active probability is above a generous threshold for enough consecutive positions, the window (of 50 to 300 bp) is marked as *active*. Only reads from active regions and in 100 bp vicinity (by default) are considered in downstream steps. Larger active regions are split.

For each active region, a graph based local assembly is performed. The reads of the active region and the reference are segmented into k-mers and are used for creating a de Bruijn graph like structure. Vertices represent k-mers and edges are connecting overlapping k-mers. The edges are weighted according to the number of reads that contain both connected k-mers. The graph is simplified and the paths through the graph can be used to infer the candidate haplotypes. Finally, each candidate haplotype generated from the graph is aligned to the reference using the Smith-Waterman algorithm.

HaplotypeCaller. GATK HC takes the assembled candidate haplotypes of the GATK RCM and uses a pair-HMM to determine the likelihood $P(R_i|H_j)$ that each candidate haplotype H_j is represented by each read R_i[70]. This is done by calculating an alignment score of the alignment of each read to each haplotype. The score is based on the states of the pair-HMM which are *match/mismatch*, *insertion* and *deletion*. The transitions probabilities between the states define the penalties for opening or extending a gap.

The alignment of a haplotype to the reference that has been previously performed by the GATK RCM yields a set of variants at those alignment positions that do not match the reference. To genotype each sample for those variants, the likelihoods from the pair-HMM are used in a Bayesian model for calculating the posterior probabilities of the genotypes given the data:

$$P(R_i|G_g) = \prod_i \left(\frac{P(R_i|H_1) + P(R_i|H_2)}{2} \right)$$

$$P(G_g|R_i) = \frac{P(G)\,P(R_i|G_g)}{\sum_g P(R_i|G_g)\,P(G_g)}$$

with $P(G)$ as the prior genotype probability.

Together, GATK HC and RCM output gVCF files for all samples. They are closely related to usual VCF files but contain additional information for each variant. They further also contain records of non-variant sites, including the confidence that the respective site is different from the reference, even if it has not explicitly been called as a variant. The *GATK* tool *CombineGVCFs* is then used to combine all gVCFs into a multi-sample gVCF, before the GATK tool *GenotypeGVCFs* is used to determine the final genotypes of all samples for all variants. It further removes low quality calls and artifacts. The calculations performed by *GenotypeGVCFs* for the determination of the final genotypes are not publicly documented, as stated in GATK's technical documentation[1], and are therefore not summarized here.

One can argue that this workflow does not constitute a classical joint calling but rather a joint genotyping. The haplotypes are generated on a per-sample basis and the initial

[1]`https://gatk.broadinstitute.org/hc/en-us/articles/360035890511`

genotypes are also based on the data of the individual samples. It is only after the combination into a multi-sample gVCF file that the information across the samples is shared. However, the maintainers of GATK argue that the genotyping step is the only part of the workflow that needs to be performed jointly[2]. Further, the gVCF files contain additional information like the reference confidence and genotype likelihoods for non-variant sites. This allows the joint genotyping to come to well-funded results for all variant sites, even if a specific site was not considered to be a variant in the sample wise calling.

2.3 Current State of Structural Variant Detection

Currently, there are various programs available for the detection of SVs in NGS data. Here, the approaches of the SV callers Delly and Lumpy are summarized as examples. Their performance is also evaluated along with PopDel and other callers in Chapter 4.

While all of the presented SV callers were originally designed for SV calling on single samples or small cohorts, they are capable of processing cohorts of many samples together using a sample wise calling and subsequent regenotyping approach.

2.3.1 Delly

Delly can detect all types of SVs, including complex SVs[74]. It uses the signatures from discordant paired end reads (or mate pair sequencing, which is not discussed here) for the detection of SVs. Split read alignments are used as further evidence and for the refinement of breakpoints.

Discordant Read Pair Analysis for Breakpoint Detection. In a first step, the discordant read pairs of the sample are collected[74]. Delly considers a read pair to be discordant if its insert size is three standard deviations or more above the median insert size or if the pair is not in FR orientation. Reads that have multiple mapping location are filtered out. The discordant pairs are binned by chromosome and sorted according to their left-most alignment positions. They are then used to build an undirected weighted graph. The vertices of the graph represent the individual read pairs and the edges indicate mutual support for the same SV. The support for an SV is based on the SV signatures discussed in Subsection 2.1. The weights of the edges are based on how different the SVs inferred from the two connected vertices are. The edges are sorted according to their weights and a maximal clique finding heuristic is applied on the graph. The detected maximal cliques then correspond to distinct SV candidates.

[2]https://gatk.broadinstitute.org/hc/en-us/articles/360035890431-The-logic-of-joint-calling-for-germline-short-variants

Split Read Alignment for Breakpoint Refinement. The read pairs of each clique, i.e. SV candidate are used to infer a set of breakpoint candidates[74]. For their refinement, Delly searches for one-end-anchored pairs that map within two standard deviations of the breakpoints because the unmapped reads of such pairs are used as candidates for split read alignments. A k-mer filter is applied for reducing the size of the split read candidate set before the split read alignment around a breakpoint candidate is performed. To improve the running time of this process, the reference sequence is modified such that all reads can be aligned as if the SV was a simple deletion. The consensus of the split read alignments gives the exact breakpoints of the SV.

In a final step, Delly checks if the SV size inferred from the discordant read pairs matches the size that is inferred from the split read alignments. All passing SVs are subsequently merged such that the breakpoints that belong to the same SV event are combined into a single call.

Delly's refinement of the breakpoints by using split read alignments greatly increases its precision. It moderately reduces its recall, mainly at lower coverages or for very short read lengths.

SV Calling on Multiple Samples. According to the user guide provided by Delly's maintainers[3], Delly should be applied on a single sample basis for high coverage genomes or small batches for low-coverage genomes. Delly provides functions for the single sample calling, the subsequent merging of the SV call sets and the regenotyping of the single samples. Section 4.5.1 contains such a workflow.

2.3.2 Lumpy/Smoove

Lumpy is described as a "general probabilistic framework for SV discovery" by its authors[54]. It is capable of integrating different signatures of SVs by representing breakpoints as pairs of probability distributions. The distributions reflect the uncertainty concerning the exact positions of the breakpoints. By combining these distributions, Lumpy calculates the integrated probability distributions that are used for predicting the SVs. Lumpy extracts the signatures of discordant read pairs and split reads internally, but is further capable of using additional evidence provided by the user in BEDPE[58] format. Figure 2.9 visualizes this approach. Lumpy is capable of detecting deletions, tandem duplications, inversions and translocations. It does not perform genotyping.

[3]`https://github.com/dellytools/delly`

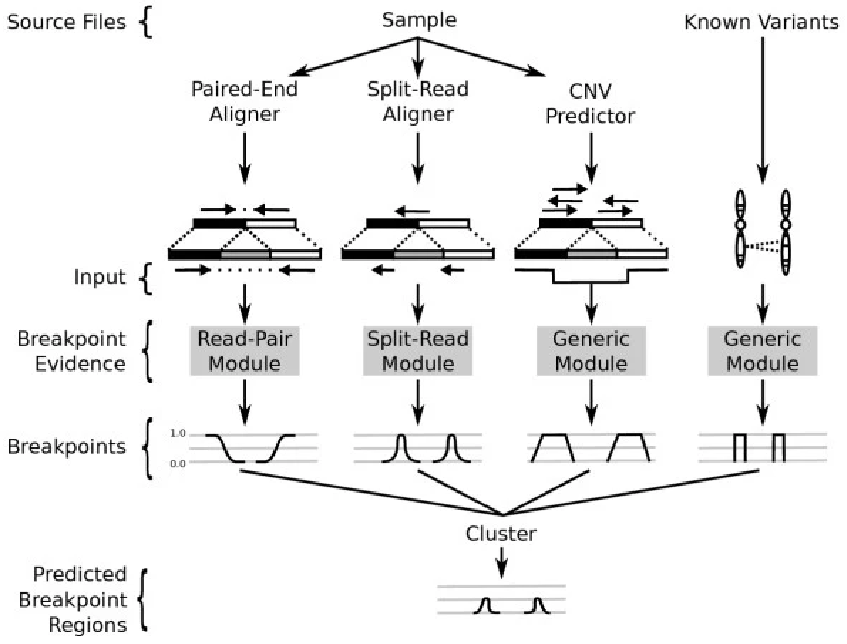

Figure 2.9: SV signature integration approach applied by Lumpy. Breakpoints are represented by probability distributions that reflect their positional uncertainty. Figure taken from Layer et al. (2014)[54].

Distributions Generated from Discordant Read Pairs. Lumpy considers the orientation and insert sizes of discordant read pairs to generate the probability distributions representing the candidate breakpoints[54].

First, Lumpy assigns the SV type to each breakpoint by considering the orientations of the read pairs (see Section 2.1). Then, the probability distributions of the breakpoints are generated by calculating the probabilities that two reads are spanning each position of the breakpoint regions. This in turn depends on the observed distribution of insert sizes in the sequenced sample and the observed mapping locations of the reads. For a read pair to span some assumed breakpoint position, its insert size must be large enough such that it covers the range between its leftmost read and the breakpoint. If one calculates this probability for a breakpoint position that is only a few bp downstream of the leftmost read of a pair, one gains a high probability. If the considered location is far downstream, the distribution of insert sizes of the sample returns a low probability instead. This leads to the probability distributions of the breakpoints as seen for the *paired-end aligner evidence* in Figure 2.9.

Distributions Generated from Split Read Alignments. Like for discordant read pairs, the orientations of different split read parts are used to assign SV types to the breakpoint candidates[54]. As already discussed in Section 2.1, split read alignments typically define sharp breakpoints. Still, some ambiguity may arise from sequencing or alignment errors. To account for, this Lumpy places probability distributions around the breakpoints induced by the split read alignment. The distributions have their maxima centered at the points where the read has been split and exponentially decrease with increasing distance. The quality of the sequencing data and alignments control the rate at which the probability distributions decay.

SV Calling on Multiple Samples. Lumpy can perform a joint calling by integrating the probability distributions from multiple samples to determine the breakpoints of the SVs. However, for larger cohorts the recommended workflow consists of a single sample approach with subsequent merging[4].

Wrapping Lumpy via Smoove. The recommended way of using Lumpy is via the Smoove[69] pipeline[5]. It manages the extraction of discordant read pairs and split reads required by Lumpy and performs filtering steps (see below) that reduce the running time and false-positive rate of Lumpy. It further applies SVTyper[14] for genotyping, because Lumpy does not perform this step. Steps applied by Smoove include:

- extraction of discordant read pairs and split reads
- filtering of extracted read pairs based on region, coverage and quality
- calculation of required metrics for each sample
- SV calling using Lumpy
- genotyping using SVTyper
- processing of the VCF output

Where possible these steps are performed in parallel.

The authors of Smoove recommend a joint calling approach for small cohorts of less than 40 samples[6]. For bigger cohorts, a sample wise calling with subsequent merging and regenotyping is recommended. Section 4.5.1 contains such a workflow.

In the benchmarks of this thesis, Lumpy was always applied via Smoove.

[4]`https://github.com/brentp/smoove#population-calling`
[5]`https://github.com/arq5x/lumpy-sv#quick-start`
[6]`https://github.com/brentp/smoove#small-cohorts-n---40`

Chapter 3

Joint SV Calling and Genotyping in PopDel

This chapter focuses on the main contribution of this thesis: The SV calling tool PopDel. The basic ideas behind its approach and details regarding applied algorithms, as well as implementation, will be discussed.

3.1 Basic Idea and Aim of PopDel

The aim behind PopDel is to allow for efficient joint calling of SVs in tens of thousands of whole genomes. Previous efforts in the field of small variants have shown the scalability and benefits of joint calling approaches, some of which are an increase in the number of detected variants, especially in the regions with low sequence coverage and a decrease in false positive calls resulting from improved filtering[70]. Not only the detection, but also the genotyping can benefit from the consideration of multiple samples as demonstrated by the tool GraphTyper, which uses *pangenome graphs* to detect small variants and calculate their genotypes in population scale sequencing data. But for structural variants, the potential of joint calling approaches has not yet been fully explored.

PopDel approaches the challenge of joint SV calling in two steps: In a first step, relevant information from the BAM file of each genome is extracted and stored in small read pair profiles (subsequently only called profiles). The profile format is explained in detail in the Subsection 3.2.4 and allows to minimize I/O - one of the key challenges for joint calling on population-scale. The subsequent joint calling, described in detail in Section 3.3, only relies on the profiles and uses the provided information from all samples together to calculate likelihoods for the presence of SV along with their genotypes while moving along the genome in tiling genomic windows. Using the data from all samples jointly in a statistical

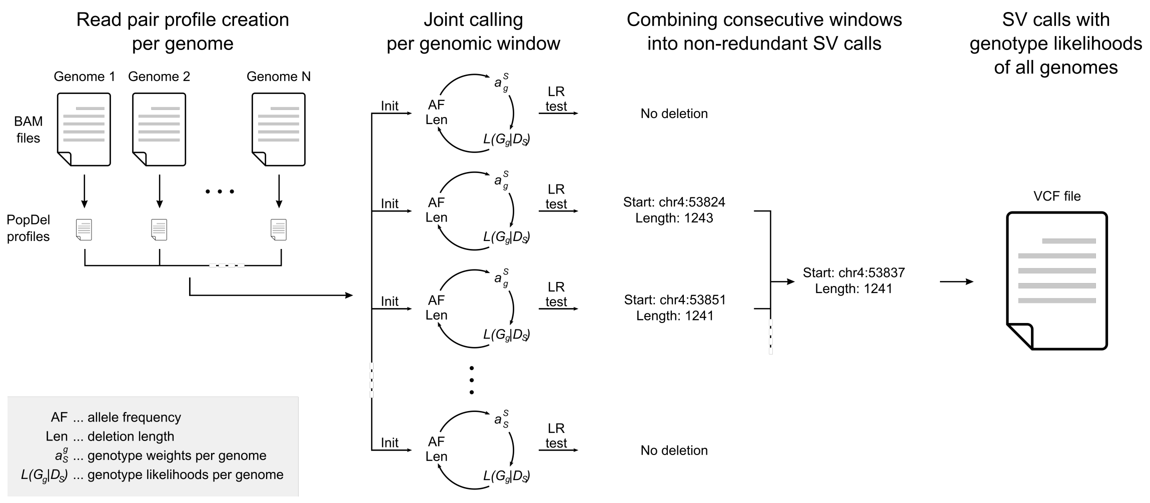
Read pair profile creation per genome
Joint calling per genomic window
Combining consecutive windows into non-redundant SV calls
SV calls with genotype likelihoods of all genomes
Genome 1
Genome 2
Genome N
BAM files
PopDel profiles
Init
AF
Len
a_g^S
$L(G_g|D_S)$
LR test
No deletion
Start: chr4:53824 Length: 1243
Start: chr4:53851 Length: 1241
Start: chr4:53837 Length: 1241
VCF file
AF ... allele frequency
Len ... deletion length
a_S^g ... genotype weights per genome
$L(G_g|D_S)$ ... genotype likelihoods per genome

Figure 3.1: PopDel's joint deletion calling workflow. For the detection of SVs other than deletions the step "joint calling per genomic window" is performed with a window-wise calling procedure tailored for the respective SV type.

framework specifically designed for the integration of multiple samples, allows PopDel to detect variants which might be missed when looking only at single samples. Careful integration of *sample-specific genotype weights* for the likelihoods guarantee that variants are not overseen if they are only present in a few or even a single sample among thousands of other samples. Overlapping variants are detected by inspecting windows that harbor multiple SV candidates with multiple initializations of the variant parameters. Figure 3.1 shows the general workflow for the profile creation and joint calling and genotyping of deletions. For duplications, inversions and translocations, the step *joint calling per genomic window* is performed with the window-wise calling procedures tailored for the respective SV type.

All together, this allows PopDel to scale to tens of thousands of genomes in a user friendly workflow and to produce reliable SV calls and genotypes in single samples as well as population-scale data.

The initial aim of this thesis and PopDel only encompassed the detection and genotyping of genomic deletions at population-scale. The approach has since been extended to other types of SVs, specifically duplications, inversions. The calling and genotyping of duplications and inversions is however still a work in progress. Further, an early prototype for the detection and genotyping of interchromosomal translocations has been implemented.

For an overview of PopDel's parameters, please consult Section A.1 of the Appendix. A detailed user-guide for PopDel is available in its GitHub-Wiki[1].

[1] `https://github.com/kehrlab/PopDel/wiki`

3.2 The PopDel Profiling

This section describes the first step in PopDel's analysis workflow: The creation of the read pair profiles from the alignment file of a genome.

3.2.1 Basic Idea and Workflow of the PopDel Profiling

As described in previous sections, one of the main problems of population-scale variant calling is the scaling to big sample numbers. To allow for an efficient calling process and reduce unnecessary I/O operations, this thesis introduces the PopDel read pair profiles. The read pair profiles are designed to hold only the relevant information required for the later joint calling step, which are

- chromosomes and positions of read pairs on the reference genome
- insert sizes of the reads in a pair when mapped to the reference genome
- orientation of the reads in a pair when mapped to the reference genome
- number of soft clipped bases of the reads' 3'-ends when mapped to the reference genome
- for read pairs with the reads mapping to different chromosomes: Chromosome and position of the other read in the pair

Only information of read pairs satisfying the basic quality criteria described in Subsection 3.2.2 are included in the profile. Further, all information are stored in windows of 256 bp for reasons of compression described in Subsection 3.2.4. Together with the binary compression and the profile index (see Subsection 3.2.5) this allows to store the information for calling in as little as 1-2 % of the disk space requirements of the original BAM file, and access the information at any position of the profile without the need to linearly iterate through the whole file. This also leads to the possibility to efficiently limit the calling to individual sections of the genome without much overhead, which is especially practical for distributing many calling jobs across multiple nodes of a compute cluster for parallel processing. The variant calling problem is embarrassingly parallel when approached on a per-chromosome basis. For example, a user could start 24 instances of PopDel's calling process on the same set of profiles, each working on a different chromosome. The output of the individual processes can then later be concatenated. More small-scale partitioning of the genome to the sub-chromosomal level is also possible.

An additional motivation and advantage behind the creation of the profiles is that it reduces the overhead for later analyses when additional samples are added after the variant calling has already been performed on the previous samples. Only the profiles of the newly added samples have to be created before they can be used for the joint calling together with the already existing ones, saving a lot of time and computational resources.

The profiles are created from the position-sorted and indexed BAM files of short whole-genome reads. CRAM files are also supported and require the FASTA file of the reference sequence if it is not embedded in the CRAM file. In the first phase of the profiling, the empirical distribution of the insert sizes of the read pairs is sampled from some well behaved genomic regions. These regions are predefined for the most recent builds of the human genome, but can be specified by the user for other use cases. Since one alignment file can hold the information of multiple sequencing libraries for the same sample, all information is collected on a per-read-group basis to allow for different libraries with different properties like different insert sizes distributions.

After the sampling, PopDel iterates all records of the alignment file in a second linear pass. Each record is compared against the quality criteria and filters described in Subsection 3.2.2. Only records that satisfy all criteria and for which the other read in the pair also passes all filters are further processed. Because one read in a pair typically occurs somewhere downstream in the alignment file, the read that occurs first in a pair that passes all filters is stored in a buffer, until the downstream read is encountered. This strategy eliminates the need for traversing the file in search of the downstream read in the pair, which would lead to a substantial increase in I/O and running time. For each read pair that is passing all filters the information listed above are determined. These information are aggregated for all read pairs in a 256 bp window. Once such a window has been passed and no more records can occur in the window, the window is compressed and written to the output stream. The details of this process are described in Subsection 3.2.4.

During the whole process PopDel keeps track of the file positions in the output file. After all records of the alignment file have been processed, these information are used to build an index of the profile that is stored at the beginning of the profile. The index is described in Subsection 3.2.5.

One BAM file can contain data from multiple *read groups* of the same individual. Typically, one read group defines a set of reads that where generated in the same run of the sequencing machine using the same sequencing library. In a BAM file, they are declared by a special tag. PopDel operates on the read group level. This means that information like the distribution of insert sizes is estimated for each read group of the sequenced genome individually. This allows PopDel to work with BAM files that contain read groups from different sequencing libraries with different properties regarding the size of fragments and their distribution.

3.2.2 Quality Criteria and Filters

When examining the records of a BAM or CRAM file, PopDel applies some basic quality criteria. By not further processing filtered records, the following objectives are achieved: Reduction of running time for profile creation, reduction of disk space required to store the profiles, reduction of artifacts and noise caused by (likely) faulty records, and reduction of running time for the calling. By default they are chosen to be as permissive as possible to avoid a loss of sensitivity caused by overfiltering. One of the main criteria for the decision is the status of the SAM flag of the records. They are encoded as strings of bits with length 12. Each bit encodes for a specific property of the record and can be set during the alignment process. As strings of bits can be interpreted as (base-ten) numbers, one can combine multiple flags into a single unsigned 16 bit integer. This integer can then be used together with the SAM flag of the records in bitwise AND operations to check all conditions simultaneously. A record and its associated record are filtered out if one or more of the following conditions apply:

- one or both reads of a pair do not map to the reference (i.e. SAM flag 0x4 or 0x8 is set)
- the read is not a primary alignment (i.e. SAM flag 0x100 is set)
- the read fails platform/vendor quality checks (i.e. SAM flag 0x200 is set)
- the read is a PCR or optical duplicate (i.e. SAM flag 0x400 is set)
- the record is a supplementary alignment (i.e. SAM flag 0x800 is set)

Further, the following conditions must be met to ensure that the fundamental information required by PopDel (i.e. the insert size or location of both reads in the pair) are present and the read locations assigned by the read mapper are reasonably certain:

- the read must be paired (i.e. SAM flag 0x1 must be set)
- at least 50 bp of the read must be aligned to the reference after accounting for clipped bases
- the mapping quality of the record must be at least 1
- the alignment score of the record divided by its length must be at least 0.8

With exception of the pairedness of the reads, all of the above values are user configurable to allow for special use-cases and individual preferences. To ensure that the distribution of the insert sizes is only sampled from well behaved read pairs, there are additional requirements for the records that are used for the sampling of the insert size histograms:

- the upstream read must align in forward orientation (i.e. SAM flag 0x10 must not be set)
- the downstream read must align in reverse orientation (i.e. SAM flag 0x10 must be set)
- both reads of the pair align to the same chromosome

Read pairs that map to different chromosomes are additionally compared against a blacklist of genomic regions. A default blacklist that contains gap regions and repetitive regions of GRCh38 is provided, but the blacklist can be fully defined by the user. If one or both reads of a pair fall into a blacklisted region, the read pair will not be added to the profile. Read pairs that do not map to different chromosomes are not affected by this blacklist. The aim of the blacklist is the reduction of the number of read pairs that are falsely considered as evidence of translocations because one read of the pairs maps to a wrong chromosome.

3.2.3 Robust Estimation of Insert Size Histogram Parameters

The distribution of insert sizes of each read group of a sample is estimated from a predefined set of regions. These regions encompass 1,000,000 bp on each of the 22 human autosomes and are defined for the most recent (hg19, hg38) human genome builds. The user is free to specify a set of individual sampling regions for special use cases or other organisms. Each read pair that passes the filters described in Subsection 3.2.2 is used for creating a read group specific empirical histogram of insert sizes. As soon as the required minimum number of read pairs per read group has been reached (default 50,000) no further sampling regions are processed. The default sampling regions can be found in Section A.2 of the Appendix.

After the sampling, initial estimates for the median μ and standard deviation σ of each insert size histogram are calculated. To mitigate the effect of outliers and get a robust estimate for μ and σ, the initial estimates are refined in an iterative process. The insert size histogram is trimmed to only contain insert sizes that fall into the interval $[\mu - 3\sigma, \mu + 3\sigma]$. μ and σ are then recalculated from this trimmed histogram and the process is repeated until μ and σ converge. The trimmed histogram and the refined estimates are then stored in the PopDel read pair profile as described in the next section.

3.2.4 The PopDel Read Pair Profile Format

The PopDel read pair profile is defined as a binary file format, whose definition is given in Table 3.1. The main directive in designing the profile format was to create a format that allows for the storing of all information necessary for the subsequent joint SV calling, while requiring as little disk space as possible. Since the need to hold much data in the main memory would quickly become a problem when performing the calling on many samples simultaneously, the profile format also has to allow for efficient streaming and present the information in small blocks. Further, to avoid linear scans of the file to find a certain position, the profile format should contain its own index. This is essential if the joint calling should be restricted to a small portion of the genome. The final implementation of the PopDel read pair profile that achieves these goals can be conceptually divided into seven parts:

1. profile identifier: magic string and profile version
2. index for jumping to a given positions or chromosomes
3. sample information: Names and number of read groups
4. information on each read group: read length, histogram of insert sizes, median and standard deviation of insert size distribution
5. Chromosome information: names, lengths and number of reference sequence chromosomes
6. single-chromosome block: coordinate sorted list of 256 bp windows with read pair information: genomic coordinate, number of read pairs starting in the window, information on distance between the reads of the pair, orientation and soft-clipping.
7. multi-chromosome block: coordinate sorted list of 256 bp windows with information on read pairs mapping to different chromosomes: number of read pairs falling into the window, genomic coordinates of both reads in a pair, information on soft-clipping and orientation.

An example of a human readable representation of the sorted lists of windows is given in Table 3.2 for read pairs mapping to the same chromosomes and in Table 3.3 for reads mapping to different chromosomes.

The size of the single windows is set to 256 bp because the exact genomic positions of the read pairs in the windows are not stored explicitly, but can be calculated as the sum of coordinate of the beginning of the window and the stored offset value of the read pair. That way, the maximum size for the offset, which is stored for each read pair, is limited to 256 bp, thus making it suitable to be stored in a (8 bit) char, rather than an (32 bit) integer, requiring only $\frac{1}{4}$ of the space per position. Further, as all read pairs in a window of the single-chromosome block originate from the same chromosome, it is sufficient to store the chromosome identifier only for the window and not for the individual read pairs, further reducing the required disk space.

Additional compression is reached by encoding and packing of the read pair orientations in a window. As one read can align either in forward or reverse orientation, there are four possible combinations of orientations in a read pair: Forward-reverse (FR), forward-forward (FF), reverse-forward (RF) and reverse-reverse (RR). These four combinations can be encoded as pairs of two bits. As the smallest processable unit in most systems is a byte, which is in turn made up of eight bits, one byte can hold information on the orientations of four read pairs. While storing the orientation of each read pair individually would take one byte, packing four 2-bit encoded orientations into one char reduces the amount of required disk space by 75 %. Another factor enabling further compression of orientation information is the circumstance that the majority of read pairs is expected to be in FR orientation. For example, the profile for the genome NA12878, mapped to GRCh38 (517,486,045 total records in the BAM file), contains 225,615,536 read pairs in the single-chromosome block. 225,233,784 (99.83 %) of those read pairs are in FR orientation and only 339,808 (0.15 %), 22,657 (0.01 %), 19,287 (0.01 %) read pairs are in RF, FF, or RR orientation. Consequentially, it is on average beneficial to only store the packed orientations for read pairs that have an orientation different from FR. This comes at the price of storing information on which read pair an orientation in the string of packed orientations O is referring to. This is solved by storing the rank of the read pairs in the windows (i.e. the order of occurrence) that have an non-FR orientation in an additional string R of unsigned integers. This way, the i-th orientation in O refers to the read pair that is referenced as the i-th element of R. Storing R requires 16 times as much space as storing only the orientations, but since only the orientation information for ~0.17 % of read pairs has to be explicitly stored when using R, this easily amortizes. In case the number of read pairs with non-FR orientation in a window is not a multiple of four, the packed string of orientations is padded with trailing zeros.

Similarly, the information on the clipping of the read pairs in a window is only stored if it is different from zero. Only 6,119,731 read pairs in the aforementioned profile of NA12878 have clipped bases on either 3'-end, accounting for only 2.7 % of all read pairs in the profile, making the approach of storing a combination of read pair identifier and number of clipped bases feasible. As the number of clipped bases is stored in chars, no packing like for the orientations is required.

The *multi-chromosome block* stores the information on read pairs where both reads map to different chromosomes. Therefore, the format for storing the required read pair information as described for the single-chromosome block applies only partially. Firstly, the concept of distance between the reads of a read pair no longer holds as they are mapping to different chromosomes. Secondly, a record in the multi-chromosome block requires different information on the chromosomes for both reads in a pair. Table 3.3 gives an example of the human readable representation of how these read pairs are stored. Each window is characterized by the chromosome it is located on and its start position. Each read pair in the window is specified by the offset of the 3'-end of the one read mapping into the window from the starting point of the window. In addition, the reference ID of the chromosome and the explicit location the other read is mapping to is stored. Soft-clipping is stored for both the first and the second read in the pair. The orientations of the read pairs are packed as described above, but this time also FR orientations are included. The reason behind this is that the assumption that most of the read pairs are in FR orientation does no longer hold, so that the need to store the string of read pair IDs would likely cost more storage space than the packing is saving: The multi-chromosome block for the NA12878 profile consists of information on 915,010 read pairs. Of those, 457,089 (50.2 %) are in either FR or RF orientation, 228,271 (24.95 %) in FF orientation and the remaining 229,650 (25.1 %) are in RR orientation. The FR and RF reads cannot strictly be distinguished because the order depends on the chromosome one is considering first. Another special point about the multi-chromosome block is that it holds the information of each read pair in two windows, one for each chromosome a read of the pair is mapping to. Since only a minor portion of all read pairs is stored in the multi-chromosome block it is feasible to store all of the information redundantly. This is necessary for the efficiency of the streaming of the profiles during the joint calling which requires all information about a window to be readily and locally available without searching or jumping in the file. Therefore, the information on the read pairs in Table 3.3 are present in the windows on chr21 and also in those on chr22. Otherwise, only one of the novel junctions of a balanced translocation could be detected on one chromosome, even though another chromosome is involved.

All above mentioned factors together and the fact that the profile format does not rely on the actual sequence of the read pairs allows for a very efficient storing of the information required for the joint calling. Typically, the size of the profile is between 1-2 % of the original BAM file size. For example, the file size of the original BAM file used for the creation of the aforementioned NA12878 profile amounted to 51,943,573,793 bytes ($\approx$ 51.9 GB). The profile on the other hand, can be stored in as little as 883,288,656 bytes ($\approx$ 0.8 GB), thus requiring only 1.7 % of the original file size. This makes it feasible to store the profiles of many samples.

Table 3.1: Definition of the PopDel profile format.

Field	Description	Type	Value
magic	PopDel magic string	char[7]	POPDEL\1
version	Version of the profile format	uint16	3
l_region	Index region size	uint32	
n_file_offsets	# file offsets in the index (index size)	uint32	
n_trans_file_offsets	# file offsets in the translocation index (translocation index size)	uint32	
List of file offsets (n=n_file_offsets)			
file_offset	File offset (index entry)	uint64	
List of file offsets (n=n_trans_file_offsets)			
trans_file_offset	File offset (translocation index entry)	uint64	
l_sm_name	Length of the sample name plus 1 (including NUL)	uint32	
sm_name	Sample name; NUL-terminated	char[l_sm_name]	
n_rg	# read groups	uint32	
List of read group names and insert size histograms (n=n_rg)			
l_rg_name	Length of the read group name plus 1 (including NUL)	uint32	
rg_name	Read group name; NUL-terminated	char[l_rg_name]	
median_isize	Median insert size in this read group	int16	
stddev_isize	Standard deviation of insert size in this read group	int16_	
read_length	Length of reads in this read group	uint16	
hist_start	First insert size listed in histogram	uint16	
hist_end	Last insert size listed in histogram plus 1	uint16	
List of read pair counts per insert size (n=hist_end - hist_start); insert size histogram			
count	# read pairs with corresponding insert size	uint32	
n_ref	# reference sequences	uint32	
List of contig names (n=n_ref)			
l_ref_name	Length of the reference name plus 1 (including NUL)	uint32	
ref_name	Reference sequence name; NUL-terminated	char[l_ref_name]	
l_ref	Length of the reference sequence	uint32	
List of windows (until translocation_guard is reached)			
refID	Reference sequence ID, $0 \leq$ refID $<$ n_ref	uint32	
pos	Start position of the window (0-based)	uint32	
List of read group entries for the window (n=n_rg)			
n_rp	# read pairs	uint32	
n_rp_clip	# read pairs with clipping, $0 \leq$ n_rp_clip $\leq$ n_rp	uint32	
n_rp_o	# read pairs with non-FR orientation, $0 \leq$ n_rp_o $\leq$ n_rp	uint32	
List of clipping of read pairs for the read group in the window (n=n_rp_clip)			
rp_id	ID (=position in the window)	uint_32	
clip	# Clipped bases	char	
List of packed orientations of non-RF oriented read pairs in the window (n=n_rp_o)			
rp_ids	IDs (=position in the window)	uint32 $[\lceil \frac{\text{n\_rp\_p}}{4} \rceil]$	
orientation	Packed orientation of non-FR oriented reads	char $[\lceil \frac{\text{n\_rp\_o}}{4} \rceil]$	
List of read pairs for the read group in the window (n=n_rp)			
pos_offset	Offset of read pair position from window start position	char	
idist	Distance between the 3'-ends of the read pair in bp	int32	
translocation_guard	Indicates the start of the multi-chromosome block	uint32	4,294,967,295
List of translocation windows (until the end of file)			
refID_1	Reference sequence ID of read 1, $0 \leq$ refID_1 $<$ n_ref	uint32	
pos_1	Start position of the window of read 1 (0-based)	uint32	
List of read group entries for the window (n=n_rg)			
n_rp	# read pairs in the windows	uint32	
List of packed orientations of all read pairs in the window (n=n_rp)			
orientation	Packed orientation of all reads.	char $[\lceil \frac{\text{n\_rp}}{4} \rceil]$	
List of read pairs for the read group in the windows (n=n_rp)			
pos_offset_1	Offset of read 1 position from window start position	char	
clip_1	# 3' clipped bases in read 1	char	
refID_2	Reference sequence ID of read 2, $0 \leq$ refID_1 $<$ n_ref	uint32	
pos_2	Start position of read 2 (0-based)	uint32	
clip_2	# 3' clipped bases in read 2	char	

Table 3.2: Human readable representation of the PopDel profile format. The first two columns give the positional information of a 256 bp window on the genome.

Chr.	Position	Read pair information
chr21	5035777	1:-2(0):FR, 2:16(0):FR, 38:44(0):FR
chr21	5036033	0:7577(25):RF, 18:7559(66):RF, 26:19(67):FR
chr21	5036289	19:10(0):FR, 31:-93(0):FR, 33:-11(0):FF
chr21	5036545	16:31(0):FR, 26:103(0):FR, 36:-110(0):RR

The read pair information is given as a comma separated list. The value before the first ':' gives the positional offset of the record, followed the distance between the 3'-end of the reads negative values indicate overlapping reads in a pair. In brackets, the sum of soft clipped bases at the 3'-ends is given. The two letter code encodes for the orientation of the read pair: FR: Upstream read forward, downstream read reverse; RF: Upstream read reverse, downstream read forward; FF: Both reads forward; RR both reads reverse.

Table 3.3: Human readable representation of the multi-chromosome block of the PopDel profile format.

Chr.	Position	Read pair information
chr21	10489601	223(4):R+chr22:45595763(0):F
chr21	10489857	19(0):R+chr22:45595723(0):F
...	...	...
chr22	45595649	74(0):F+chr21:10489876(0):R, 114(0):F+chr21:10489824(4):R

The first two columns give the positional information of a 256 bp window on the genome. The read pair information is given as a comma separated list. Each record can be divided into two parts, separated by a '+'. Firstly, similar to the single-chromosome block of the profile, the positional offset from the window start of the read mapping to the window is given, followed by the number of clipped bases at the read's 3'-end in brackets and the single letter encoded orientation (F - forward; R - reverse). After the '+', the information on the other read in the pair is given: The mapping position of the 3'-end, followed again by the number of clipped bases and orientation. Because PopDel is only aware of the local information (i.e. the information in the current window) when performing the calling, the same read pair information is stored again in the respective window of the other read in the pair, so that potential breakpoints can be detected on both chromosomes involved.

3.2.5 The PopDel Profile Index

A PopDel read pair profile contains its own index at the beginning of the file. Similar to the parts of the profile that are used for storing the read pair information, the index can be divided in two blocks: One block indexes the single-chromosome block, subsequently called the *single-chromosome index* and the other block indexes the multi-chromosome block, called the *multi-chromosome index*. Because the concept for both index blocks is identical only the single-chromosome block will be described. The concept behind the indices is reminiscent of the linear part of the SAM file index[58]. The aim of the indices is to provide information on where in the profile records of a specific genomic region are stored. This is done in a sparse fashion using the genomic starting positions of the 256 bp windows. The user can specify an index region size s (default 10,000 bp) to define how sparse the index is. An index region size of $s = 10,000$ bp means that for every 10,000-th genomic position of the linearized genome the exact file position where the 256 bp window containing the genomic position is located in the profile is stored. In this context, linearization of the genome means that all chromosomes are brought into a fixed sequential order (as given by the ordering of the chromosomes in the BAM header of the genome) and the positions are calculated as if the genome was one long contiguous sequence. Given a genome with chromosomes $c_0, c_1, \ldots, c_n$, let $l(c)$ be a function that returns the length of the sequence of the chromosome c in base pairs. Given some chromosome c_x of the genome and a position p_{local}, with $0 < p_c < l(c_x)$ the global position p_{global} is calculated as:

$$p_{\text{global}}(x, p_{\text{local}}) := p_{\text{local}} + \sum_{i=0}^{i<x} l(c_i) \tag{3.1}$$

The index does not need to explicitly hold the chromosome names of the stored positions because they are implicitly given by the global positions. Therefore, the index itself can be represented and stored as a list I of file positions. Each entry I_i holds the file position of the 256 bp window that contains the global position p^i_{global}.

To retrieve the file position stored in the index for a given genomic region PopDel first calculates the global position from it using Equation 3.1 and then divides this position by the index region size (rounded down) to get the rank i in I. Since the index is stored as a list of unsigned 64 bit integers, this enables the calculation of the file position where the index field is located by calculating $i \cdot 64 + 119$. 119 is the sum of the size of the fields before the index (see Table 3.1). Since the multi-chromosome index is located after the single-chromosome index, the size of the single-chromosome index has to be added to the result when calculating the file positions for the multi-chromosome index. In those cases where the exact global position is not stored in the index, it can be found by linear traversal of the profile's windows after jumping to the closest lower indexed position.

3.3 Joint Variant Calling in PopDel

This section describes the second step in PopDel's analysis workflow, the joint structural variant calling. Here, PopDel takes the read pair profile(s) of one or many genomes that have been mapped against the same reference genome as an input. It processes the profiles jointly to create a list of structural variants and per-sample genotypes. The calling does neither rely on the BAM files nor on the presence of the reference genome. The methods described in this section for the variant calling on multiple samples are also applicable when processing a single sample.

3.3.1 General Notation and Remarks

For the sake of comprehensibility and in order to avoid too many levels of subscripts or superscripts, all subsequent formulas and explanations assume a single read group per sample. The generalization for multiple read groups per sample is explained at the end of Subsection 3.3.3.2.

A *genomic window* or just *window* refers to a consecutive number of positions on the same chromosome in the genome. The positions are given with respect to the reference. Windows in PopDel are tiling the chromosomes in a non-overlapping fashion: Let $w_i = [l_i, l_i + w[$ be the i-th window on some chromosome, with l_i defining its left starting position on the chromosome and window-length w. The first window on a chromosome is defined as $w_0 = [0, w[$. For all subsequent windows on the same chromosome holds $w_i = [i \cdot w, (i + 1)\, w[$.

A read pair that maps to the same chromosome as w_i is considered to *overlap* w_i or to *be contained in* w_i if one or more positions between the (potentially soft clipped) 3'-ends of the pair (including the mapping position of the leftmost 3'-end) fall into the interval of the window. Let r_1, r_2 be the mapping positions of the 3'-ends of the reads in the pair and assume without loss of generality that $r_1 \leq r_2$. (r_1, r_2) then overlaps all w_i for which $l_i \leq r_1 < l_i + w$ or $r_1 \leq l_i < r_2$ hold. The reasoning behind this design choice is that SV signatures that were discussed in Section 2.1 are mainly exhibited in between the 3'-ends of the reads in a pair. The reads themselves offer little additional evidence for the detection or genotyping of an SV. For reads that are in any other orientation than RF, this definition of overlap also reduces the number of windows that contain a read pair. This in turn reduces the computational load because a read pair will be considered in less windows during the window-wise calling. Figure 3.2 visualized definition of read pairs that overlap one or more windows.

Throughout the subsequent paragraphs, the following notation will be used: Let $\mathcal{S}$ denote the set of all samples in the joint calling and let $S \in \mathcal{S}$ be a single sample. Let μ^S denote median read pair distance (see Section 2.1) of S and let σ_S denote the standard deviation

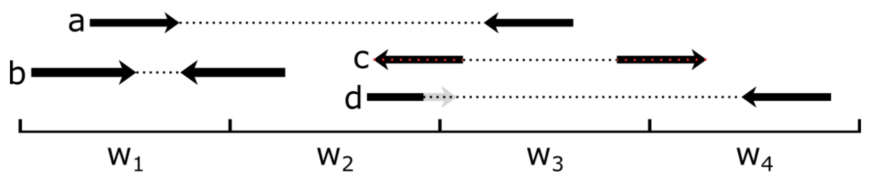

Figure 3.2: Read pairs overlapping windows. The dashed lines between the reads (arrows) indicate the region in which PopDel considers a read pair to overlap the windows (w_1 - w_4) of the genome. Read pair a overlaps w_1 to w_3. Read pair b overlaps only w_1. Read pair c overlaps w_2 to w_4 because the read pair is in RF orientation. Read pair d overlaps w_2 to w_4, because the clipping of its left read (indicated by the gray coloring of the tip of the arrow) moves its 3'-end mapping location from w_3 to w_2.

of the insert size histogram of sample S. The *deviation* of the read pair distance of a pair with read pair distance x is calculated as $\delta = x - \mu_S$. The set of all read pair distance deviations of all read pairs that overlap the current genomic window is denoted as Δ^S.

3.3.2 Basic Idea and Workflow

In the beginning of the joint calling process, PopDel first loads the insert size histograms of all profiles. The paths to the profiles can be given as a text file or directly via the command line. Subsequently, parameters of the calling, like the minimum required deletion length and the minimum likelihood ratio for a potential variant to be considered as significant, are calculated based on the distribution of insert sizes if those values have not been specified by the user. Next, the list of regions that are to be processed is initialized. If the user has not specified any regions of interest the calling should be limited to, the lists consist of all chromosomes of the genome as given in the header of the profiles. Overlapping regions of interest are merged. The joint calling then commences in a window-wise fashion. A window of 30 bp is moved across the genome (or the defined regions of interest respectively) in a non-overlapping fashion. In each window, the profiles are used to determine how many read pairs of each sample are overlapping the window.

For the detection of deletions the read pair distances of the read pairs overlapping the window are clustered by their deviation from the sample's median read pair distance using a simple hierarchical clustering scheme to gain a set of initial deletion values. The initial deletions are then refined together with their frequency in the set of samples in an iterative approach based on the maximization of the genotype likelihoods. A likelihood ratio test is performed to assess the statistical significance of a refined potential deletion in a window. Owing to multiple initializations via the initial deletion sizes, PopDel detects deletions of different sizes that overlap the same window. Sample-specific genotype weights guarantee that deletions that are only present in a few or even a single sample can still be detected. The whole process for detecting and refining deletions is described in more detail in Subsection *3.3.3.2*.

Windows containing duplications and inversions are detected separately from the windows that contain deletions. First, the count of read pairs of each orientation is determined for the window in question. If the number of read pairs whose orientations support a duplication (or inversion respectively) reaches a threshold, they are used together with all other read pairs that overlap the window to calculate the genotype likelihoods for the variants in the window. The subsections *3.3.3.3* and *3.3.3.4* further elaborate on these processes.

In a third process, that is still in a very early stage of development, windows containing translocation breakpoints are detected and genotyped. This process does not only use the records stored in the single-chromosome block of the profiles but also those stored in the multi-chromosome block (see Subsection 3.2.4). If enough read pairs with mappings to different chromosomes are detected in a window, the window is examined more closely for the presence of interchromosomal translocations. This process is described more closely in the Subsection *3.3.3.5.*

After a certain number of windows has been processed (default: 200,000) the window-wise calls are combined based on variant types, variant properties and positional distance. This process, described in Subsection 3.3.3.6, yields a single SV call per distinct SV. Finally, the calls are written to the output in VCF 4.3 format.

3.3.2.1 Data Management for Window-wise Calling

For the window-wise calling of SVs, PopDel streams the read pair profiles of all samples and holds the read pair information in an efficient buffering structure. For each sample, it consists of the *start entry table*, the *end entry table* and the *active set*.

Start Entry Table. One line of the start entry table describes one read pair. It consists of the position of the read pair (i.e. the mapping position of its leftmost 3'-end), the read pair's deviation from the sample's median read pair distance, the number of bp clipped from the pair's 3'-ends, as well as the orientation of the pair. Owing to the ordering in which the read pairs are stored in the profiles, the start entry table is automatically sorted according to the positions of the read pairs upon loading the pairs from the profile. No additional sorting is required. The table is implemented as a SeqAn-String. The leftmost table in Figure 3.3 shows an example of a small portion of the start entry table. The information stored for five read pairs is shown.

End Entry Table. One line of the end entry table holds the ID of read pair. The ID of a read pair is simply the index of the record in the cyclic start entry table that identifies the read pair. One line of the end entry table further holds the position of the last window the read pair overlaps. The end entry table is kept sorted in ascending order with respect to these window positions at all times. This is done by not simply appending a new entries to the end of the table, but inserting it directly before the next greater (with respect to the ordering) entry of the table. If no such entry exists, the new entry is appended to the end of the table. The table is implemented as a SeqAn-String. The center table in Figure 3.3 shows an example for the same five read pairs as shown for the start entry table. With $w = 30$ denoting the window size and r_1 as the position of a read pair that is stored in the start entry table, the position of the last windows of this read pair is calculated as $\left\lfloor \frac{r_1+\mu_S+\delta}{w} \right\rfloor \cdot w$.

Active Set. The active set is implemented as an unordered set of unsigned 32-bit integers, which allows for access, removal and addition of elements in constant time on average. The active set holds the IDs of those read pairs that are contained in the currently processed windows. It is updated using the start and end entry tables each time a new window is processed in the window-wise calling: The start entry table tells what elements have to be added to the active set while the end entry table tells what elements have to be removed.

As the window-wise calling proceeds, C++ iterators that point to the current positions in the tables are updated. Each time a new genomic window is processed, the iterator of the start entry table is advanced until the first entry whose position is greater or equal to the starting position of the new window occurs. Then, until the first entry in the table whose positions lies after the new window is encountered, the IDs of the entries are added to the active set. A similar procedure is applied to the end entry table: The iterator pointing to the end entry table advances until the first entry is encountered that holds a position that is greater than the new window position. All IDs in the end entry table are read until the end window position no longer falls into the new window. The IDs read from the end entry table are removed from the active set.

To obtain the information of all read pairs that are active in some window, PopDel simply has to read the IDs that are currently stored in the active set and access the start entry table at the given indices. Since the referenced entries of the start entry table are expected to be located close to each other in main memory, this benefits from cache locality. Different interface functions allow the targeted retrieval of read pair information in the calling and genotyping routines of PopDel, e.g. obtaining only those read pairs that have a specific orientation.

Start Entry Table

i	pos	δ	clip	orientation
0	30	2	0	FR
1	31	0	0	FR
2	45	21	0	FR
3	60	5	0	FR
4	63	0	0	FR
⋮	⋮	⋮	⋮	⋮

End Entry Table

lastWinPos	ID
30	0
30	1
60	3
60	4
60	2
⋮	⋮

State of Active Set

window	ActiveSet
[0,30[	{}
[30,60[	{0,1,2}
[60,90[	{2,3,4}
[90,120[	{}
⋮	⋮

Figure 3.3: Example of data tables and active set in PopDel. The start entry table holds the start positions, deviations from the median read pair distance, the numbers of clipped bases at the 3'-ends and the orientations of the buffered read pairs. The end entry table holds the last window for each entry of the start entry table and the index of the record in the start entry table. For this example the median read pair distance was assumed as 1. The active set holds the IDs of the read pairs that are contained in the respective window. The active set is only present in the state of the currently processed window. The start and end entry tables hold all read pairs of the currently buffered windows (default: 200,000 windows) of one chromosome.

3.3.3 Central Formulas and Algorithms

SVs can be characterized by various factors: breakpoint positions, which also imply their size, variant type (deletion, inversion, duplication, translocation) and their frequency in the cohort. The formulas and algorithms discussed in this section rely on the properties of the SVs described in Section 2.1 to infer the information necessary for the characterization of the SVs.

3.3.3.1 Processing of Insert Size Histograms

The insert size histogram H is based on the insert sizes measured during the sampling of the profiling phase (see 3.2.3). It is important for the assessment of deletions, duplications and inversions. It is stored in the profiles as an array of raw read pair counts per insert size and is transformed when loaded during the calling phase. The aim of the transformation is to let the value stored for each δ_i in H^S reflect the probability of observing a read pair with deviation δ_i from the sample's median insert size, in a window of size w. Let $H^S_{\text{abs}}(x)$ be the function that returns the number of read pairs with insert size x sampled from genome S. For x that lie outside of the sampled range of insert sizes, $H^S_{\text{abs}}(x)$ is defined as 0. In a first step (that can be skipped by the user) PopDel applies a distance based smoothing of the raw counts to gain a more even distribution of insert sizes. It weights each value of $H^S_{\text{abs}}$with its 40 nearest neighbors:

$$H^S_{\text{smooth}}(x) := \frac{\sum_{j=-20}^{20} H^S_{\text{abs}}(x+j) \cdot e^{-\frac{j^2}{40}}}{\sum_{j=-20}^{20} e^{-\frac{j^2}{40}}} \tag{3.2}$$

After the smoothing, PopDel corrects for window size and insert size by applying the following scaling:

$$H^S_{\text{trans}}\left(\delta_i, r\right) := H^S_{\text{smooth}}\left(\delta_i + \mu^S_i\right) \cdot \frac{w + \max\left(\delta_i + \mu^S_i - 2r, 0\right)}{w} \tag{3.3}$$

$\max\left(\delta_i + \mu^S_i - 2r, 0\right)$ reflects the read pair distance of a read pair with insert size $\delta_i + \mu^S_i$ and read length r. Because the histogram was only sampled from read pairs in FR orientation, the calculation of the read pair distance from the insert size does not have to account for other orientations. The reasoning behind the scaling factor is that a read pair is more likely to be observed if it has a larger insert size. This is because the PopDel only considers the space between the 3'-ends of the pair when checking if a read pair is contained in a window (see Subsection 3.3.1). The window size w also influences how many possible locations are there for a read pair to be contained in a window. By adding the read pair distance to the window size, the number of possible positions where a read pair can be observed by PopDel is obtained. Scaling the smoothed histogram with the quotient of this number of positions where a read pair can be observed and the window size consequently normalizes for the window size and accounts for the read pair distance (that depends on the insert size and read length) of the pair.

Unlike $H^S_{\text{abs}}\left(\cdot\right)$, $H^S_{\text{trans}}\left(\cdot\right)$ takes the deviation δ_i from the median insert size μ^S_i as an argument. This is because PopDel operates on these deviations during the calling, and not the absolute insert size. H^S_{trans} is then scaled with the sum of all of its entries to gain a probability density function:

$$H^S\left(\delta_i\right) := \frac{H^S_{\text{trans}}\left(\delta_i\right)}{\sum_d H^S_{\text{trans}}\left(d\right)} \tag{3.4}$$

To avoid later attempts of division by 0 or logarithms of 0, a pseudo-count of $\frac{\max\left(H^S\right)}{500}$ is used in cases where $H^S_{\text{abs}}\left(\delta_i\right)$ would normally return 0.

In later steps, PopDel does not work with the insert sizes of the read pairs but with the *read pair distance*. Since the read pair distance is always measured between the mapped 3'-ends of the reads in a pair, this allows a more consistent handling of read pairs regardless of their orientation. In all subsequent sections, μ_S denotes the median read pair distance of sample S. δ will denote the observed deviation of the read pair distance of some read pair from μ_S. Compatibility between the insert size histogram and the read pair distances during the calling is guaranteed by internal addition of the read lengths to the read pair distance where required.

3.3.3.2 Window-Wise Joint Calling and Genotyping of Deletions

The window-wise detection of deletions has the aim of calculating a likelihood ratio Λ that compares the likelihood of a *deletion model* to a *reference model.* According to Wilk's theorem, likelihood ratio tests qualify for a comparison against the known thresholds of the χ^2-distribution, since the distribution of likelihood ratios is asymptotically χ^2-distributed for large sample sizes[87]. A central condition is that the parameter space of the null hypothesis model must be a subset of the alternative hypothesis model. In PopDel, the null hypothesis states that the window is not affected by a deletion (reference model). The alternative hypothesis states that the window is affected by a deletion of length l (deletion model). PopDel's deletion model relies on the same parameters as the reference model and further introduces the deletion length l as an additional parameter. Consequently, the parameter space of the reference model is a subset of the deletion model and Wilk's theorem can be applied:

$$\Lambda = \frac{\mathcal{L}\left(\text{no del}\right)}{\mathcal{L}\left(\text{del of length } l\right)} \tag{3.5}$$

$$-2\log\Lambda \sim \chi^2$$

Per default, PopDel uses a a p-value threshold of 0.01 (one-tailed) to assess whether a deletion is present in the current window or not. Per Wilk's theorem, the degrees of freedom for testing for significance are equal to the difference in the dimensions of the parameter spaces of the compared models. A discussion on the degrees of freedom can be found at the end of this subsection.

To calculate the likelihoods for the likelihood ratio test, all parameters defining the potential deletion(s) in a window have to be estimated. This is done in an iterative fashion. The window-wise detection of deletions begins with the initialization of the set of sizes of potential deletions that overlap the current window. To this end, PopDel takes the third quartile $Q_3\left(\Delta^S\right)$ of each samples' read pair distance deviations in the window and collects them in a list. The entries of this list are then clustered in a greedy clustering with 50 bp intercluster distance. The mean of each cluster is subsequently compared against a threshold $\tau = 4\sigma_c$ where σ_c is the smallest σ_s of all samples that contributed to the cluster. Mean values passing the threshold are used as initial deletion lengths l^{init}. The pseudocode for this process is described in detail in Algorithm 3.1.

If the clustering yields a list of values for l^{init}, the parameters of the deletion are iteratively refined along with the genotype likelihoods of all samples and the joint deletion likelihood. The likelihood ratio test (Equation 3.5) asses the significance of the variant after the iteration terminates. The iteration relies on the formulas described in the subsequent paragraphs and is performed for each l^{init}.

Algorithm 3.1 Initialization of deletion length estimates

```
 1: buffer = [ ]
 2: for all $S \in \mathcal{S}$ do
 3:     Add $Q_3(\Delta^S)$ to buffer
 4: end for
 5: if buffer is empty then
 6:     No potential deletions in this window
 7: end if
 8: sort buffer in ascending order
 9: initDelLengths = [ ]
10: clusterSum = list[0]
11: clusterSize = 1
12: for i in 1:length(buffer) - 1 do
13:     $\delta_{prev}$= buffer[i-1]
14:     $\delta$ = buffer[i]
15:     if $\delta_{\text{prev}} + 50 > \delta$ then
16:         clusterSum += $\delta$
17:         clusterSize += 1
18:     else
19:         $\tau = 4 \cdot \min\{\sigma_S | \text{for all } S \text{ that contributed to the cluster}\}$
20:         if $\frac{\text{clusterSum}}{\text{clusterSize}} > \tau$ then
21:             Add $\frac{\text{clusterSum}}{\text{clusterSize}}$ to initDelLengths
22:         end if
23:         clusterSum = $\delta$
24:         clusterSize = 1
25:     end if
26: end for
27: $\tau = 4 \cdot \min\{\sigma_S | \text{for all } S \text{ that contributed to the cluster}\}$
28: if $\frac{\text{clusterSum}}{\text{clusterSize}} > \tau$ then
29:     Add $\frac{\text{clusterSum}}{\text{clusterSize}}$ to initDelLengths
30: end if
31: Return initDelLengths
```

Initial Genotype Likelihoods. From the initial value of the deletion length, the initial values of the genotype likelihoods $\mathcal{L}\left(G_g|\Delta^S\right)$ given the observed set of read pair distance deviations Δ of a sample S are calculated. The calculations rely on the density-scaled insert size histograms described in Equation 3.3 and 3.4:

$$\mathcal{L}\left(G_0|\Delta^S\right) = \prod_{\delta \in \Delta^S} H^S\left(\delta\right) \tag{3.6}$$

$$\mathcal{L}\left(G_1|\Delta^S\right) = \prod_{\delta \in \Delta^S} \frac{H^S\left(\delta\right) + H^S\left(\delta - l\right)}{2} \tag{3.7}$$

$$\mathcal{L}\left(G_2|\Delta^S\right) = \prod_{\delta \in \Delta^S} H^S\left(\delta - l\right) \tag{3.8}$$

The rationale behind the equations of the genotype likelihoods are as follows: All read pairs are assumed to be independent of each other. Therefore, the total likelihood of genotype G_g for sample S has to be calculated as the product of the probabilities to observe each individual read pair distance deviation $\delta \in \Delta^S$. As explained in Subsection 3.3.3.1, $H^S\left(\delta\right)$ reflects the probability to observe a read pair with read pair distance δ in the window. For the homozygous reference genotype, one expects all read pairs of the sample in the window to follow the distribution represented by H^S. Consequentially, $\mathcal{L}\left(G_0|\Delta^S\right)$ is calculated as the product of the probabilities returned by $H^S\left(\delta\right)$. As discussed in Subsection 2.1.1, a deletion of size l causes a shift of the distribution of observed read pair distances by l. By adjusting the observed read pair distances by l, the observed read pair distances should again match the distribution represented by H^S. Because a homozygous deletion affects both haplotypes of a genome, all read pairs are expected to follow the shifted distribution. The genotype likelihood for the homozygous deletion model $\mathcal{L}\left(G_2|\Delta^S\right)$ is therefore calculated as the product the probabilities returned by $H^S\left(\delta - l\right)$. In case of a heterozygous genotype, only one haplotype is affected by the deletion. One half of the read pairs are thus expected to originate from the reference haplotype and the other half is expected to originate from the deletion haplotype. The equation for $\mathcal{L}\left(G_1|\Delta^S\right)$ is therefore a mixture of $\mathcal{L}\left(G_0|\Delta^S\right)$ and $\mathcal{L}\left(G_2|\Delta^S\right)$ to equal parts.

Initialization of Allele Frequency. The initial allele frequency f^{init} is calculated by counting the number of read pairs in the current window whose read pair distance deviation roughly matches l^{init} and dividing the count by the number of haplotypes of all samples combined:

$$f^{\text{init}} = \frac{\sum_{S \in \mathcal{S}} \sum_{\delta \in \Delta^S} b_{l,2\sigma_S}(\delta)}{2\,|\mathcal{S}|}, \quad b_{l,2\sigma_s}(\delta) = \begin{cases} 1 & \text{if } \max\left(l - 2\sigma_S, \frac{l}{2}\right) \leq \delta \leq l + 2\sigma_S \\ 0 & \text{else} \end{cases} \tag{3.9}$$

Expected Genotype Frequencies. The initialized frequency estimates give rise to the expected frequencies $\mathcal{L}(G_g, f)$ of the three genotypes G_0 (no variant allele), G_1 (one variant allele / heterozygous variant), G_2 (two variant alleles / homozygous variant) in the cohort. The expected frequencies are derived from the Hardy-Weinberg equilibrium[34], which is discussed more closely in Subsection 4.1.3. The expected frequencies are calculated as:

$$\mathcal{L}(f, G_0) = (1 - f)^2 \tag{3.10}$$

$$\mathcal{L}(f, G_1) = 2f(1 - f) \tag{3.11}$$

$$\mathcal{L}(f, G_2) = f^2 \tag{3.12}$$

Sample-Specific Genotype Weights. The *sample-specific genotype weights* a_g^S are designed to assign a higher weight to genotypes with a good likelihood in a sample and a lower weight to those with a low likelihood in a sample. They are used for the update of the deletion parameters and genotype likelihoods. This allows PopDel to detect deletions that are present at a very low allele frequency in the cohort as long as the signal from the sample(s) carrying the deletion is good. Still, observing a deletion in multiple samples will give additional credibility to the deletion and lead to a higher likelihood. For calculating the weights, the product of the genotype likelihoods of sample S and the expected frequency of genotypes given the observed allele frequency in the cohort is calculated. Then, the ratio of this value and the sum of this product across all three genotypes is calculated:

$$a_g^S = \frac{\mathcal{L}\left(G_g|\Delta^S\right)\mathcal{L}(f, G_g)}{\sum_{j=0}^{2} \mathcal{L}(G_j|\Delta^S)\,\mathcal{L}(f, G_j)} \tag{3.13}$$

Intuitively, this gives a measure of how likely it is to observe genotype G_g given the set of read pair distance deviations in the current window in sample S and the observed allele frequency of the variant allele in the cohort, compared to all possible genotypes and expected genotype frequencies in the cohort.

Sample-Specific Reference Shift. To account for potential biases in the sequencing data, PopDel introduces the sample-specific reference shift ϵ_S. Different biases (e.g. the GC-content of the sequence) can have an influence on the read pair distance of the reads mapped to a window. To mitigate this influence on potential deletion calls, the model allows for a shift of up to $\epsilon_S \leq \sigma_S$ base pairs to maximize the likelihoods of the reference allele(s). Initially ϵ_S is set to 0. Together with the sample-specific genotype weights a_g^S, ϵ_S allows for the calculation of $P_{l,\epsilon_S}^S(\delta)$, which denotes a weighted probability that an observed deviation δ of the read pair distance originates from a distribution shifted by l, rather than ϵ_S:

$$P_{l,\epsilon_S}^S(\delta) = a_1^S \frac{H^S(\delta - l)}{H^S(\delta - \epsilon_S) + H(\delta - l)} + a_2^S \tag{3.14}$$

For each sample, ϵ_S is then updated by weighting the observed read pair distance deviations with $P_{l,\epsilon_S}^S(\delta)$ and calculating the ratio to the sum of all $P_{l,\epsilon_S}^S(\delta)$:

$$\epsilon_S^{\text{new}} = \frac{\sum_{\delta \in \Delta^S} \delta \cdot P_{l,\epsilon_S}(\delta)}{\sum_{\delta \in \Delta^S} P_{l,\epsilon_S}(\delta)} \tag{3.15}$$

If ϵ_S becomes greater than σ_S, it is set to 0 to avoid the distortion of the length estimates for homozygous deletions, as observed in early versions of PopDel.

Update of Deletion Length. The deletion length estimate l is updated in a similar fashion to the update of the sample-specific reference shift. This is because both the reference shift and the deletion length cause a shift of the observed read pair distances from the expected distributions (see Figure 2.1b). The deletion length is estimated using read pairs of all samples in the cohort. It is updated by extending Equation 3.15 with a sum across all samples to achieve a joint calculation that considers all observed read pair distance deviations of all samples in the window:

$$l^{\text{new}} = \frac{\sum_{S \in \mathcal{S}} \sum_{\delta \in \Delta^S} \delta P_{l,\epsilon_S}(\delta)}{\sum_{S \in \mathcal{S}} \sum_{\delta \in \Delta^S} P_{l,\epsilon_S}(\delta)} \tag{3.16}$$

Update of Allele Frequency. The allele frequency is updated using the sample-specific genotype weights:

$$f^{\text{new}} = \frac{1}{2|\mathcal{S}|} \sum_{S \in \mathcal{S}} \left(a_1^S + 2a_2^S\right) \tag{3.17}$$

Because a_2^S represents the likelihood of observing a homozygous deletion given the data and the expected genotype frequencies, it is counted with a weight of two (corresponding to the two affected alleles). a_1^S corresponds to likelihood of observing a heterozygous deletion given the data and the expected genotype frequencies. Since a heterozygous deletion only affects one allele, a_1^S is only counted with a weight of one. Dividing the weighted sum of sample-specific genotype weights across all samples by the total number of alleles in the cohort (i.e. $2|\mathcal{S}|$) yields the new allele frequency.

Update of Genotype Likelihoods. Once all parameters of the potential deletion have been updated, the genotype likelihoods are updated. The update is conceptionally very close to the initialization of genotype likelihoods (Equations 3.6 - 3.20), but also considers the sample-specific reference shift ϵ_S for the reference alleles:

$$\mathcal{L}\left(G_0|\Delta^S\right) = \prod_{\delta\in\Delta^S} H^S\left(\delta - \epsilon_S\right) \tag{3.18}$$

$$\mathcal{L}\left(G_1|\Delta^S\right) = \prod_{\delta\in\Delta^S} \frac{H^S\left(\delta - \epsilon_S\right) + H^S\left(\delta - l\right)}{2} \tag{3.19}$$

$$\mathcal{L}\left(G_2|\Delta^S\right) = \prod_{\delta\in\Delta^S} H^S\left(\delta - l\right) \tag{3.20}$$

Reference Model and Deletion Model. The updated genotype likelihoods and the sample-specific genotype weights together enable the calculation of the deletion model and reference model for the likelihood ratio test in Equation 3.5. With the prior probability π of observing a deletion (default value of 10^{-4}) the models are set up as:

$$\mathcal{L}\left(\text{no del}\right) = (1-\pi)\prod_{S\in\mathcal{S}} \mathcal{L}\left(G_0|\Delta^S\right) \tag{3.21}$$

$$\mathcal{L}\left(\text{del of length } l\right) = \pi\prod_{S\in\mathcal{S}}\sum_{g=0}^{2} a_g^S \mathcal{L}\left(G_g|\Delta^S\right) \tag{3.22}$$

Termination of Iteration. For each value in the list of initial deletion lengths the above calculations are repeated until one or more of the following termination conditions are met:

- the newly calculated combination of deletion length and frequency has been observed in a previous iteration
- the maximum number of iterations (default 15) is exceeded
- f^{new} drops below a threshold (default 10^{-10})
- l^{new} drops below a threshold (default 95 %-quantile of $4\sigma_S$ for all $S\in\mathcal{S}$)

The potential deletion is discarded for the current window if one of the latter two cases applies. A special case occurs when the iteration starts alternating back and forth between two sets of deletion parameters. In this case, the set of parameters yielding the higher likelihood ratio is chosen.

Breakpoint Estimation. After the termination of the iteration, PopDel estimates the start and end positions of the variant in the reference. This is done by collecting the mapping positions of the (potentially clipped) 3'-ends of all read pairs that support the variant in a sorted list. A read pair with read pair distance deviation δ overlapping the window is considered to support a deletion of size l if δ lies within the boundaries given by l and the sample's distribution of read pair distances as defined in Equation 3.23:

$$l - Q_{99}\left(H^S\right) \leq \delta \leq l + Q_{99}\left(H^S\right) \tag{3.23}$$

where $Q_{99}\left(H^S\right)$ is the 99% quantile of the sample's read pair distance histogram.

From the sorted list of 3'-end positions of supporting read pairs the 80% quantile is calculated and taken as the start position of the variant. The reasoning behind this approach is that in the presence of a deletion the 3'-end of the reads located upstream but still overlapping the breakpoint can be clipped. This causes the affected reads to have the same clipped 3'-mapping location directly before the deleted sequence. By taking the 80% quantile instead of the maximum, the effect of outliers is compensated. Evaluations on simulated data have shown that this approach is on par with other SV callers that explicitly consider split-read alignments (see Paragraph Breakpoint Precision in Subsection 4.3.3.1).

The end position of a deletion is calculated in a similar way. Instead of the reads that are located upstream of the deletion, the downstream reads are considered. Like before, the (potentially soft-clipped) 3'-end mapping positions of the downstream reads of the pairs that support the deletion are collected. Instead of the 80% quantile, the 20% quantile is calculated and taken as the end position of the variant.

The range given by the start and end position estimates does not necessarily match the previously estimated deletion length l exactly. Therefore, if all downstream reads of the supporting read pairs are found to have no clipping at their 3'-ends and the estimated end position does not match the variant start position plus $l \pm \sigma_s$, PopDel falls back to calculate the end position of the variant as its start position plus l. A shortcoming of this approach for the estimation of the positions of the variant is the reliance on the exactness of the mapping positions of the read pairs. In the presence of repeats at the breakpoints, they easily become imprecise, resulting in deviations of the estimates.

Handling of Multiple Read Groups per Sample. The BAM format allows a single sample to consist of multiple read groups. Read groups can originate from different sequencing libraries with different properties. This can cause differences in the distribution of read pair distances PopDel uses for the detection of deletions as described above.

For working with multiple read groups PopDel creates a distinct insert size histogram for each read group of a sample. All histograms are stored in the read pair profile of the sample.

During the joint calling process all read pairs are analyzed based on the read group they are assigned to. This means that the read pair distance of a read pair from read group 'A' will be evaluated based on the histogram of the read group 'A'. All other parameters related to the distribution of read pair distances are also calculated and handled based on the respective read group.

This changes the calculation of the deletion and reference model (see Equations 3.22 and 3.21 for comparison) as follows:

$$\mathcal{L}\left(\text{del of length } l\right) = \pi \prod_{S\in\mathcal{S}} \prod_{R\in\mathcal{R}^{S}} \sum_{g=0}^{2} a_g^R \mathcal{L}\left(G_g|\Delta^R\right) \tag{3.24}$$

$$\mathcal{L}\left(\text{no del}\right) = (1-\pi) \prod_{S\in\mathcal{S}} \prod_{R\in\mathcal{R}^{S}} \mathcal{L}\left(G_0|\Delta^R\right) \tag{3.25}$$

with $\mathcal{R}^{\mathcal{S}}$as the set of all read group in sample S and Δ^R denoting the read pair distance deviations of read group R in the current window.

The genotype likelihoods (see Equations 3.18-3.20 for comparison) are now calculated as:

$$\mathcal{L}\left(G_0|\Delta^R\right) = \prod_{\delta\in\Delta^R} H^R\left(\delta-\epsilon_R\right) \tag{3.26}$$

$$\mathcal{L}\left(G_1|\Delta^R\right) = \prod_{\delta\in\Delta^R} \frac{H^R\left(\delta-\epsilon_R\right) + H^R\left(\delta-l\right)}{2} \tag{3.27}$$

$$\mathcal{L}\left(G_2|\Delta^R\right) = \prod_{\delta\in\Delta^R} H^R\left(\delta-l\right) \tag{3.28}$$

Here H^R stands for the transformed and density-scaled histogram of insert sizes for read group R.

The sample-specific genotype weights (Equation 3.13) become read group and genotype-specific weights:

$$a_g^R = \frac{\mathcal{L}\left(G_g|\Delta^R\right)\mathcal{L}\left(f,G_g\right)}{\sum_{j=0}^{2}\mathcal{L}\left(G_j|\Delta^R\right)\mathcal{L}\left(f,G_j\right)} \tag{3.29}$$

Consequently, the update of the allele frequency (Equation 3.17) and deletion length (Equation 3.31) are also adapted:

$$f^{\text{new}} = \frac{1}{2\left|\mathcal{S}\right|} \sum_{S\in\mathcal{S}} \sum_{R\in\mathcal{R}^{S}} \left(a_1^R + 2a_2^R\right) \tag{3.30}$$

$$l^{\text{new}} = \frac{\sum_{S\in\mathcal{S}}\sum_{R\in\mathcal{R}^S}\sum_{\delta\in\Delta^S}\delta P^R_{l,\epsilon_R}(\delta)}{\sum_{S\in\mathcal{S}}\sum_{R\in\mathcal{R}^S}\sum_{\delta\in\Delta^S}P^R_{l,\epsilon_R}(\delta)} \tag{3.31}$$

with

$$P^R_{l,\epsilon_R}(\delta) = a^R_1 \frac{H^R(\delta - l)}{H^R(\delta - \epsilon_R) + H(\delta - l)} + a^R_2 \tag{3.32}$$

Discussion on Degrees of Freedom. As mentioned when introducing the likelihood ratio in Equation 3.5, the degrees of freedom one has to use when comparing $-2\log\Lambda$ against the thresholds of the χ^2-distribution are the difference in the dimensions of the parameter spaces of the deletion model and the reference model. In the main publication presenting PopDel[67], my co-authors and I assumed the deletion length to be the only parameter discerning the deletion model from the reference model, resulting in the choice of one degree of freedom.

After further consideration, I now argue that the likelihood of the deletion model does in fact not only depend on the size of the deletion but also on its allele frequency in the data. This makes two degrees of freedom the appropriate choice. As can be seen when comparing the equations for the two models (Equation 3.21 and 3.22), the deletion model depends not only on the genotype likelihoods (Equation 3.18 - 3.20), but also on the sample-specific genotype weights a^S_g. As described in Equation 3.13, the weights depend on the expected genotype frequencies $\mathcal{L}(f, G_g)$, which in turn are calculated based on the frequency f, which has been initialized based on the data (Equation 3.9).

An argument against the consideration of f as an individual parameter was that its estimation is heavily based on the deletion length l. I argue here that this ostensible dependency is just a side effect of the iterative nature of the approach applied by PopDel for estimating the properties of a deletion. It naturally introduces a dependency of all variables on each other in the equations because already known and refined values of one variable (e.g. l) are used to further refine the estimate for another variable (e.g. f). This circular dependency in the formulas does not change the circumstance that deletion length and frequency are separate properties of the data. The actual frequency of a deletion in the data is not determined by its size or vice versa. Therefore, the necessity to estimate the allele frequency of a deletion from the data adds to the uncertainty of the deletion model. Thus, an additional degree of freedom is required when comparing $-2\log\Lambda$ against the thresholds of the χ^2-distribution.

3.3.3.3 Window-Wise Joint Calling and Genotyping of Duplications

PopDel's joint calling and genotyping of duplications is currently implemented as a prototype and is still subject to improvements. The data from all samples in the cohort is used jointly to determine the breakpoints of a potential duplication, but currently no joint likelihood for the overall presence of a duplication is calculated like done in Equation 3.5 for deletions. If a duplication has been determined using the data from all samples, each sample is genotyped for this jointly determined duplication. This is done by calculating a window-wise genotype likelihood for each sample.

Initialization. In a first step, the read pairs that overlap the current window are counted and classified by their orientation. Let $\left|o^S\right|$ be the number of read pairs of sample S that overlap the current window and have orientation o, with $o \in \{\mathrm{FR}, \mathrm{RF}, \mathrm{FF}, \mathrm{RR}\}$. As described in Subsection 2.1.2, duplications are indicated by the presence of read pairs in RF orientation. PopDel requires at least one sample to have at least 10 % (and an absolute minimum of 2 read pairs) of its read pairs that overlap the current window in RF orientation to initialize the duplication calling for this window:

$$\left|\mathrm{RF}^S\right| \geq 2 \tag{3.33}$$

$$\frac{\left|\mathrm{RF}^S\right|}{\sum_o \left|o^S\right|} \geq 0.1 \tag{3.34}$$

Start and End Point Estimation. If a sample passes the above thresholds, the start and end point of the potential duplication are calculated. This is done using an approach similar to the position estimation for deletions. A list of the (potentially soft-clipped) 3'-end mapping positions of the upstream read pairs that support the duplication is created and sorted. The list consists of read pairs from all samples. A read pair is considered to support the duplication if it has RF orientation and is overlapping the current window. Further criteria like a consistent read pair distance of all supporting read pairs (reminiscent to Equation 3.23) are to be considered in future implementations. Since the supporting upstream reads are in reverse orientation, the 20 % quantile of the list is taken as the starting point of the duplication. The end position is estimated analogously by creating the sorted of (potentially soft-clipped) 3'-end positions of supporting read pairs and taking the 80 % quantile.

Sample-Wise Genotype Likelihoods. As a tandem duplication only introduces new junctions, even a sample that is a homozygous carrier of the tandem duplication carries read pairs that support the reference haplotype. The zygosity of the duplication however changes the ratio of pairs that support the reference or duplication. Observations of the ratio of read pairs that support the reference allele to read pairs that support the variant allele on simulated data led to the following empirical estimates for the expected ratios: For a heterozygous carrier, approximately 50 % of the read pairs were observed to support the reference and 50 % were observed to support the duplication. In a homozygous carrier, the expected ratio of reference supporting read pairs to duplication supporting read pairs is 1/3 and 2/3. Notably, these observed ratios deviate from theoretical expectations, which will be discussed in the next paragraph. The currently implement estimation of the sampe-wise genotype likelihoods is based on these observed ratios. The estimations uses all read pairs of any orientation that overlap the window:

$$\mathcal{L}\left(G_0|\Delta^S\right) = \prod_{\delta\in\Delta^S} H^S\left(\delta\right) \tag{3.35}$$

$$\mathcal{L}\left(G_1|\Delta^S\right) = \prod_{\delta\in\Delta^S} \frac{H^S\left(\delta\right) + H^S\left(\delta - l\right)}{2} \tag{3.36}$$

$$\mathcal{L}\left(G_2|\Delta^S\right) = \prod_{\delta\in\Delta^S} \frac{1}{3}H^S\left(\delta\right) + \frac{2}{3}H^S\left(\delta - l\right) \tag{3.37}$$

Where l is calculated as the absolute difference between the start and end positions of the potential duplication that have been calculated from the quantiles of the sorted lists of 3'-end positions in the previous step.

In the current implementation of the duplication genotyping, read pairs that have an increased read pair distance due to the presence of another variant in another sample could also lead to a high genotype likelihood $\mathcal{L}\left(G_1|\Delta^S\right)$ or $\mathcal{L}\left(G_2|\Delta^S\right)$ for a duplication, if the variant sizes are similar and the genotyping has been initiated due to the presence of RF oriented read pairs. In future implementations that include the calculation of a joint duplication likelihood, this will be addressed by not only considering the read pair distances when calculating the genotype likelihoods but also the orientation of the read pairs. Another shortcoming of the current model is that the adjustment of δ by l does not account for the distance of the reads to the start and end positions of the duplication and potential soft-clipping, which both influence the increase of the read pair distance. For large duplication and low read pair distances, the model is dominated by the influence of l. For low l and larger read pair distances however, the inaccuracy becomes larger. An approach for the appropriate adjustment of δ is presented for the genotyping of inversions in Subsection 3.3.3.4, but not yet implemented for duplications.

Discussion on the Deviation from the Theoretically Expected Ratio of Duplication Supporting Read Pairs. When only considering read pairs that map inside of the duplicated sequence, i.e. excluding the windows that contain the novel junction, one theoretically expects the following ratios of read pairs in a window: On a single duplication haplotype (and therefore also in homozygous carriers of the duplication), $\frac{2}{3}$ of the read pairs that overlap a window are in FR orientation and therefore support the reference. The remaining $\frac{1}{3}$ of the read pairs that overlap the window are in RF orientation, since they overlapped the novel junction of the duplication in the sequenced genome (see Subsection 2.1.2). The reason for the higher number of FR read pairs lies in the inserted duplicated sequence: Read pairs that originate form the inserted duplicated sequence in the sample genome are mapped to the original sequence in the reference genome. In contrast, since only one novel junction occurs between the duplicated sequences in the sample genome, the amount of read pairs that overlap the junction is not increased. This leads to the described ratio. For heterozygous duplications, the FR read pairs from the non-variant haplotype also map to the location of the duplicated sequence. The additional FR read pairs thereby change the share of expected FR read pairs to $\frac{3}{4}$ and the share of RF read pairs to $\frac{1}{4}$.

Window-wise observations based on simulated data however suggested a ratio of approximately 50 % FR read pairs and 50 % RF read pairs for heterozygous carriers of a duplication. For homozygous carriers, approximately 33 % FR read pairs 66 % RR read pairs where observed. Due to the small insert size of the simulated data, the absolute number of observed FR read pairs in a window further showed considerable fluctuations. The sources of this discrepancy between the theoretically expected and observed ratios of FR read pairs and RF read pairs are currently unknown and require further investigation. Window sizes larger than the currently used 30 bp for the reduction of the fluctuation of the observed number of FR read pairs in a window, and a targeted genotyping only based on the windows that contain the breakpoint of a duplication are two possible approaches that are currently considered to this end.

3.3.3.4 Window-Wise Joint Calling and Genotyping of Inversions

PopDel's joint calling and genotyping of inversions is currently implemented as a prototype and is still subject to improvements. The same limitations as mentioned for the detection of duplications in Subsection 3.3.3.3 hold.

As described in Subsection 2.7, inversions are indicated by read pairs in FF orientation as well as read pairs in RR orientation. PopDel considers read pairs in FF and RR orientation separately. It can therefore detect and genotype the same inversions from both sources of evidence independently. This typically causes the same inversion to be detected twice.

The window-wise detection of inversions calculates a genotype likelihood for each of the three possible genotypes for each sample in the cohort. The approach follows the same scheme as for duplications (see Subsection 3.3.3.3). The approach will be described for read pairs in FF orientation only, but RR read pairs are treated analogously.

Initialization. In a first step, the read pairs are compared against the same thresholds given in Equation 3.33 and 3.34, except that they are applied for read pairs in FF or RR orientation.

Start and End Point Estimation. If a sample passes the thresholds, the start and end points of the potential inversion are calculated. Similar to the process described for duplications, the sorted list of 3'-end mapping positions of all read pairs in FF orientation is generated using all samples and the borders of the variant are determined from the quantiles of the positions. For read pairs in FF orientation, this means that the starting position of the inversion is estimated from the 80 % quantile of the 3' positions of the upstream reads. The end position is estimated from the 80 % quantile of the 3'-end positions of the downstream reads. For read pairs in RR orientation, both positions are estimated using the 20 % quantiles. The start and end point estimation currently does not account for varying breakpoints at the left and right novel junctions of the inversion, like often caused by micro-deletions. The explicit characterization of the four possible breakpoints is left for future implementations of PopDel.

Breakpoint-Spanning Read Pairs. For inversions, only the read pairs that overlap the breakpoints of the variant provide a good signature for genotyping. Other than copy number variants like deletions or duplications, the read depth of read pairs that do not overlap the breakpoint are not expected to change. Therefore, the sample-wise genotype likelihoods are only calculated from read pairs that have one read located outside of the inverted region and on read within it. To account for soft-clipping or inaccuracies in the location of the breakpoints or the mapping locations of the reads, a tolerance of 50 % of the read length is applied, depending on the orientation of the read pair. Formally, let a and b denote the 3'-end mapping locations of the upstream and downstream read of a read pair p. Let further L denote the left breakpoint of the inversion and let R denote the right breakpoint of the inversion. Let r denote the length of a read in the pair. A read pair is considered to overlap only the the inversion breakpoint L if

$$\text{overlap}\,(a,b,L,R) = \begin{cases} a-\frac{r}{2} \leq L < b+\frac{r}{2} < R & \text{if orientation}\,(p) == \text{FR} \\ a \leq L < b < R & \text{if orientation}\,(p) == \text{RF} \\ a-\frac{r}{2} \leq L < b-\frac{r}{2} < R & \text{if orientation}\,(p) == \text{FF} \\ L \leq a+\frac{r}{2} < R < b+\frac{r}{2} & \text{if orientation}\,(p) == \text{RR} \end{cases} \tag{3.38}$$

For the overlap of R the conditions are define analogously. The breakpoint overlap is also defined for read pairs in RF orientation, even though read pairs in RF orientation are not a signal for inversions. This is because the sample-wise genotype likelihoods described below consider all read pairs of all orientations in the window. Read pairs in RF orientation are interpreted as a signal against the presence of an inversion, just like FR oriented read pairs.

Sample-Wise Genotype Likelihoods. Let $\mathcal{R}^S_{\text{span}}$ be the set of all read pairs of sample S in the window that fulfill any of the conditions in Equation 3.38 and therefore span one breakpoint. On an inversion haplotype, all read pairs that span a breakpoint are expected to be in FF or RR orientation and have an increased read pair distance. The exact increase depends on the the size of the inversion, the distance of the reads to the breakpoints and potential clipping of the reads. Let $\text{ldist}(r, L) = |L - \text{pos}(r_1)|$ denote the absolute distance in bp of the 3'-end mapping position of the upstream read of read pair r to breakpoint L. Let analogously $\text{rdist}(r, R) = |R - \text{pos}(r_2)|$ denote the absolute distance of the 3'-end mapping location of the downstream read of r to the right breakpoint R. Let $\text{clip}(r)$ be the function that returns the number of clipped bases at any 3'-end of r. With μ_S as the median read pair distance of sample S, the *breakpoint-adjusted read pair distance deviation* $\delta_{\text{adj}}(r)$ of a read pair r in FF or RR orientation is then calculated as:

$$\delta_{\text{adj}}(r) = \text{ldist}(r, L) + \text{rdist}(r, R) - \mu_S - \text{clip}(r) \tag{3.39}$$

For read pairs in FR or RF orientation $\delta_{\text{adj}}(r)$ is not defined.

For the calculation of the sample-wise genotype likelihoods of an inversion let $\delta(r)$ denote the *unadjusted* read pair distance of read pair r. On a haplotype that does not carry the inversion, all read pairs are expected to follow the sample distribution of read pair distances. On a haplotype that carries the inversion, all read pairs that overlap a breakpoint of the inversion are expected to be either in FF or RR orientation and thus require adjustment like described above. Therefore, the sample-wise genotype likelihoods for each of the preliminary inversion are calculated from the breakpoint spanning read pairs as:

$$\mathcal{L}\left(G_0 | R^S_{\text{span}}\right) = \prod_{r \in R^S_{\text{span}}} H^S(\delta) \tag{3.40}$$

$$\mathcal{L}\left(G_1 | R^S_{\text{span}}\right) = \prod_{r \in R^S_{\text{span}}} \frac{H^S(\delta) + H^S(\delta_{\text{adj}}(r))}{2} \tag{3.41}$$

$$\mathcal{L}\left(G_2 | R^S_{\text{span}}\right) = \prod_{r \in R^S_{\text{span}}} H^S(\delta_{\text{adj}}(r)) \tag{3.42}$$

Since $\delta_{\text{adj}}(r)$ is not defined for read pairs in FR or RF orientation, $H^S(\delta_{\text{adj}}(r))$ returns a pseudo-count probability of $1 \cdot 10^{-5}$ for those pairs. This procedure further prevents that read pairs that exhibit an increased read pair distance due to a deletion or duplication (see Subsection 2.1.1 and 2.1.2) are mistaken as evidence of an inversion.

3.3.3.5 Window-Wise Joint Calling of Translocations

PopDel's detection and genotyping of translocations is currently implemented as a prototype in a very early, conceptual phase. While that data of all samples is used for the calculation of the translocation breakpoints, no joint likelihood for the overall presence of a translocation is calculated. Further, the processing of translocations has not been tested yet due to bugs in the current implementation that prevent it from running on multiple samples. The following description of the detection and genotyping of translocations is therefore to be understood as a draft of a possible approach that still requires substantial improvements and testing.

The window-wise detection of translocations calculates a genotype likelihood for each of the three possible genotypes for each sample in the cohort. Each sample is genotyped for all translocations that are detected in any of the samples of the cohort in the window.

Generation of Chromosomal Cluster. In a first step, the count of read pairs in the window that show signs of an interchromosomal translocation is assessed for each sample. A read pair is considered to show signs of a translocation if both of its reads map to different chromosomes. At least 10 % of the read pairs of a sample - but more than one - must show signs of an interchromosomal translocation for the sample to be further considered in the current window. If this is the case, the translocation read pairs are clustered by the *remote chromosome,* the chromosome the other read in the pair has been mapped to. The read pairs of each chromosomal cluster are then separated by their orientation. The count of read pairs of at least one orientation has to pass the same thresholds as before for the cluster to be marked as *passing* for subsequent steps.

Generation of Positional Subclusters. In a next step, all chromosomal clusters of each sample that had at least one of its clusters passing the threshold after being separated by read pair orientation are inspected together. Each cluster (that has already been separated by orientation of the read pairs) is divided into further subclusters based on the positions where the reads map on the remote chromosome. If two reads map more than $3\sigma_S$ apart on the remote chromosome, they are not considered to support the same translocation and are therefore assigned to different clusters. The clustering follows a simple greedy scheme: The reads are first sorted by position and the first subcluster is initialized from the first read in the list. New reads are added to the subcluster as long as they are close enough to the first read pair of the subcluster. If a read is encountered whose distance is too big, a new subcluster is created.

Filtering of Clusters. The number of positional subclusters per read pair orientation is counted for all chromosomal clusters of the sample. These sums of chromosomal and positional clusters per orientation are then used for checking the following conditions to assess the presence of too many contradicting clusters in the sample:

a) more than one chromosomal cluster has been marked as *passing* after the clustering by chromosome

b) there are more than two chromosomal and positional clusters for any orientation

c) there is a chromosomal and positional cluster of reads in FF orientation or RR orientation and at the same time a chromosomal and positional cluster of reads in FR orientation or RF

If any of the above conditions are met, the sample is marked as *noisy* for the current window and not further considered in subsequent steps.

Determination of Remote Chromosome. After the chromosomal and positional clusters have been filtered for all samples, the clusters of all samples are processed together. They are used to create a set of remote chromosomes by collecting the remote chromosomes of all chromosomal clusters that were marked as *passing* from samples that were not marked as *noisy*. For each remote chromosome and orientation, the 3'-end mapping positions of the read pairs in the biggest positional subcluster are collected and stored per sample in the lists P^o_{local} and P^o_{remote}, depending on whether they map to the current or remote chromosome. Here, $o \in \{\text{FR, RF, FF, RR}\}$ denotes the orientation of the read pairs in a cluster. The size of all positional subclusters for all orientations is summed up per sample and used for quantifying the support for the translocation in each sample. Based on the most numerous orientation in the positional subclusters, the preliminary translocation is assigned an orientation which will determine what read pairs are considered in the subsequent steps.

Estimation of Local and Remote Position. Next, the sample-wise positional estimates for the translocation breakpoints are calculated from the subclusters. As well the position on the current chromosome p_{local} as the position on the remote chromosome p_{remote} have to be estimated for each sample. For each orientation o, one local position estimate is calculated as the q % quantile Q_q of P^o_{local} of the sample:

$$p^o_{\text{local}} = Q_q\left(P^o_{\text{local}}\right), q = \begin{cases} 80 & \text{if } o == \text{FR} \\ 20 & \text{if } o == \text{RF} \\ 80 & \text{if } o == \text{FF} \\ 20 & \text{if } o == \text{RR} \end{cases} \tag{3.43}$$

The remote position estimates of the sample are calculated similarly from different quantiles:

$$p^o_{\text{remote}} = Q_q\left(P^o_{\text{remote}}\right), q = \begin{cases} 20 & \text{if } o == \text{FR} \\ 80 & \text{if } o == \text{RF} \\ 80 & \text{if } o == \text{FF} \\ 20 & \text{if } o == \text{RR} \end{cases} \tag{3.44}$$

After the local and remote positional estimates for all samples have been calculated, they are used together to get the final positional estimate for the translocation. The median across all samples for p^o_{local} and p^o_{remote} are calculated. Let median(p^o_{local}) denote this median for the local positions across all samples and median(p^o_{local}) for the remote positions respectively. The final local position of the translocation is calculated as the weighted mean of median(p^{FR}_{local}) and median(p^{RF}_{local}) if the translocation has been assigned an FR or RF orientation, or from median(p^{FF}_{local}) and median(p^{RR}_{local}) otherwise. The weights are the numbers of positions that contributed to the calculation of the medians across all samples. This process currently does not account for varying breakpoints on the local and remote chromosomes, like for example caused by micro-deletions at the breakpoints. Accounting for all breakpoint requires a calling and genotyping of both breakpoints on both involved chromosomes, resulting in a total of four breakpoints per translocation event. This individual characterization of all breakpoints is left fort future implementations of PopDel.

Once the local and remote positions of the translocation have been calculated, it is given to a buffer and stored along with the sample-wise local and remote positions per orientation and the sample counts of the cluster sizes per orientation and sample. They are later used in the step for the combination of windows (see Subsection 3.3.3.6) to calculate the likelihoods for the genotypes of all samples.

3.3.3.6 Combination of Windows Representing the Same SV

The window-wise calling of SVs results in a single SV being called for multiple windows. The number of windows depends on the size of the variant. To avoid redundancy of the output, those window-wise calls are combined. Each distinct SV is then written as a single line in the final VCF output.

The window-wise calling is performed until a defined number of windows (default 200,000 windows) has been processed an buffered. Afterwards, the buffered windows are checked for their eligibility for being combined with other windows in the buffer. The criteria for the combination of windows and the means of combining the windows are different for the different SV classes and are described in the subsequent paragraphs.The combination of windows for the different SV classes is performed independently.

Deletions. The deletion calls of all buffered windows are sorted in ascending order according to their positions on the reference, deletion length and likelihood ratio as first, second and third sorting key. Let w_i be the first of n windows for which a deletion call has been made in the window-wise deletion calling, i.e. the deletion likelihood ratio Λ passed the threshold. Let further w_{i+k} be another such window with $0 < k < n$. Let l_i and l_{i+k} denote the deletion length estimates of w_i and w_{i+k}. Let r_x denote the range that is spanned by the breakpoints of the deletion in w_x. w_i is compared with w_{i+k} for sufficient similarity. Sufficient similarity is defined as fulfilling the following criteria:

a) $|l_i - l_{i+k}| \leq \max\left(\frac{\min(l_i, l_{i+k})}{2}, 2\overline{\sigma}\right)$, with $\overline{\sigma} := \frac{\sum_{S \in \mathcal{S}} \sigma_S}{|\mathcal{S}|}$

b) r_i and r_{i+k} overlap by at least $\min\left(\frac{\min(r_i, r_{i+k})}{4}, \min(r_i, r_{i+k}) - 2\overline{\sigma}\right)$

If both criteria are satisfied, w_i and w_{i+k} are marked for combination. If only condition a) is satisfied, the windows are only marked for combination if the deletion was likely interrupted. To this end, the following condition is tested to check if an interruption is leading to a mismatch of the estimated deletion length and the range spanned by r_x:

c) $(|r_i| < l_i \vee |r_{i+k}| < l_{i+k}) \wedge \text{delStart}(w_{i+k}) - \text{delStart}(w_i) < \min(l_i, l_{i+k}) + 4\bar{\sigma}$

Here, $|r_x|$ denotes the length of the deletion derived from the spanned range r_x in contrast to l_x that reflects the deletion length estimated during the window-wise calling.

If both a) and c) are met, the end breakpoint in w_i is updated to match that of w_{i+k} and the windows are marked for combination.

The checks are performed for increasing k until a window $w_{i+k'}$ is encountered that does not meet the above conditions. Afterwards, all windows between w_i and $w_{i+k'}$ (including w_i but not $w_{i+k'}$) are combined. The combined deletion length is calculated from the median of all deletion length estimates of the combined windows. The combined starting position of the deletion is calculated from the median of the starting position estimates of the combined windows. Further, the genotype likelihoods of the samples are recalculated so that only windows that lie within the borders of the deletion are considered. The genotype likelihoods of the same sample for the combined windows are not independent because they have been calculated using the same read pairs that overlap the deletion. Consequentially, the genotype likelihood of a sample in the combined windows cannot be integrated by multiplying them. As a pragmatic resort, the harmonic mean of the sample-wise genotype likelihoods of the windows is taken instead as a representative for the combined windows. After the genotype likelihoods have been recalculated for all samples, the allele frequencies of the deletion in the cohort is estimated from the allele counts as inferred from the genotypes of the samples. The genotypes are determined from the best genotype likelihood of each sample.

After the windows between w_i and $w_{i+k'}$ have been combined, the process is repeated starting from the first window that has not been combined $w_{i+k'}$ and the subsequent windows. Once the process reaches the end of the sorted list of windows, all windows that represent the same deletion have been combined.

Duplications and Inversions. The combination of windows containing duplications and inversions is performed separately from each other in PopDel but since they share a similar approach, they both will be explained on the example of duplications. Differences when combining windows of inversions are highlighted at the end of the paragraph.

The duplication calls of all buffered windows are sorted in ascending order according to the mapping positions of their left and right breakpoint as a first and second sorting key and to the position of the window as a third sorting key. Subsequently, a scheme similar to the combination of deletion windows is applied. Therefore, the same notation is used here. The following conditions must both be fulfilled for marking two windows w_i and w_{i+k} as eligible for combination:

a) the left breakpoints of the duplications in w_i and w_{i+k} are located within $4\bar{\sigma}$ of each other

b) the right breakpoints of the duplications in w_i and w_{i+k} are located within $4\bar{\sigma}$ of each other

Like for deletions, the check for similarity is performed for increasing k until the first window $w_{i+k'}$ occurs that does not meet one or both of the above conditions. Then, the combination of all windows marked for combination commences.

The combined starting position of the duplication is calculated from the 20 % quantile of the starting position estimates of the combined windows, while the combined end position is calculated from the 80 % quantile of the end position estimates of the combined windows. For each sample, the genotype likelihoods of the windows that lie between the breakpoints of the duplication are combined by calculating their harmonic mean. The reasoning behind this is the same as for deletion windows. Subsequently, the genotypes of all samples are set according to their best genotype likelihood and the cohort-wide allele frequency of the variant is set by counting the variant alleles as given by the genotypes.

After the windows between w_i and $w_{i+k'}$ have been combined, the process is repeated starting from the first window that has not been combined $w_{i+k'}$ and the subsequent windows. Once the process reaches the end of the sorted list of windows, all windows that represent the same duplication have been combined.

For inversions, the main differences are:

- Window-wise calls from read pairs in FF orientation are treated separately from window-wise call from read pairs in RR orientation.
- Only those windows that contain the breakpoints of the inversion contribute to the combination of sample-wise genotype likelihoods via the harmonic mean.

This approach for inversions leads to each inversion being detected twice, assuming a clear signature from read pairs in either FF or RR orientation. Combining the junctions implied by read pairs in FF and RR orientation into single events is considered for a future implementation.

Translocations. The combination of windows that passed the window-wise translocation calling described in Subsection 3.3.3.5 follows a different approach than the combination of windows for the other SV types. This is because the results of the window-wise translocation calling are less final than the results of the other SV types. The genotypes of the translocations have yet to be calculated from the combined evidence across multiple windows. This is possible in an efficient manner even for larger numbers of samples. While the genotypes of other SV types rely on the read pair distances, the genotypes of translocations currently only rely on the number of read pairs that map to different chromosomes. Because the translocation calling in PopDel is currently in a very preliminary phase, the process of combining windows and the genotyping of the samples for the translocations is also to be considered as an early prototype.

First, the window-wise translocation calls are sorted in ascending order according to the their remote chromosome as a first sorting key, their position on the current chromosome and their position on the remote chromosome as a second and third sorting key. The position of the window is used as a fourth sorting key. As each window can hold multiple clusters, each with read pairs of a distinct orientation (see Subsection 3.3.3.5) no read pair orientation is used as a sorting key when ordering the windows. The ordering of windows does not necessarily lead to all window-wise calls that refer to the same translocation event following each other. This is because there can be different translocations starting in close-by windows, with different positions on the same remote chromosome. Because the windows are first sorted by the remote chromosome, then by the positions on the local chromosome and lastly by the positions on the remote chromosome, this can lead to an interruption of the run of windows for one of the translocations. Therefore, a different combination scheme than the one for the other SVs is applied. Let w_i be the first of n window-wise translocation calls sorted in above fashion. Let w_{i+k} with $0 < k < n$ be another such window-wise call. w_i and w_{i+k} are considered similar, if all of the following conditions are met:

a) The remote chromosome of w_i and w_{i+k} is the same.

b) The breakpoint positions on the current chromosome p_{local} of w_i and w_{i+k} are located within $4\bar{\sigma}$ of each other.

c) The breakpoint positions on the remote chromosome p_{remote} of w_i and w_{i+k} are located within $4\bar{\sigma}$ of each other.

The check for similarity is performed for increasing k. If w_i and w_{i+k} are similar, they are immediately combined by appending the sample-wise estimates of the local and remote breakpoint according to the read pairs of different orientations (denoted as p^o_{local} and p^o_{remote} in Subsection 3.3.3.5) of w_{i+k} to those of w_i. Further, the sample-wise counts of the clusters per orientation of f w_i and w_{i+k} are summed up. w_{i+k} is then invalidated such that it cannot be combined with any other window. The check with increasing k for windows that are similar to w_i does not stop when a window does not meet the above conditions for similarity. It continues until a window $w_{i+k'}$ is encountered that has a different remote chromosome than w_i or the local position of w_i and $w_{i+k'}$ are more than $2\bar{\sigma}$ apart. If one of those cases occur, i is increased to match the index of the first window in $]i, k']$ that has been invalidated (i.e. it has already been combined). Similarly, k is set to the lowest index in $]i + 1, k']$ that has not been invalidated. Then, the process of checking for similarity and combining eligible windows continues.

Once the process reaches the end of the list, all windows in the buffer that represent the same translocation event have been combined. For each combined translocation, the final translocation positions and the sample-wise genotype likelihoods are calculated.

The final breakpoint positions of the translocation are calculated like $\text{median}(p^o_{\text{local}})$ and $\text{median}(p^o_{\text{local}})$ described in Subsection 3.3.3.5 with the only difference that now the combined list of positions from multiple windows is used. The final genotype likelihoods of all samples are calculated from the size of the positional clusters per orientation and their inferred support or non-support for the translocation. Let C^S denote the size of the positional clusters of sample S. Let n^S denote the sum of read pairs of S that do not support the translocation (i.e. they map to the same chromosome) in the combined windows. Let t^S denote the sum of sizes of positional clusters of S that support the translocation. For translocations that are assigned FR or RF orientation, the added sizes of the positional clusters with FR and RF orientation make up the number of translocation supporting reads. If instead the translocations is assigned an FF or RR orientation, t^S is derived from the sizes of the positional clusters with FF and RR orientation accordingly. π denotes a pseudo-count probability introduced to allow for noise (currently set to 0.001). The sample-wise genotype likelihoods are then calculated as:

$$\mathcal{L}\left(G_0|C^S\right) = \log_{10}(1-\pi)\cdot n^S + \log_{10}(\pi)\cdot t^S \tag{3.45}$$

$$\mathcal{L}\left(G_1|C^S\right) = \log_{10}\left(\frac{1}{2}\right)\cdot\left(n^S + t^S\right) \tag{3.46}$$

$$\mathcal{L}\left(G_2|C^S\right) = \log_{10}(\pi)\cdot n^S + \log_{10}(1-\pi)\cdot t^S \tag{3.47}$$

Because the same interchromosomal translocation event involves two chromosomes, the same event is typically detected twice by PopDel, once on each involved chromosome. Those calls are not combined by PopDel.

3.3.4 Availability

PopDel is written in C++ using the SeqAn library. Its source code is freely available on GitHub[2]. Further, it can be installed via the Conda package manager from the *bioconda* channel[3].

The functionality for the detection of duplications and inversions is currently part of the *develop* branch of the GitHub repository. A documentation of PopDel's functionality, parameters and example use-cases is available via its GitHub wiki[4].

[2]`https://github.com/kehrlab/PopDel`
[3]`https://anaconda.org/bioconda/popdel`
[4]`https://github.com/kehrlab/PopDel/wiki`

Chapter 4

Performance Evaluation of PopDel

This chapter compares the performance and accuracy of PopDel to other state-of-the-art SV callers. It introduces the simulated and real data sets used for the benchmarks, and explains the different performance metrics. Further, the results of the benchmarks are presented and interpreted.

Please note, that the benchmarks that are based on real data only comprise deletions. In those benchmarks, PopDel 1.2.2 was applied. The results of the deletion benchmarks (with exception of the analysis of the Simons Genome Diversity Cohort in Section 4.7) were first presented in the central publication of this thesis that also presented PopDel[67]. The prototypes of PopDel's duplication and inversion calling and genotyping (PopDel 2.0.0-alpha) are only covered by simulated data in Subsections 4.3.3.2 and 4.3.3.3.

4.1 Applied Performance Metrics

This section focuses on the description of the setup of the different benchmarks used for comparing PopDel and other SV callers, as well as the calculation and interpretation of the various applied performance metrics.

4.1.1 Definition of Matching Variants

The predicted properties of an SV, like breakpoint locations and size, often vary within certain boundaries, depending on the SV caller that made the prediction[30]. Therefore, a definition of when two SVs are considered to be identical (subsequently referred to as *matching*) is required for the calculation of most performance metrics.

Definition of Matching Variants for Real Data. A useful metric for the comparison and matching of SVs is the *reciprocal overlap*[92]. It measures the (relative) number of base pairs by which two or more genomic ranges overlap. Let $A = (s_A, e_A)$ and $B = (s_B, e_B)$ be two genomic ranges on the same chromosome, defined by their start and end coordinates s and e. Let further $\text{len}(A) = e_A - s_A$ denote the length of A and B respectively. Without loss of generality, assume that A ends before or with B, i.e. $e_A \leq e_B$. The reciprocal overlap can then be calculated as:

$$\cap_{rec}(A, B) := \frac{\max(e_A - \max(s_A, s_B), 0)}{\max(\text{len}(A), \text{len}(B))} \tag{4.1}$$

When given in percentages, a reciprocal overlap of 50% means that A and B overlap by exactly half of the bases of the longer range. The focus of the evaluation applying the reciprocal overlap lies on the presence or absence of a detected variant from a set of variant calls rather than the quality of the estimates for position and length. Therefore, a generous threshold of 50% reciprocal overlap is applied to decide if two variants are the same, if not stated otherwise. The ranges A and B are given by the start and end points of the compared SVs.

As an additional criterion, the SV type predicted by the caller has to match the true SV type.

Definition of Matching Variants for Simulated Data with Consideration of Genotypes. The call sets on the simulated data were additionally evaluated using an alternative criteria for matching variants. This comparison approach considers the proximity of a predicted variant to the simulated starting position and the agreement on SV size separately and also requires the genotype of a predicted variant to match the ground truth. Under this matching criteria, two variants of the same SV type are considered to match, if the predicted start position is within 300 bp of the true start position and the predicted length of the SV does not differ by more than 150 bp from the real length. To determine the number of correctly predicted alleles, the predicted genotypes are compared with the read genotypes: Let $G_{pred}(v) \in \{0, 1, 2\}$ be the number of predicted variant alleles for variant v and let analogously $G_{\text{sim}}(v)$ be the real number of variant alleles for v. The number of correctly predicted alleles ($G^+(v)$) and the number of incorrectly predicted alleles ($G^-(v)$) are calculated as:

$$G^-(v) = |G_{\text{pred}}(v) - G_{\text{sim}}(v)| \tag{4.2}$$

$$G^+(v) = 2 - G^-(v) \tag{4.3}$$

Variants with the special case were $G_{\text{pred}}(v) = G_{\text{sim}}(v) = 0$ are excluded from counting because counting them would allow a variant caller to artificially inflate its count of correctly predicted alleles by calling false positive variants and correctly genotyping these variants with $G_{pred}(v) = 0$.

A called variant that is not part of the the truth set is counted with one incorrect allele if it was assigned a heterozygous genotype in the simulated genome by the caller. If it was assigned a homozygous carrier genotype by the caller it is counted with two incorrect alleles.

A true variant that is not detected by the SV caller is counted as one incorrect allele if the true variant has a heterozygous genotype in the simulated genome. If the true variant has a homozygous carrier genotype, it is counted with two incorrect alleles.

4.1.2 Single-Sample-Scale Metrics

For the evaluation of the callers performance on a single sample the commonly used metrics precision and recall are applied. In this context, a true positive (TP) is defined as a predicted variant call that has a reciprocal overlap of at least 50% with a true variant (see formula 4.1) and are of the same variant type. A false positive (FP) is defined as a predicted variant that does not match a true variant. A false negative (FN) occurs when a true variant is not matched by a predicted variant. Each variant from the truth set may only be matched with one variant from the call set and vice versa. Let further $\text{TP}(\mathcal{V})$ denote the number of TP in the set of predicted variants $\mathcal{V}$ and let $\text{FP}(\mathcal{V})$ and $\text{FN}(\mathcal{V})$ be defined analogously. Single-sample-scale metrics can of course also be used to measure the performance in scenarios where more than one sample is involved. Nevertheless, they are described here, since they are the main measures for single-sample benchmarks.

Precision. The precision is a commonly used measure for estimating the reliability of positive classifications (i.e. a variant calls). In the context of variant detection it reports how many of the predicted variants are actually correct and can be found in the truth set. A perfect precision of 1 indicates that a positive classification is always true. A precision of 0 means that all of the positive classifications are wrong.

$$\text{Precision}(\mathcal{V}) := \frac{\text{TP}(\mathcal{V})}{\text{TP}(\mathcal{V}) + \text{FP}(\mathcal{V})} \tag{4.4}$$

For the precision to be correct and meaningful, it is important that the truth set is actually comprehensive. Using an incomplete truth set would result in a reduced precision for a variant caller that correctly detects those variants that are not part of the truth set, thereby falsely suggesting a lower performance of the variant caller. Truth sets of simulated data are especially useful for measuring the precision because all simulated variants are known. One still has to keep in mind that simulated data is not capable of reflecting all properties of real data. For this reason, the precision measured on simulated data should be interpreted as an upper bound that will not be reached on real data in most cases. For real data, measuring the precision is more challenging, since it is hard to create a truth set that contains *all* variants of a biological sample. The Genome in a Bottle (GIAB) sample HG002 from the well studied Ashkenazi Jewish trio is a valuable resource for this purpose[92]. It provides a gold standard SV call set that is reported to be comprehensive in well-defined regions of the genome. This allows the meaningful estimation of the precision on real data, albeit limited to the defined regions. This excludes large portions of the genome where the correct detection of SVs, especially using short-read technologies, is more difficult, e.g. due to the presence of highly repetitive sequence. For the sake of readability, the precision is often given in percentages of one.

Recall. In the context of SV calling, the recall measures how reliably true SVs are detected by the caller. It is often also termed *sensitivity*. A perfect recall of 1 means that all variants in the truth set are also present in the call set. A recall of 0 indicates that no variant from the truth set has been found.

$$\mathrm{Recall}\left(\mathcal{V}\right) := \frac{\mathrm{TP}\left(\mathcal{V}\right)}{\mathrm{TP}\left(\mathcal{V}\right) + \mathrm{FN}\left(\mathcal{V}\right)} \tag{4.5}$$

Similar to the precision, it is vital for the recall that the truth set is comprehensive and reliable. For the sake of readability, the recall is often given in percentages of one.

F1 Score. The F_1 score is a special case of the F score. It combines precision and recall by being calculated as their harmonic mean:

$$F_1\left(\mathcal{V}\right) := \frac{\mathrm{TP}\left(\mathcal{V}\right)}{\mathrm{TP}\left(\mathcal{V}\right) + \frac{1}{2}\left(\mathrm{FP}\left(\mathcal{V}\right) + \mathrm{FN}\left(\mathcal{V}\right)\right)} = 2 \cdot \frac{\mathrm{Precision}\left(\mathcal{V}\right) \cdot \mathrm{Recall}\left(\mathcal{V}\right)}{\mathrm{Precision}\left(\mathcal{V}\right) + \mathrm{Recall}\left(\mathcal{V}\right)} \tag{4.6}$$

Same as precision and recall, the F_1 score lies between 0 (precision or recall of 0) and 1 (perfect precision and recall). While it gives a quick overview of both metrics combined, depending on the use case, one should not solely rely on the F_1 score. For some cases a high precision is more important or desirable than a high recall or vice versa. This makes the consideration of the individual values for precision and recall necessary. For the sake of readability, the F_1 score is often given in percentages of one.

Ratio of Correct Alleles. When not only considering whether or not a variant has been detected in a sample, i.e. the genotype for respective sample and variant is 0/1 or 1/1, the previous binary classification problem becomes a multiple one. Each of the three possible genotypes of a variant represents one of the possible classifications per sample.

To obtain a simple measure of the correctness of the genotypes, it is useful to move the level of the measurement from the genotype level to the allele level. This means that not the number of correct and incorrect genotypes are counted but the number of correct and incorrect alleles. In this case, a predicted 0/0 genotype for a sample that actually has a 1/1 genotype for the variant in question will be counted as zero correct alleles and two incorrect alleles (see formula 4.2 and 4.3), while a 0/1 prediction would result in one correct and one incorrect allele being counted. The ratio of correct alleles for the set of predicted variants $\mathcal{V}$ is calculated as:

$$\text{Ratio of correct alleles}\,(\mathcal{V}) := \frac{\sum_{v\in\mathcal{V}} G^{+}(v)}{\sum_{v\in\mathcal{V}} G^{+}(v) + G^{-}(v)} \tag{4.7}$$

Call Set Overlap. Especially when comparing multiple SV callers, it can be informative to inspect which variants from one call set are also part of the other call sets and which are not. This agreement of call sets can be displayed in the form of *Venn diagrams* and many conclusions (sometimes requiring manual investigation of the variants in the sets) can be drawn from the overlapping areas: Many callers agreeing on a particular variant that is not part of the truth set can be an indication that the variant is missing from the truth set or that the variant callers suffer from the same wrong conclusion about the variant. Reasons for this may be the presence of sequencing artifacts or incorrect alignments. The overlap of a single call set with the truth set that is not covered by any other tool gives insight into the number of variants that are only detected by the respective caller. The opposite case, where one caller is the only one not detecting a certain set of variants can help to discover weaknesses of the approach by the caller.

4.1.3 Family-Scale Metrics

Performing benchmarks in the presence of family information enables the comparison of the predicted variants and genotypes with the theoretically expected patterns of inheritance. Since this evaluation approach does not rely on a ground truth set it is also applicable for less well studied genomes for which no comprehensive gold standard set of SV calls is available. In the course of the evaluation of PopDel, the metrics *Mendelian inheritance error rate* (MIER) and the *transmission rate* have been calculated on the Ashkenazi Jewish trio (see Subsection 4.4.2, MIER only), and the 49 parent-offspring-trios (subsequently only referred to as trios) of the Polaris Kids Cohort (see Subsection 4.6). The two metrics complement each other, in a sense that the MIER is useful for the evaluation of common variants that are present in many genomes, while the transmission rate is useful for the evaluation of rare variants. In the context of family scale metrics, variants are counted per parent-child-trio. This means that a variant call with at least one variant allele predicted in any member of a trio will be counted as one variant for this trio and as another variant for another trio, if it is also genotyped with at least one variant allele for that trio.

Mendelian Inheritance Error Rate. The MIER measures the rate at which the genotype combinations of a trio violate the Mendelian laws of inheritance. Specifically, this means that the child must only exhibit a specific variant allele if it is present in at least one of the parents. If one parent is a homozygous carrier of a variant, the child must also carry the variant on at least one haplotype. If both parents are homozygous carriers, the child must be homozygous carrier too. Figure 4.1 visualizes all possible genotype combinations in a trio and their conformity with Mendelian inheritance. Let $|\mathcal{V}|_c$ be the number of variants detected for all trios with predicted genotype combinations that abide the Mendelian rules, also referred to as *consistent* variants. Let analogously $|\mathcal{V}|_{\bar{c}}$ be the number of predicted variants for all trios whose predicted genotypes do not follow the Mendelian rules. The MIER is calculated as:

$$\mathrm{MIER}\left(\mathcal{V}\right) := \frac{|\mathcal{V}|_{\bar{c}}}{|\mathcal{V}|_c + |\mathcal{V}|_{\bar{c}}} \tag{4.8}$$

to avoid an inflation of the MIER, variants that are genotyped as homozygous reference in all members of a trio are excluded from the calculation. The MIER has also limitations. The genotypes of a variant being consistent does not necessarily mean that they are indeed the true genotypes. It just mean that this specific combination of genotypes is one of multiple plausible combinations. This circumstance results in a hypothetical variant call set where all samples are genotyped as heterozygous carriers for all variants to achieve a perfect MIER of 0. The same holds for a call set with all samples always genotyped as homozygous carriers. Since systematic biases in the genotyping of the SV callers is something a good benchmark should address, it is important to apply an additional filter for those cases if possible. The solution applied in the presented benchmark is the application of a filtering step based on the Hardy-Weinberg equilibrium (HWE)[34]. Although the HWE assumes an ideal population of infinite size, with (among others) random breeding, absence of selection, migration and mutation, its principles can be applied to real world population genetics, since the model behaves close enough to real world data. The HWE gives the expected frequencies at which different alleles can be observed. With p and q as the allele frequencies of two alleles P and Q at a locus, it states:

$$p^2 + 2pq + q^2 = 1 \tag{4.9}$$

The summands of the formula can be interpreted as the expected frequencies for the three genotypes in the population. p^2 and q^2 are the expected frequencies of homozygous carriers of the P-alleles and Q-alleles respectively. The expected frequency of carriers of both the

P-allele and Q-allele is given by $2pq$.

Taking a hypothetical variant that has been genotyped as 0/1 in all members of all trios in a reasonably big cohort, the allele frequencies would be $p = q = 0.5$. The HWE would thus dictate a distribution of 25% homozygous non-carriers, 50% heterozygous carriers and 25% homozygous carriers, which does not match the predicted 100% heterozygocity rate, indicating that the genotyping of the variant is most likely wrong. Using a χ^2-test one can further compare the predicted distribution of genotypes with the ideal distribution given by the HWE to assess if the two distributions are different at a statistically significant level. Variants whose distributions differ at the chosen significance level (here: p-value = 0.01) are removed from the analysis.

The MIER ignores the possibility of *de novo* mutations. If there is a true *de novo* mutation in an offspring that is correctly genotyped, this will be counted as an inconsistent genotype combination if the parents are also correctly genotype as homozygous non-carriers. For the overall correctness of the approach this is of no grave concern because the *de novo* mutation rate for SVs is neglectably small[46].

In order to avoid biases in the evaluation of PopDel and the other SV callers, no pedigree information was provided to the tools.

Transmission Rate. The transmission rate can be used to assess how often a variant allele from a *heterozygous* parent was inherited by the offspring, i.e. transmitted. The underlying rationale is that a parent that is heterozygous for a specific variant should theoretically transmit this variant to its offspring in 50 % of the cases. The effects of selection and disease-causing alleles can skew the expected values, but for cohorts of healthy people, like those discussed in this thesis, these effects are negligible. Let $|\mathcal{V}|_{G^{\text{trio}}}$ be the number of variants with a fixed combination of genotypes G^{trio} for the trio and let $|T|_{G^{\text{trio}}}$ be the number of variant alleles transmitted from a heterozygous parent to the child for that genotype combination. The transmission rate for this combination of genotypes is then calculated as:

$$\text{Transmission rate}\big(\mathcal{V}, T, G^{\text{trio}}\big) := \frac{|T|_{G^{\text{trio}}}}{|\mathcal{V}|_{G^{\text{trio}}}} \tag{4.10}$$

There are some restrictions to the genotype combinations. Combinations not in line with the Mendelian rules (marked red in Figure 4.1) are excluded because they are treated as errors. Cases were both parents are homozygous carriers or both are homozygous non-carriers are excluded as well because they leave no room for any variations in the genotype of the child (marked yellow in Figure 4.1) and are therefore not informative for the calculation. This leaves the cases where one parent is a homozygous carrier or non-carrier while the other is heterozygous (marked blue in Figure 4.1) and the cases where both parents are

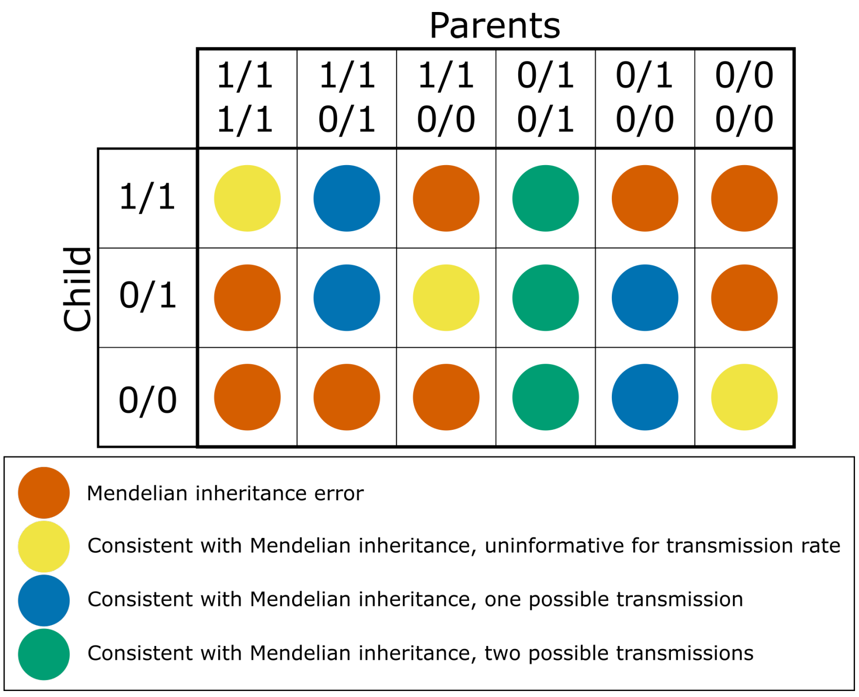
Parents
1/1 1/1
1/1 0/1
1/1 0/0
0/1 0/1
0/1 0/0
0/0 0/0
Child
1/1
0/1
0/0
Mendelian inheritance error
Consistent with Mendelian inheritance, uninformative for transmission rate
Consistent with Mendelian inheritance, one possible transmission
Consistent with Mendelian inheritance, two possible transmissions

Figure 4.1: Visualization of Mendelian consistency and informativeness for the transmission rate. Column give the genotype combinations of the parents and rows give the genotype of the child. Colors indicate agreement with the Mendelian laws of inheritance and the informativeness for the calculation of the transmission rate.

heterozygous (marked green in Figure 4.1). In the former cases, the homozygous parent will never (non-carrier) or always (homozygous carrier) transmit the variant allele to the child. By observing the other allele of the offspring one can see if the other heterozygous parent transmitted the variant allele or not. This leads to an expected transmission rate of 50 % for these combinations of genotypes. If both parents and the child are heterozygous, it is not possible to tell which parent transmitted which allele. This would required phasing of the variants, which is beyond the scope of the discussed benchmarks. Nevertheless, it is still possible to state that the child is expected to be heterozygous in 50 % of the cases and a homozygous carrier or non-carrier in 25 % of the cases each (also see formula 4.9).

For the assessment of the detection and genotyping of rare variants it is further possible to limit the transmission rate calculation to those variants that were only detected in a single trio.

4.1.4 Population-Scale Metrics

WGS data sets like those of the Polaris Diversity Cohort or of the Simons Genome Diversity Project[62] contain genomes from different (sub)populations. This allows the inspection of the call sets created by a variant caller to assess if the detected variants match known population-scale patterns.

Variant Count. Previous studies have shown that the number of detected variation varies between different populations from different continental groups[16;72;82]. For example, it is known that genome samples from African (AFR) populations tend to exhibit more variants than those of other continental populations (see Figure 4.2a)[16;72]. A possible explanation for this common observation can be found in the *Out of Africa hypothesis* (OOAH), which states that the origin of the modern human lies on the African continent. When early humans migrated from Africa to other parts of the world, they were subject to migration bottlenecks because only relatively small groups of them actually migrated. This limitation of the new population's gene pool led to a genetically more homogeneous population than the original pre-migration population[72]. The OOAH therefore not only explains the difference between populations but also the variation within the populations. Because the AFR population was not subjected to migration bottlenecks to the same extend as the other continental populations, the diversity within the AFR population is observably higher. This results in a higher standard deviation of the distribution of variants.

Principal Component Analysis and t-Distributed Stochastic Neighbor Embedding. The *principal component analysis* is a statistical method often used for linear dimensionality reduction. The principal components are obtained by calculating the eigenvectors of the covariance matrix of a data set. The principal components are orthogonal to each other. Each principal component explains a portion of the variance of the data set. The principal component explaining the biggest portion of the variance is called the first principal component. Using the first two principal components of a data set, one can plot the previously high dimensional data in two dimensions and reveal underlying structures.

In the context of population-scale variant calling, one can assume that the population an individual originates from is one of the main factors that influence the genetic variation. Consequently, it can be expected that the first principal components captures the variation caused by the population of the samples and separate samples from multiple populations accordingly. This separation patterns are already known from previous studies on small variants and SVs (see Figure 4.2b).[16;82]

Specifically for the discussed benchmarks, the PCA was computed as follows: Deletions detected by a variant caller where converted into a Matrix M, representing the number of detected deletion alleles for each sample. The rows of M represent the sample and the columns represent the different deletions, ordered by genomic location. $M_{i,j} \in \{0, 1, 2\}$ contains the number of deletion alleles detected in sample i for deletion j according to the predicted genotype. Duplicate samples (i.e. rows) were removed because they are not informative for the PCA. Further, variants in high linkage disequilibrium where removed to avoid correlation between the columns of the M. This was done by calculating the Spearman correlation for neighboring variants and removing those with a significant correlation coefficient (p-value ≤ 0.05). The approach was iteratively applied until no more significant correlations could be found. After filtering and shuffling of the sample order, the PCA was computed using the function *prcomp* of the R programming language with the options for 0-centering and unit variance scaling enabled. When plotting the first two principal components of the data one can observe a distinct clustering of the samples (see Figure 4.2b). Coloring or labeling the data points according to the samples' reported ancestries reveals how good the clustering matches the known populations.

For a better visualization of the clustering of the samples, one can alternatively apply a *t-distributed stochastic neighbor embedding*[36;85] (t-SNE) instead of a PCA. While the PCA can be used as a *linear* dimensionality reduction technique, the t-SNE is *non-linear*. This enables a better visualization of high dimensional data in two dimensions. This comes at the cost of distorting the distances of dissimilar data points, whereas the distances of similar data points are more preserved[65;85]. In this thesis, t-SNE was computed on the same filtered data as the PCA using the R function *tsne* from the *M3C* package using the default options for all non-cosmetic options. A computed t-SNE is not canonical. The exact coordinates and relative positions of the inferred clusters can vary to some degree, based on the ordering of the samples and the random state of the algorithm. Therefore, one should compare the results of multiple initializations and carefully interpret the embedding using additional background information for the validation wherever possible.

PCA and t-SNE are a very sensitive tools capable of revealing hidden structures. Considering that the population is a strong factor determining the genetic makeup of an individual, a good clustering of samples by population must not be interpreted as a proof that a variant call set is of high quality. Rather it shows that the predicted variants carry some biological meaning and are not purely of technical or random nature. On the other hand, failure to produce a good clustering, is a clear indication of severe shortcomings in the analyzed call set.

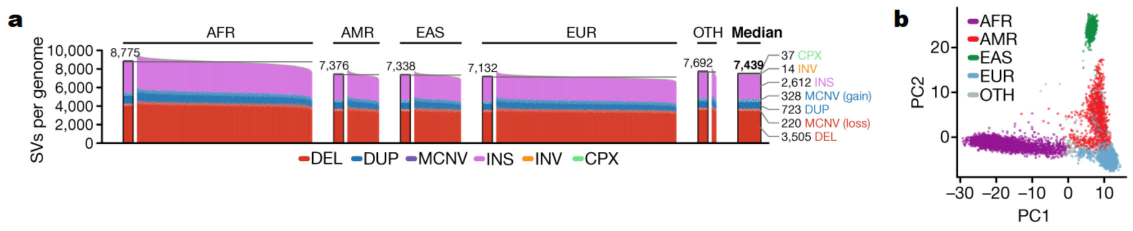

Figure 4.2: Population differences by detected SVs. **a** Number of detected SVs by continental population (African/African American, AFR; Latino, AMR; East Asian, EAS; European, EUR; Other, OTH) and SV type (complex, CPX; inversion, INV; multiallelic copynumber variation; MCNV; duplication, DUP; deletion, DEL). **b** Principal component analysis showing the separation of the continental populations. Figures adapted from Collins et al. (2020)[16].

4.2 Compared SV Callers

In the following benchmarks, PopDel was compared with different established SV callers. Those SV callers are Delly (0.7.8), GRIDSS (1.8.1), Manta (1.6.0) and Smoove (0.2.4, using Lumpy 0.2.13 and SVtyper 0.7.0). The parameters of the SV callers in the different benchmark scenarios are described in the respective subsections.

4.3 Simulated Data

Simulated data can be used for benchmarking under controlled conditions. All properties of the simulated variants are known, making an accurate assessment of precision and recall or the vicinity of the predicted and simulated breakpoints possible. Given enough storage space and computational resources, one can simulate thousands of samples to enable comparisons on the performance and behavior of the benchmarked SV callers for growing numbers of well-defined samples. All parameters, like coverage, read length and insert size distribution, of the simulated "sequencing" can also be controlled, guaranteeing controlled conditions for the comparison. A considerable shortcoming of simulated data is that it does not completely capture the complexity of real data. When consulting benchmarks, one regularly observes that the benchmarked tools perform better on simulated data than on real data. This can be attributed to the simulated reads themselves being more exact and of better quality than their real counterparts but also to the variants that are being simulated. As described in Section 1.2 and 2.1, different sequences, e.g. repeats, can make a mutation at a location more likely to occur. If one simulates variants ignorant of the sequence context, the association of variants with the sequence context is not captured by the results. This reduces the transferability of the benchmarking results. Despite these shortcomings, simulated data is a common approach to assess some baseline in the benchmark and can be used complementary to the analysis on real data. The subsequent sections describe in detail how the data for the benchmarks of this thesis was simulated and how the variant callers were executed on the data and how the performance metrics discussed in previously were assessed. Further, the results are presented and discussed.

4.3.1 Data Simulation Protocol

4.3.1.1 Deletions

The simulated deletion data can be divided into two cohorts. The first cohort (*uniform deletion simulation*) consists of 1,000 simulated samples of the human chromosome 21 and contains 2,000 uniformly distributed deletions with a uniform size distribution between 100 and 10,000 bp, and uniformly distributed allele frequencies between 0 and 1. Gap regions of the genome were excluded and deletions were required to be at least 1,000 bp apart. The second cohort (*G1k simulation*) consists of 500 simulated samples of the human chromosomes 17-22. For this cohort, the deletions were not randomly generated, but taken from the 1000 Genomes Project[17] (G1k)[1]. This allows to better reflect the influence of the sequence context and size distribution of the deletions in the simulated data. Because the number of simulated deletions per chromosome from the G1k is comparatively low (3,246 deletions on chromosome 17-22), it was necessary to simulate multiple chromosomes to get robust measures for precision and recall.

The uniform deletion simulation cohort was generated by creating a set of 2,000 haplotypes from the chromosome 21 sequence of GRCh38 and inserting the simulated variants sampled according their allele frequencies into the sequences. For the G1k simulation cohort, the G1k deletions were sampled instead and inserted into 1,000 haplotypes of the chromosome 17-22 sequences of GRCh37. GRCh37 was selected for this cohort because the G1k variant call set is based on GRCh37. The haplotypes were combined into 1,000 (uniform deletion simulation) and 500 (G1k simulation) diploid genomes respectively. The generation of the simulated NGS reads from the simulated genomes was performed using *art_illumina*[38] (see Appendix Section A.3 for detailed parameters). Subsequent alignment to GRCh38 and GRCh37 was performed using BWA-mem[56;57]. Parameters included '-R' for including read group tags and '-M' for marking shorter split read alignments as *secondary*. The scripts for the simulation of deletions are available on GitHub[2].

4.3.1.2 Duplications

Only one cohort of duplications was simulated for the initial testing of PopDel's duplication calling and genotyping. The simulation of duplications was performed similar to the approach for the uniformly distributed deletions described in Subsection 4.3.1.1. The

[1]`http://ftp.1000genomes.ebi.ac.uk/vol1/ftp/phase3/integrated_sv_map/ALL.wgs.mergedSV.v8.20130502.svs.genotypes.vcf.gz`

[2]`https://github.com/kehrlab/PopDel-scripts`

duplication simulation cohort consisted of 10 simulated samples of the human chromosome 21. The sizes and locations of the variants were identical to the 20 haplotypes of the first ten samples of the uniform deletion simulation cohort. Instead of deleting the sequence in the specified regions, the sequences were duplicated and inserted again at their original position to generate tandem duplication. The generation of the simulated NGS reads from the simulated genomes was performed using art_illumina with the same settings as for the deletion data simulation. Subsequent alignment to GRCh38 was performed using BWA-mem with the same settings as for the deletion data simulation.

4.3.1.3 Inversions

Only one cohort of inversions was simulated for the initial testing of PopDel's inversion calling and genotyping. The simulation of inversions was performed similar to the approach for the uniformly distributed deletions described in Subsection 4.3.1.1. The inversion simulation cohort consisted of 10 simulated samples of the human chromosome 21. The sizes and locations of the variants were identical to the 20 haplotypes of the first ten samples of the uniform deletion simulation cohort. Instead of deleting the sequence in the specified regions, the sequences were replaced by their reverse complement to generate inversions. The generation of the simulated NGS reads from the simulated genomes was performed using art_illumina with the same settings as for the deletion data simulation. Subsequent alignment to GRCh38 was performed using BWA-mem with the same settings as for the deletion data simulation.

4.3.2 Comparison Protocol

Benchmarking on simulated data was performed for the SV callers Delly, GRIDSS, Manta, PopDel (1.2.2 for the deletion benchmarks, 2.0.0-alpha for the inversion and duplication benchmarks) and Smoove. All tools were run on increasing numbers of samples in steps of 1 for up to 10 samples, steps of 10 up to 100 samples, and steps of 100 up to 500 (G1k simulation) or 1,000 samples (uniform deletion simulation). Whenever supported by the tools, the calling was limited to the simulated chromosomes. Benchmarking on the simulated duplication and inversion inversion data was performed with increasing numbers of samples in steps of 1 up to 10 samples. Specifically, the tools were applied as follows:

Delly was applied using mostly default parameters. Additionally, the option '-n' for disabling small indel realignment was applied, to reduce running time. As small indels were not part of the benchmark, keeping the small indel realignment active would only increase the running time of Delly with not benefits in terms of precision and recall. For the deletion simulation benchmark, variants other than deletions were removed from the call set. No filtering for the value of the VCF's filter field was applied because doing so was observed to have a negative effect on the sensitivity without benefits for the precision.

GRIDSS was run using default parameters and a heap size of 8 GB for the Java virtual machine. Non-deletion variants and variants that had the VCF's filter field report "LOW_QUAL" were removed from call sets prior to evaluation. Because GRIDSS only produces breakpoint calls, an adapted version of the annotation script provided by GRIDSS' authors in its GitHub repository[3] was used to convert the breakpoints into deletion calls. The adaptations were necessary because the original version of the script would not run without errors. As GRIDSS does not report genotypes, it was excluded from genotype based analyses.

GRIDSS was only tested on the uniform deletion simulation data.

Manta was executed using mostly default parameters. Additionally, its configuration step was run with the '–region' option to limit the calling to the simulated chromosome(s). The workflow was executed in the 'local' mode (as opposed to being distributed across multiple nodes of a compute cluster) using one thread. For the deletion simulation benchmark, non-deletion variants were removed from the call set prior to evaluation. No filtering for the value of the VCF's filter field was applied because doing so was observed to have a negative effect on the sensitivity without benefits for the precision. Manta produces different files for SVs that are called with different degrees of confidence/evidence. For the evaluation only the file containing the SVs called with high confidence was used because the other SVs had a very high false positive rate without considerable benefit for the recall.

Manta was excluded from the evaluation of the simulated inversion data, because it mostly detected the inversions as breakends (VCF notation: BND) and occasionally as insertion but never correctly resolved as inversion.

PopDel was applied using it's default parameters while limiting the processing of the samples to the simulated chromosomes. As dictated by it's workflow, the profiles of all samples were created individually using *PopDel profile* and the joint calling was performed using *PopDel call*. No further filtering of the predicted variants was performed.

PopDel 1.2.2 only implemented the detection and genotyping of deletions at the time of the uniform and G1k deletion benchmark. For the duplication and inversion benchmarks PopDel 2.0.0-alpha was applied using the same parameters.

[3]https://github.com/PapenfussLab/gridss/blob/master/example/simple-event-annotation.R

Smoove was applied using the settings recommended by its authors on Smoove's GitHub page: The recommended file for the exclusion of certain regions on GRCh37 or GRCh38 was used and mappings to chromosomes other than the simulated ones were excluded using the '–excludeChroms' option. Since Smoove does not recommend joint calling for large cohorts, the workflow was as follows:

1. Single-sample SV calling and genotyping using *Smoove call.*
2. Merging of the calls using *Smoove merge.*
3. Genotyping all individual samples for the merged calls using *Smoove genotype.*
4. Joining all SV calls and genotypes from step 3 using *Smoove paste.*

No further filtering of the predicted variants was performed.

4.3.3 Results

The results for the detection and genotyping of deletions, duplications and inversions for simulated data will be presented in the following subsections.

4.3.3.1 Results for Simulated Deletion Data

Precision, recall and F_1 score were measured as described in Subsection 4.1.2 on the filtered call sets for increasing batch sizes. Evaluations were performed on both the uniform simulation data and the G1k simulation data, with and without consideration of the genotypes.

Without Consideration of Genotypes. When evaluating only the presence or absence of variant calls without considering the individual genotypes of the samples, one observes that all compared SV callers achieve an almost perfect precision. This indicates that every variant that is called or for which a sample is genotyped as a carrier is most likely present and that the number of false positives is very low (see Figure 4.3a). As the number of samples in the cohort increases, the precision stays stable for all compared SV callers, with the exception of GRIDSS. GRIDSS suffers from an increasing number of false positive predictions for 50 and more samples. Further, GRIDSS was not capable of processing more than 400 human chromosome 21 samples simultaneously, which - together with its lack of genotypes - is why it was excluded from further analysis of joint variant calling. The recall of all tools is close to 0.9, with PopDel achieving the highest value for most batch sizes. The recall of all tools shows a slight but statistically significant increase for growing sample numbers (one-sided p-value (Spearman): $3.2 \cdot 10^{-4}$(PopDel), $8.4 \cdot 10^{-8}$(Delly), $2.8 \cdot 10^{-15}$(Smoove), $6.2 \cdot 10^{-3}$(Manta), $8.5 \cdot 10^{-6}$(GRIDSS)). The tables of this evaluation are part of the Appendix (Section A.4).

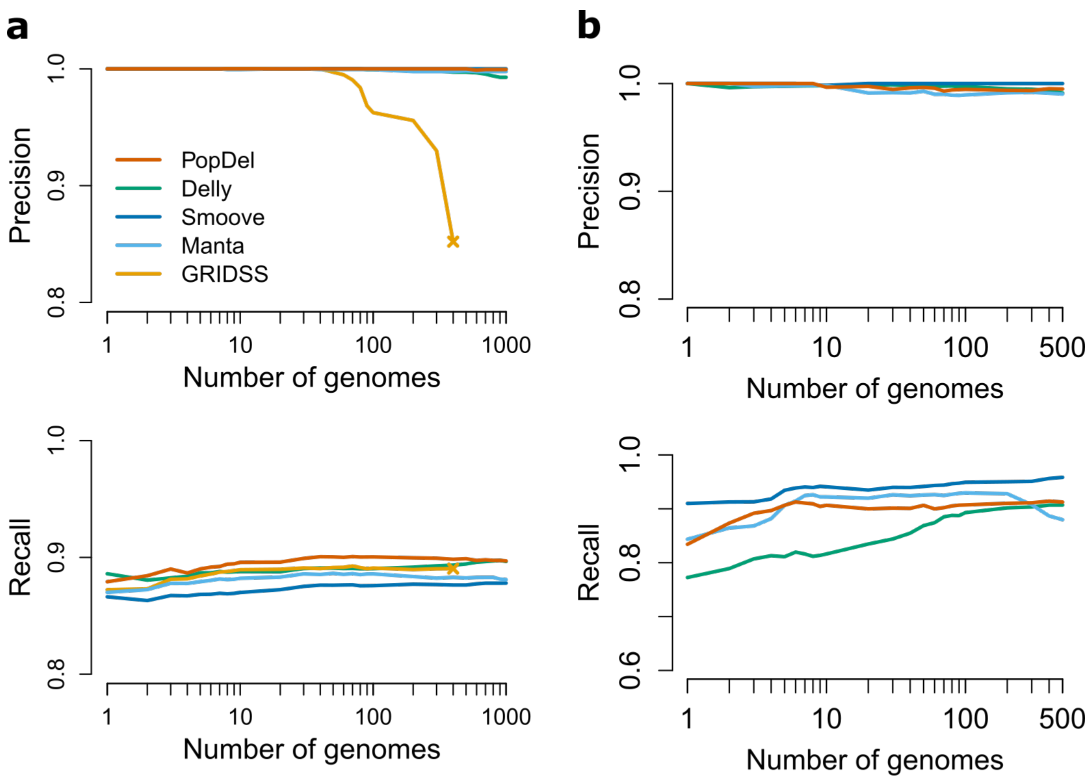

Figure 4.3: Precision and recall without genotype consideration on simulated deletion data. A cross at the end of a line indicates that the respective tool could not process more than the marked number of samples together. **a** Uniform deletion simulation cohort with up to 2,000 deletions in up to 1,000 simulated human chromosome 21 samples. **b** G1k simulation cohort with 3,246 deletions in up to 500 simulated human chromosome 17 - 22 samples.

Similar trends can be observed in the analysis of the G1k simulation cohort (Figure 4.3b). The precision is close to perfect for all tools, with Smoove performing the best (only 1 false positive for seven to ten samples, 0 otherwise). PopDel is generating some false positive calls starting at 9 samples and its overall number of false positives for 500 samples is second lowest (0 (Smoove), 15 (PopDel), 26 (Delly), 28 (Manta)). The recall on the G1k simulation data is overall lower for all tools except Smoove, which interestingly has a higher recall in this scenario than in the uniform deletion simulation scenario (recall between 0.91 - 0.96 compared to 0.86 - 0.88), even though the G1k simulation was designed as the more difficult scenario. One hypothesis that could explain this behavior was that the recommended list of excluded regions applied for Smoove affects more variants of the uniform deletion simulation data than of the G1k simulation because the uniform deletion simulation has the simulated variants more densely distributed across the chromosome and is ignorant of the sequence context (except gap regions that are avoided). But since only 15 deletions from the truth set for 1,000 samples of the uniform deletion simulation overlap with the excluded regions of Smoove, this cannot explain the notable difference in recall because those 15 variants only would amount to an recall increase of 0.0075 if they were called. Further investigation of the possible reason for Smoove's performance is therefore required.

For all tools the trend of the recall increasing with a growing number of samples can again be observed, and is more pronounced than for the uniform deletion simulation data. The p-values of the one-sided Spearman correlation test are $2.6 \cdot 10^{-3}$(PopDel), $9.7 \cdot 10^{-7}$(Delly), $1.3 \cdot 10^{-6}$(Smoove) and $1.0 \cdot 10^{-2}$(Manta). The tables of this evaluation are part of the Appendix (Section A.4).

Overall, the genotype-free precision and recall benchmark on both simulated data sets shows that PopDel reliably detects deletions, with a stable performance for increasing sample numbers and that its performance is on par with other state-of-the-art SV callers. Increasing the number of samples in the analyzed cohorts has a positive influence on the recall without adversely affecting the precision.

With Consideration of Genotypes. Figure 4.4 shows the ratio of correct alleles (formula 4.7). As shown in Figure 4.4a, the all tools genotype more than 90% of all alleles correctly for the uniform deletion simulation data, with PopDel making the most correct predictions. The performance is stable across all sample number for all tools.

For the genotyping evaluation on the G1k simulation data, the results are more spread out (Figure 4.4b). While PopDel and Delly show a lower ratio of correct alleles than before, Manta and Smoove perform better on the G1k simulation data, with a correct allele ratio of up to 0.96 (Manta) and 0.95 (Smoove). PopDel's ratio of correct alleles lies between 0.89

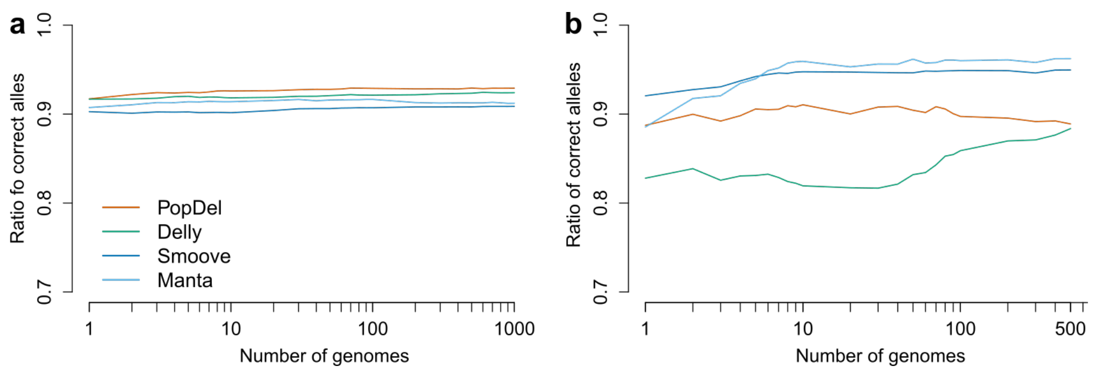

Figure 4.4: Evaluation of genotype correctness on simulated deletion data. **a** Ratio of correctly classified alleles in the uniform deletion simulation cohort with up to 2,000 deletions in up to 1,000 simulated human chromosome 21 samples. **b** Ratio of correctly classified alleles in G1k simulation cohort with 3,246 deletions in up to 500 simulated human chromosome 17 - 22 samples.

and 0.91 and is therefore slightly lower than for the uniform simulation data, for which it lies between 0.92 and 0.93. Similar to the recall in Figure 4.3, most tools show a trend of an increasing ratio of correct alleles for growing sample numbers, with significant Spearman correlations (one-sided) for Delly (p-value $= 5.7 \cdot 10^{-4}$), Smoove (p-value $= 1.5 \cdot 10^{-6}$) and Manta (p-value $= 1.5 \cdot 10^{-6}$). PopDel's genotyping performance is not significantly influenced by the number of samples in a cohort. The tables of this evaluation can be found in Section A.5 of the Appendix.

The results demonstrate that PopDel reliably genotypes deletions without being influenced by the number of samples in the cohort.

Breakpoint Precision. The starting positions of the deletions reported by Delly, Manta, PopDel and Smoove on the first ten samples of the uniform simulation data were evaluated. The predicted starting positions of all true positive deletions were taken from the VCFs and compared to the true starting positions of the simulated variants to create the histograms of start position deviations shown in Figure 4.5 below. A predicted starting position was assigned a deviation of 0 bp if it matched the true variant starting position exactly. Negative deviations indicate an estimate that is too far upstream, while a positive deviation indicates an estimate that is too far downstream. Left or right alignment was not performed because the true positions were known even for deletions falling into repetitive regions. By not using the left or right alignment the histograms show the tendencies of the different tools to over- or underestimate the true positions. This measure does not provide any information on the exactness of the length estimate of the deletion or the exactness of the end position of the deletion. The exactness of the deletion starting positions reported by PopDel (a) are on par with Delly (b), Smoove (c) and Manta (d). The histograms show that all tools

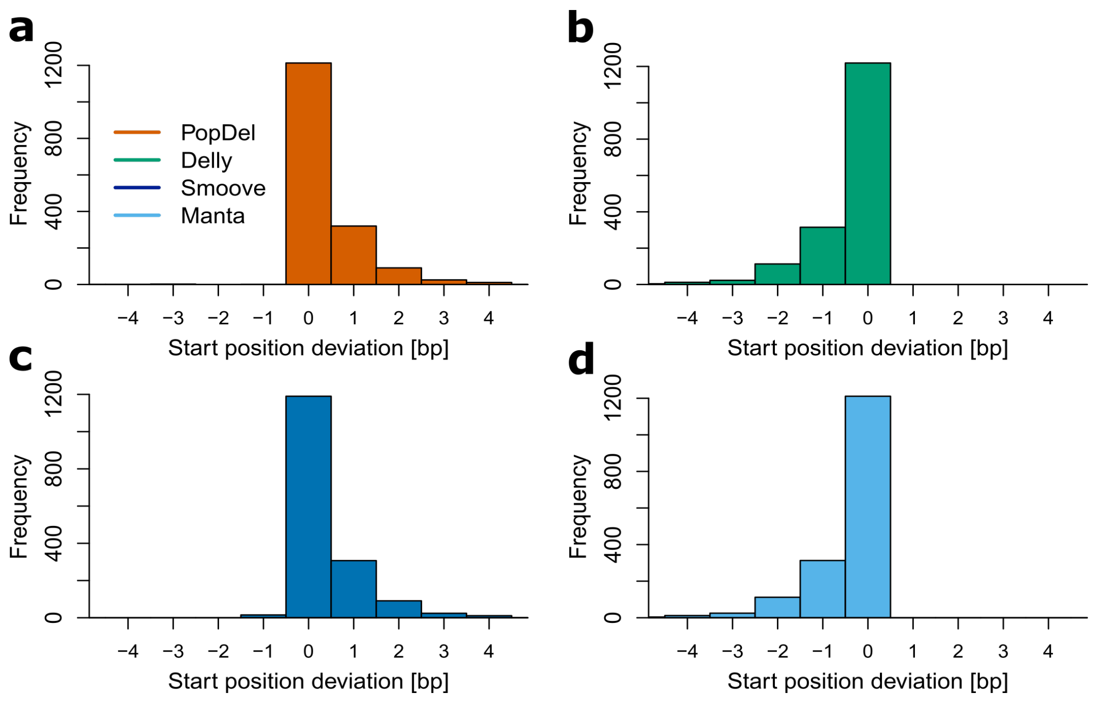

Figure 4.5: Histograms of deletion start positions deviations. The comparison is based on ten samples of the uniform deletion simulation data of human chromosome 21. **a** Start position deviations for PopDel on 1704 deletions. **b** Start position deviations for Delly on 1686 deletions. **c** Start position deviations for Smoove on 1653 deletions. **d** Start position deviations for Manta on 1677 deletions. A deviation below 0 indicates that the estimate is upstream of the true position, a positive deviation indicates that the estimate is downstream of the true position.

estimate the starting positions mostly exact: PopDel estimates the starting positions of 71.1 % of its variants calls with a deviation of 0 bp from the real starting positions. Delly (72.3 %), Smoove (71.2 %) and Manta (72.2 %) reach marginally higher values, despite explicitly considering split reads alignments in their methods. The deviations of all tools are mostly limited to one direction. PopDel and Smoove estimate the imprecise starting locations mostly further downstream, while Delly and Manta estimate imprecise starting locations to be further upstream. The results reflect PopDel's approach for the estimation of the deletion starting positions described in Subsection 3.3.3: As PopDel takes the 80 %-quantile of the clipped 3'-mapping positions of the forward reads of read pairs that support the deletion, an overestimation of the positions (i.e. a shift downstream) is more likely to occur than an underestimation. Local ambiguities in the alignments, which for example can be caused by repeats around the breakpoint, can lead to such deviations.

Running Time and Memory Consumption. Running time and memory consumption of the tools were measured on the uniform deletion simulation cohort for all batches of different sample numbers using the Unix */usr/bin/time* command. Measurements were

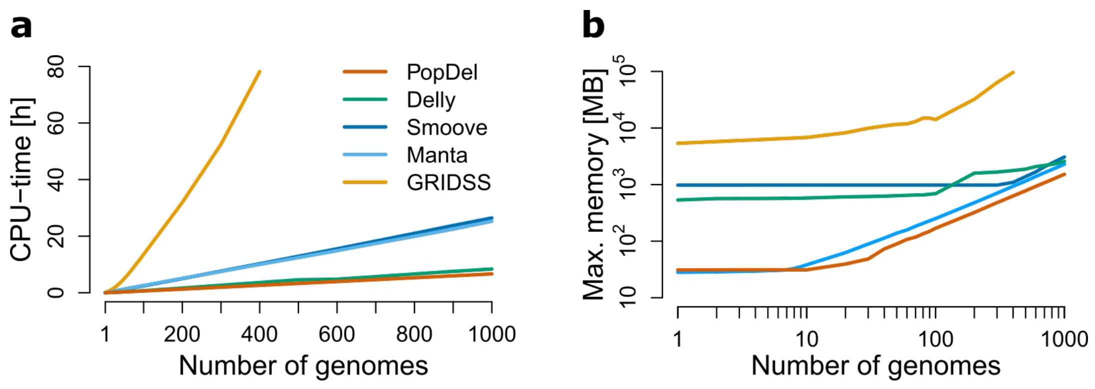

Figure 4.6: Running time and memory consumption on uniform deletion simulation data. **a** CPU time for increasing number of simultaneously analyzed samples. **b** Maximum memory consumption for increasing number of simultaneously analyzed samples.

performed on a dedicated workstation to avoid the influence of other running processes. The workstation was equipped with an Intel Xeon E5-1630v3 8 × 3.5 GHz and 64 GB RAM. The measured time includes the running time of all processes required to create the final VCF file from the input BAM files of the batch and was calculated as the sum of CPU seconds the process spent in user mode ($\%U$) and the number of CPU seconds spent in kernel mode ($\%S$). If a tool's workflow consisted of multiple steps (e.g. PopDel's profiling and calling steps), the time was calculated as the sum of the time all steps required. Memory was measured as the maximum resident set size ($\%M$) at any point of the workflow for the respective number of samples.

As shown in Figure 4.6a, running time of all tools scales linearly in the number of simultaneously analyzed samples. With 22 s CPU time for a single sample and 6 h 39 min for 1,000 samples, PopDel has the lowest running time for all batches, followed by Delly, which takes 38 s for one sample and 8 h 23 min for processing 1,000 samples. Manta and Smoove take almost four times as long for processing 1,000 samples (25 h 17 min and 26 h 26 min respectively). GRIDSS' running time is far above all other tools, taking 78 h 9 min for 400 samples, which is the maximum number of samples GRIDSS could process.

Figure 4.6b shows the memory consumption of the tools. For up to ten samples the memory consumption of PopDel's calling step is below that of the profiling step, which is why it is virtually constant at 31 MB for batches of up to ten samples. For 20 samples and above, the joint calling dominates the memory consumption which is now increasing in a linear fashion. PopDel's maximum memory consumption peaks at 1.5 GB at 1,000 samples. The initial memory consumption of Manta is slightly lower (28 MB), but shows a stronger increase after 8 samples, peaking at 2.3 GB for 1000 samples. This is below the values achieved by Delly (2.6 GB) and Smoove (3.1 GB). Delly and Smoove both require a high initial memory of 534 MB and 979 MB respectively. With 5.3 GB, GRIDSS' memory consumption has the highest initial value, orders of magnitude above that of PopDel or Manta, and also has a much higher peak memory consumption of 96 GB at 400 samples. A possible explanation for this amount of measured memory consumption that is above the physically available memory of the machine used for running the tools is that the Java virtual machine (JVM) GRIDSS is running on causes some memory pages to be counted multiple times, causing an overestimation of the resident set size. This counting problem of shared memory pages is a known shortcoming of the resident set size as a measurement. An alternative (which was not applied in this benchmark) would be the application of the proportional set size instead. The tables of running time and memory consumption can be found in Section A.6 of the Appendix.

Summarizing, the results show that PopDel's requires less memory and CPU time than other tools. Further, PopDel exhibits good scaling of both resources for increasing sample numbers.

Measurements for Varying Buffer Sizes. For increased flexibility, PopDel provides a parameter for controlling the number of buffered windows during the joint calling. Despite PopDel's already low memory requirements, it might be desirable to reduce the number of buffered windows for a further reduction of memory consumption when processing very large cohorts with limited available RAM. This comes at the cost of an increased running time, as shown in Figure 4.7. For the quantification of the memory and CPU time trade-off, 100 samples of the uniform deletion simulation data were analyzed with *PopDel call* while applying settings ranging from 10,000 to 8,000,000 windows for the buffer size. As marked by the filled red dot in the figure, the default value of 200,000 windows strikes a balance between CPU time and memory consumption. Compared to the memory consumption using default settings, it was possible to reduce the required memory by 87 % (22 MB compared to 187 MB) at the cost of an 341 % increase in CPU time (47 min compared to 11 min). Increasing the number of buffered windows had little effect in this example due to the small size of chromosome 21: Given the approximate size of 48,000,000 bp for chromosome 21, a window size of 30 bp and a buffer size of 200,000 windows, one buffer size contains already $\frac{1}{8}$th of the whole chromosome. This leaves only little room for reducing the overhead caused by clearing and filling the buffer multiple times. Buffer sizes above 1,600,000 windows can already contain the whole information for chromosome 21, virtually causing the whole profiles of the 100 samples to be loaded into the main memory. Further increasing the buffer size therefore has no effect on memory consumption and CPU time. For bigger chromosomes, this point is reached later. A detailed table of the measured results can be found in Section A.6 of the Appendix.

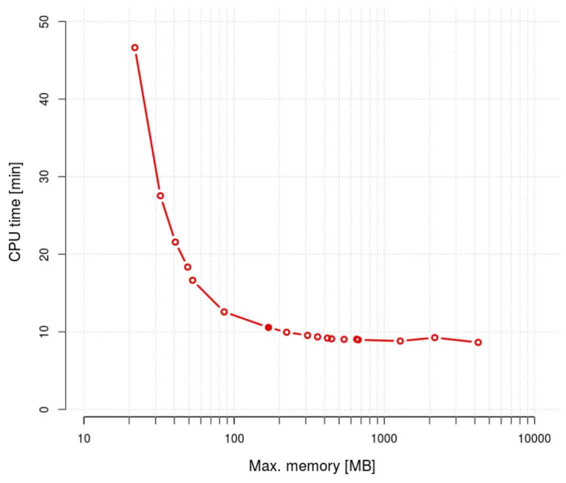

Figure 4.7: Memory and CPU time trade-off of PopDel's joint calling procedure. Joint calling was performed on 100 samples of the uniform deletion simulation data of human chromosome 21 with varying settings for the number of buffered windows. The filled red dot indicates the default value for the number of buffered windows (200,000).

4.3.3.2 Results for Simulated Duplication Data

Precision and recall on the simulated duplication data were measured as described in Subsection 4.1.2 on the filtered call sets for increasing batch sizes. Evaluations were performed with and without consideration of the predicted genotypes.

Without Consideration of Genotypes. All SV callers achieve a near-perfect precision when evaluating the presence or absence of duplications in the call sets. Only Manta calls between one and two false positive duplications for batch sizes between five and ten samples. Therefore, no plot of precision is shown. The near-perfect precision of all SV callers is indicative of the relative simplicity of the simulated data and of the clear read pair signature the SV callers use for the detection of duplications. Detailed tables of the measured results can be found Section A.4 of the Appendix.

Figure 4.8a shows the recall for Delly, Manta, PopDel and Smoove. PopDel achieves the highest recall, ranging from 0.9 to 0.91. This is closely followed by Delly's recall, which lies between 0.89 and 0.90. The recall of Manta lies between 0.87 and 0.88. Smoove's recall is lowest and lies between 0.81 and 0.85. There is no indication of a strong influence of the number of samples and all SV callers exhibit a stable performance.

With Consideration of Genotypes. When considering the ratio of correctly genotyped alleles , Smoove achieves the highest ratio, which lies between 0.81 and 0.83 as shown in Figure 4.8b. Considering that Smoove has the lowest recall, this indicates that the majority of the duplications detected by Smoove are correctly genotyped in most sam-

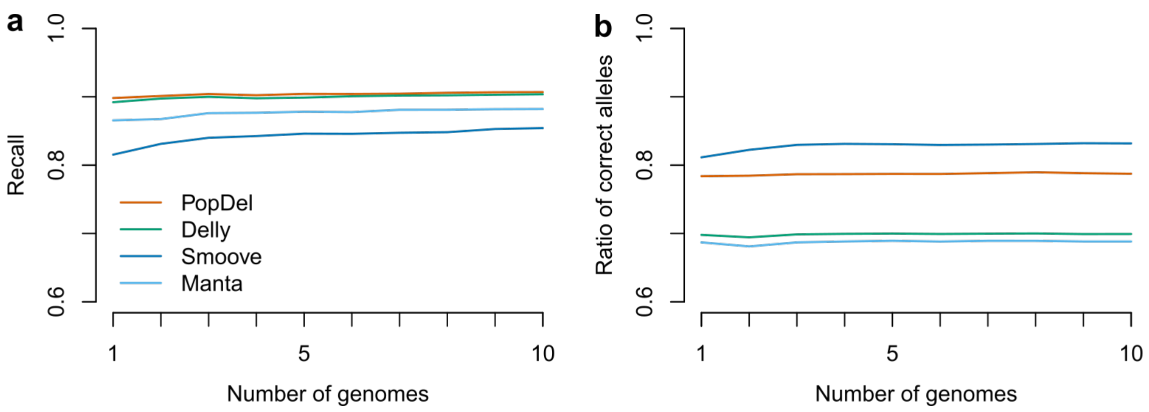

Figure 4.8: Precision and recall on simulated duplication data of up to ten samples (1,900 duplications). **a** Recall of duplication detection without consideration of genotypes. The plot for the precision is not shown because all tools achieved a near-perfect precision for all batch sizes. **b** Ratio of correctly classified alleles.

ples. With a ratio of correct alleles between 0.78 and 0.79 PopDel achieves the second best performance. Notably, Delly and Manta do not genotype any duplications as 1/1 in any of the samples. They therefore achieve the lowest and second lowest ratio of correct alleles. Since Manta has an overall lower recall than Delly, its ratio of correct alleles is lowest with values between 0.68 and 0.69. Delly achieves ratios between 0.69 and 0.7. Detailed tables of the measured results can be found Section A.5 of the Appendix.

The relatively low ratio of correctly predicted alleles achieved by all SV callers underlines the difficulty of estimating the zygosity of duplications and points out potential for future improvements of PopDel.

Running Time and Memory Consumption. Running time and memory consumption were measured as described for the uniform deletion simulation (see Subsection 4.3.3.1) for Delly, Manta, PopDel and Smoove. The absolute measurements of the duplication simulation are not directly comparable with those of the uniform deletion simulation. The running time of the duplication simulation was measure on a virtual machine that was provided 10422 MB of RAM and 4 out of 12 CPUs (Intel Core i7-8700k 12 × 3.7 GHz) of the host system. The detailed tables of the running time and memory consumption are part of Section A.6 of the Appendix.

Figure 4.9a shows the measured CPU time for processing batches of one to ten samples of the duplication simulation data. One can observe the same trends as shown for the uniform deletion simulation (Figure 4.6a). All compared tools show a linearly increasing CPU time with respect to the number of simultaneously processed samples. PopDel has the lowest CPU time for all batch sizes, linearly increasing from 20.87 s for a single sample to 214.54 s for ten samples. The next lowest CPU time is achieved by Delly (54.66 s to

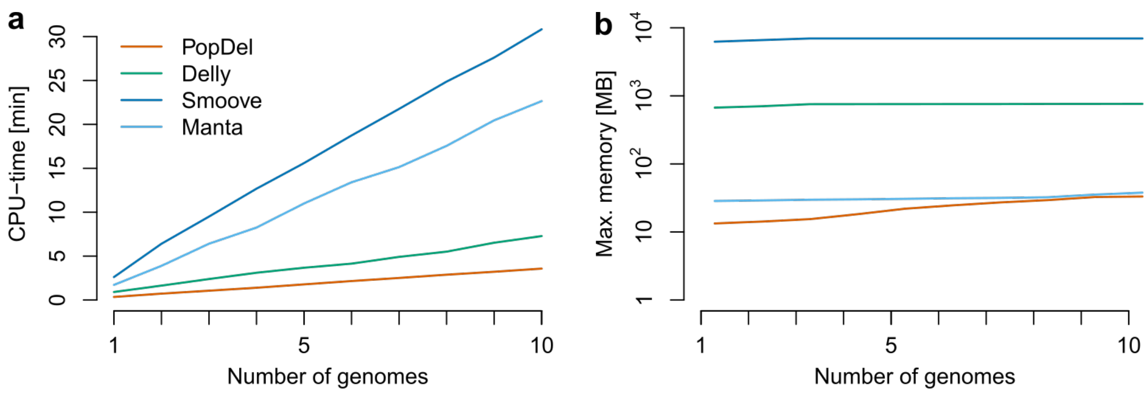

Figure 4.9: Running time and memory consumption on duplication simulation data. **a** CPU time for increasing number of simultaneously analyzed samples. **b** Maximum memory consumption for increasing number of simultaneously analyzed samples.

436.85 s), which takes on average (arithmetic mean) 2.04 as many CPU seconds as PopDel. Manta and Smoove take considerably more CPU time than PopDel and Delly. Manta's CPU time lies between 103.24 S for one sample and 1359.09 s for ten samples. Smoove takes the most CPU time, with values ranging from 156.60 s to 1849.85 s.

Figure 4.9b shows the observed maximum residual set size of the tools when processing batches of one to ten samples of the duplication simulation data. The observed trends are again similar to those shown in Figure 4.6b for the uniform deletion simulation data. PopDel has the lowest memory consumption, requiring only between 13.34 MB and 33.2 MB of RAM. Manta's memory consumption amounts to values between 28.52 MB and 37.77 MB. Delly's memory consumption is one order of magnitude higher than that of PopDel or Manta: It requires between 671,19 MB and 762.45 MB RAM. Smoove requires the most memory. With 6251.33 MB to 6967.99 MB RAM it occupies one order of magnitude more memory than Delly and two orders of magnitude more than PopDel or Manta. All tools show little increase of required RAM for increasing batch size, which is most likely attributed to the low number of samples in this scenario. A more detailed analysis of the scaling of memory consumption would require more samples, as it was the case for the uniform deletion data simulation in Subsection 4.3.3.1.

4.3.3.3 Results for Simulated Inversion Data

Precision and recall on the simulated inversion data were measured as described in Subsection 4.1.2 on the filtered call sets for increasing batch sizes. Evaluations were performed with and without consideration of the predicted genotypes.

Without Consideration of Genotypes. All tested SV callers achieve a perfect precision when evaluating the presence or absence of inversions in the call sets. Therefore,

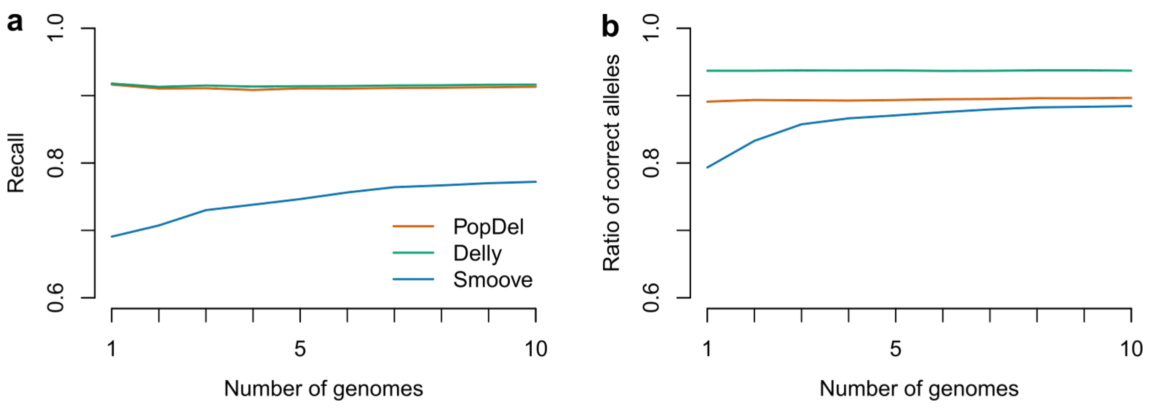

Figure 4.10: Evaluation on simulated inversion data of up to ten samples (1,900 inversion). **a** Recall of inversion detection without consideration of genotypes. The plot for the precision is not shown because all tools achieved perfect precision for all batch sizes. **b** Ratio of correctly classified alleles.

no plot of precision is shown. The perfect precision of all SV callers is indicative of the relative simplicity of the simulated data and of the clear read pair signature the SV callers use for the detection of inversions.

The highest recall is achieved by Delly (0.913 to 0.918), closely followed by PopDel (0.908 to 0.916) as shown in Figure 4.10a. Smoove's recall is considerably lower with values between 0.691 and 0.772. Upon manual inspection of Smoove's call sets, it was noticeable that Smoove classified up to 18 of its calls as breakends instead of inversions. These calls were ignored during the evaluation. Even if one were to consider these breakend calls as true positives this would only increase Smoove's recall by roughly 0.01. Detailed tables of the measured results can be found Section A.4 of the Appendix.

All tested SV callers except Smoove show a stable performance across all batch sizes. Smoove performs better for larger batch sizes but still achieves lower performance metrics than the other SV callers.

With Consideration of Genotypes. Figure 4.10b shows the ratio of correct allele predictions made by Delly, PopDel and Smoove. The highest ratio of correct alleles is achieved by Delly with values between 0.937 and 0.938. PopDel follows with ratios between 0.89 and 0.9. Smoove's ratio of correct alleles lies between 0.793 and 0.884 and is therefore lowest.

The ratios are stable across all batch sizes for all tools except Smoove. Smoove's increasing ratio of correct alleles for larger batch sizes could be explained by its increasing recall for larger batch sizes as observed in Figure 4.10a. Detailed tables of the measured results can be found Section A.5 of the Appendix.

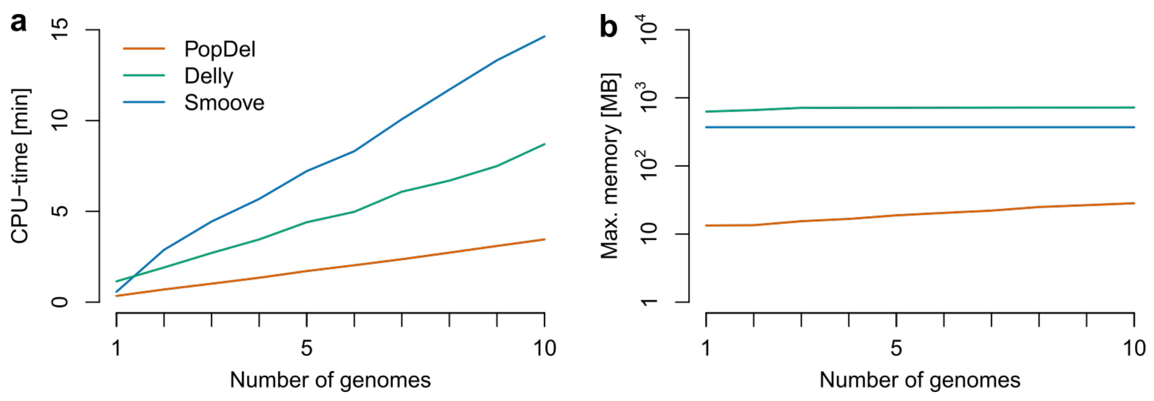

Figure 4.11: Running time and memory consumption on inversion simulation data. **a** CPU time for increasing number of simultaneously analyzed samples. **b** Maximum memory consumption for increasing number of simultaneously analyzed samples.

Running Time and Memory Consumption. Running time and memory consumption were measured as described for the uniform deletion simulation (see Subsection 4.3.3.1) for Delly, PopDel and Smoove. The absolute measurements of the inversion simulation are not directly comparable with those of the uniform deletion simulation. The running time of the inversion simulation was measure on a virtual machine that was provided 10422 MB of RAM and 4 out of 12 CPUs (Intel Core i7-8700k 12 × 3.7 GHz) of the host system. The detailed tables of the running time and memory consumption are part of Section A.6 of the Appendix.

Figure 4.11a shows the measured CPU time for processing batches of one to ten samples of the inversion simulation data. The reduced CPU time of all tools is a notable difference to the results of the duplication simulation. This could be explained by the small size of the simulated samples and the high number of variants that were inserted into them. While an inserted inversion does not alter the quantity of sequence in a genome, a duplication introduces additional sequence. This is reflected by the file size of the simulated samples: For example, the first simulated sample of the inversion and duplication simulation data are identical with respect to the number and locations of inserted variants. They only differ in the type of variants that were inserted. The BAM file of the simulated inversion sample takes up 592 MB of hard disc space. Due to the inserted duplications the simulated duplication sample amounts to 661 MB, which is an increase of 11.65 %.

Despite the overall reduced CPU time, the general trends of the duplication and deletion simulation benchmarks are also reflected in the measured CPU time of the inversion simulation displayed in Figure 4.11a. PopDel requires between 20.54 s and 207.16 s for batch sizes of one to ten samples. Smoove is the second fastest tool for one sample, taking 34.07 s, but requires more time than Delly for all subsequent batch sizes. With 878.18 s CPU time, Smoove is the slowest of the three tools for ten samples. Delly is the overall second fastest tool, requiring between 68.61 s and 521.95 s of CPU time. As in the previous test scenarios, the CPU time of all tested tools increases linearly with respect to the number of samples per batch.

Figure 4.11b shows the maximum residual set size of the tools when processing batches of one to ten samples of the inverted simulation data. PopDel requires between 13.36 MB and 28.34 MB RAM for one to ten samples, making it one order of magnitude faster than Smoove and Delly. Smoove's required memory is very constant with values around 370.41 MB for all batch sizes. Similarly, Delly shows only very little increase of its required memory: Its values range from 628.94 MB to 721.828 MB RAM. Again, this indicates that more samples are required for a full assessment of the scaling of the tools, like it has been performed for the uniform deletion simulation in Subsection 4.3.3.1.

4.4 Genome in a Bottle Ashkenazi Jewish Trio

Because simulated data cannot fully capture the challenges of real WGS data, additional benchmarks on well-studied WGS samples has been performed. One such data set is the Ashkenazi Jewish trio that has originally been sequenced in the Personal Genome Project[5]. The son of the trio, also known under its NIST ID HG002 or his Coriell ID NA24385, has been thoroughly characterized and used by the GIAB consortium for creating a set of SVs that is comprehensive in well-defined regions of the genome[92]. This allows the assessment of metrics like precision and recall, which would be much less reliable without a comprehensive ground truth. Further, the presence of the parental genomes allows for the assessment of how the calling of variants in HG002 is influenced by the parental genomes. The setup of the different tools in the tested scenarios will be described in the following subsections. Finally, the results of the deletion benchmarking on this data set will be presented and discussed.

A similar comparison was performed using NGS read data of HG001/NA12878. As the results are very similar to the ones presented here, they are part of Section A.7 of the Appendix.

4.4.1 Single Sample Variant Calling Protocol

The reads of HG002 were mapped to the human reference genome GRCh37 using BWA-mem. GRCh37 was chosen over GRCh38 because the GIAB reference call set is based on GRCh37. Shorter split read alignments were flagged by applying the BWA-mem's '-M' option. The conversion of the resulting SAM files to BAM files, sorting and indexing was carried out using *Samtools*[58]. *Picard Toolkit*[4] was used for marking optical duplicates.

[4] *http://broadinstitute.github.io/picard/*

SV detection was performed by Delly, Manta, PopDel and Smoove on the resulting BAM file as follows:

Delly was executed using its recommended options and the recommended file of excluded regions for GRCh37. Only deletions with the FILTER field set to "PASS" were kept.

Manta was applied using its recommended options. The calling was prepared with *configManty.py,* followed by the execution in local mode. Only deletions from the resulting output file containing high confidence variants with the FILTER field set to "PASS" were kept.

PopDel read pair profiles were created using *PopDel profile* (1.2.2) with default options. The variant calling by *PopDel call* (1.2.2) was performed on all autosomes in parallel using the '-r' option (see Appendix Section A.1 for on overview of PopDel's parameters). Since the coverage of the Ashkenazi Jewish trio is very high, PopDel's threshold for when a region is considered to have high coverage was increased to three times the mean coverage of the sample using the '-A' option. No additional filtering based on the FILTER field was performed.

Smoove was run with the recommended options and the recommended file of excluded regions for GRCh37. No additional filtering based on the FILTER field was performed.

4.4.2 Trio Variant Calling Protocol

For the trio variant calling benchmark the reads of the parents of HG002 (HG003, father and HG004, mother) were aligned and processed the same way as those of HG002 described above to obtain the respective BAM files. Calling on the trios was performed with the tools Delly, Manta, PopDel and Smoove the same way as for the single-sample variant calling, with the exception that all BAM files of the trio were processed jointly.

4.4.3 Comparison Protocol

The comparison on the single and trio GIAB call sets consisted of an analysis of the overlaps of the call sets created by the different SV callers and the reference deletions provided by GIAB. Before performing the individual analyses, all call sets were processed to only contain the relevant information. Therefore, non-deletion variants were removed from all call sets prior to evaluation and only deletions with a size between 500-10,000 bp were considered. Further, the deletions were required to be located on one of the autosomes. Overlap of the deletions with the GIAB high confidence regions for HG002 regions was determined using *BEDtools intersect*[71] and all deletions not completely falling into those

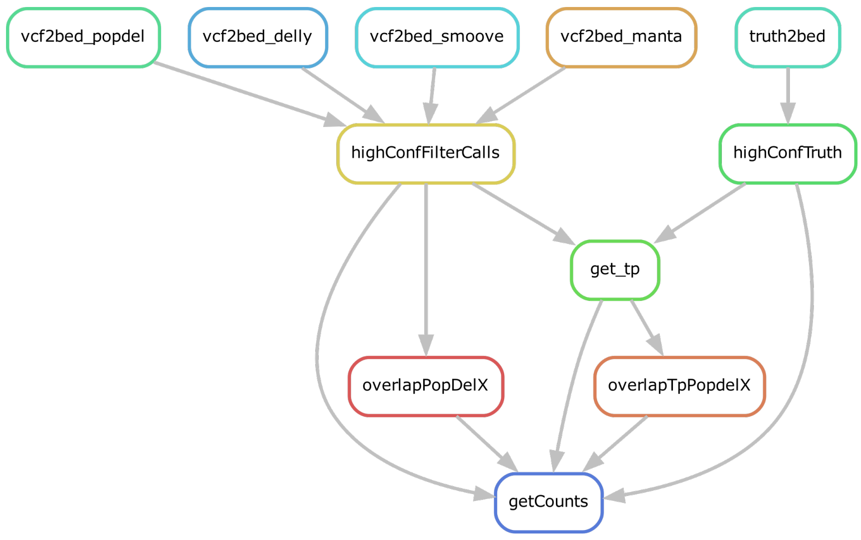

Figure 4.12: Rule graph generated by Snakemake for the determination of call set overlaps. Vertices indicate processing and evaluation functions, while arrows indicate which steps use the results generated by the previous step.

regions were removed. Finally, the VCF files were converted into BED format for easier downstream processing.

To examine to what degree the sets of deletions created by the different SV callers agree with each other and the ground truth, Venn diagrams of the call set overlaps were created. To this end, the overlaps of the call sets were determined in a hierarchical overlap approach using BEDtools. For matching two variants, the definition given in Subsection 4.1.1 requiring at least 50 % reciprocal overlap was applied. *Snakemake*[49] was used for the management of the complex evaluation workflow. An overview of the workflow is given in Figure 4.12.

4.4.4 Results

Call Set Overlap. The resulting Venn diagrams displayed in Figure 4.13 show that the four SV callers reach a good consensus of detected deletions. PopDel shares 582/639/616 of its predicted variants with Manta/Smoove/Delly. The highest overlap with the ground truth, i.e. true positives, is achieved by PopDel which correctly predicts 62/26/48 deletions that Manta/Smoove/Delly do not detect. Those tools detect 4/4/7 true deletions not detected by PopDel. There are only 40 out of 678 deletions in the truth set that are not detected by any of the tested SV callers. The lowest number of predicted variants that are not part of the truth set, i.e. false positives, is produced by Manta, which predicts 27 deletions that are not contained in the truth set due to its more conservative approach that also comes with a lower number of detected true deletions.

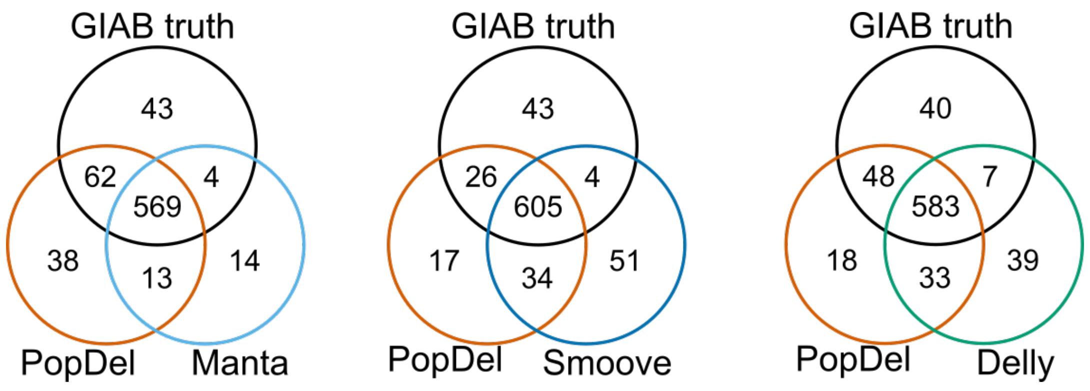

Figure 4.13: Call set overlaps of different SV callers with the GIAB reference call set for HG002 consisting of 678 high-confidence deletions. Numbers in the overlapping areas of the circles indicate that the deletions were detected in all of the overlapping call sets.

The predicted deletions result in a recall of 93.01 % (PopDel), 84.51 % (Manta), 89.82 % (Smoove*)* and 87.02 % (Delly). The measured precision is 92.52 % (PopDel), 95.5 % (Manta), 87.75 % (Smoove) and 89.12 % (Delly). The F_1-scores based on these values are 0.93 (PopDel), 0.9 (Manta), 0.89 (Smoove) and 0.88 (Delly), showing a good balance between precision and recall achieved by PopDel.

These results demonstrate that PopDel good accuracy measure on simulated data also applies to real data. The results are also reflected in the precision and recall analysis of the following section.

Precision and Recall for Varying Genotype Quality Thresholds. When filtering the predicted variants for increasingly strict genotype quality (GQ) thresholds, one can observe the well-known trade-off between precision and recall. The GQ provides a PHRED-scaled confidence measure for the correctness of the assigned genotype. A higher GQ threshold leads to an increase in precision and a decrease in recall for all examined SV caller. This indicates that the GQ is implemented in a meaningful way. On the other hand, relaxing the threshold leads to an increase in recall at the cost of a reduced precision. Finding the optimal balance between the two measures often depends on the individual use case. Figure 4.14 shows this behavior for the variants detected by Delly, Manta, PopDel and Smoove for the variants predicted on HG002 in a single-sample calling setup (dashed lines) or together with the parental genomes in a trio-setup (solid lines).

A risk of including additional genomes in the calling is that this can also increase the risk of false positive calls being made. Smoove and Delly both follow a merge and re-genotype approach (see method descriptions of Smoove and Delly in Section 2.3) and exhibit a lower precision when applied on the whole trio compared to the single-sample setup. This is most likely because the calling of the variants per sample followed by the merging leads to

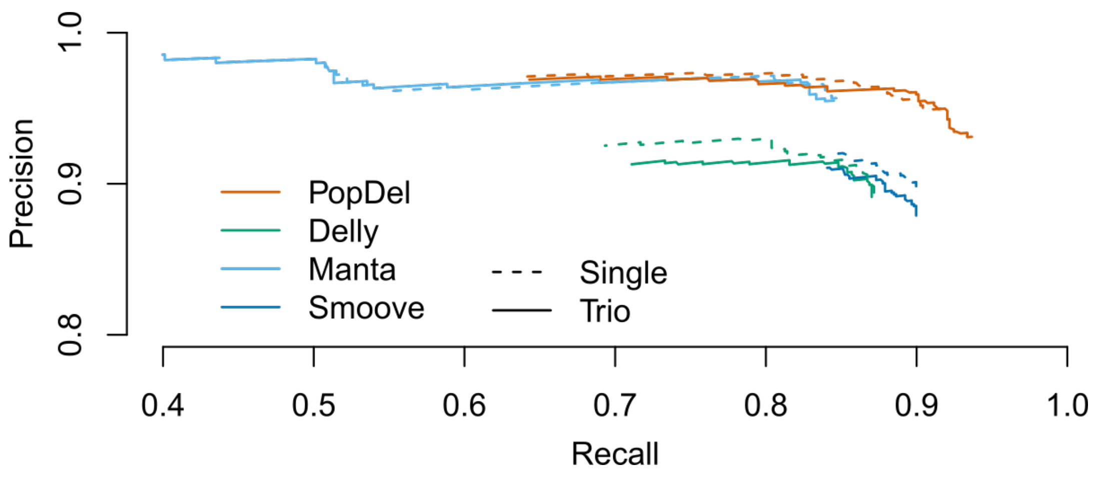

Figure 4.14: Precision and recall of SV callers with GIAB reference call set for HG002 alone (dashed lines) and jointly (solid lines) with his parents for varying genotype thresholds.

a bigger set of candidate variants than the variant calling on HG002 alone. Some of these additional candidate variants are only present in the parental genomes and not in HG002. With the aim of increasing the sensitivity, the re-genotyping phase places more relaxed thresholds on variants than the calling phase. This is because a that variant has already been detected in another genome, is more likely to be also real in the genome that is being (re-)genotyped. On the analyzed data, Smoove and Delly do lose some precision when operating on the while trio, without gaining additional recall. Delly loses up to 1.64 %-points in precision and Smoove loses up to 0.84 %-points in precision. Manta's precision and recall are hardly affected by the addition of the parental genomes and also PopDel's precision does not decrease when performing the calling on the whole trio. Notably, only PopDel's recall shows an increase in the trio-setup, underlining the benefits of the chosen joint calling approach.

4.5 Polaris Diversity Cohort

The Polaris HiSeqX Diversity Cohort (PDC; BioProject accession PRJEB20654) was created by Illumina in an effort to provide a publicly available resource for population-genomics analyses. It consists of 150 WGS samples that have been sequenced with 2x150 bp reads using a Illumina HiSeqX sequencer. The samples were selected from the resources of the 1000 Genomes Project and originate from three different continental groups, 50 samples each: African (AFR), East Asian (EAS) and European (EUR). This multi-population setup enables the evaluation based on deletion allele count per population and principal component analysis, as described in Subsection 4.1.4.

4.5.1 Variant Calling Protocol

The data of the 150 genomes was retrieved in FASTQ format and mapped to the human reference genome GRCh38 in the same fashion as described in Subsection 4.4.1 using BWA-mem. Downstream processing was performed using Samtools and Picard Toolkit. Variant calling was performed using PopDel, Delly and Smoove. Manta was excluded from this analysis because it did not complete the analysis within four weeks of compute time and no description of a suitable single-sample based workflow with subsequent merging and re-genotyping was available for Manta.

Delly could not complete the joint calling on all 150 samples together within four weeks of compute time. Therefore, a sample wise approach with subsequent merging and re-genotyping was applied instead:

1. Single-sample variant calling for each individual sample using the '*-n*' option for disabling small indel realignment
2. Removing all non-deletion variants using *BCFtools*[19] *filter*
3. Merging all variants of all samples using *Delly merge*
4. Sample-wise genotyping of all samples using *Delly genotype* and the variant set create in the previous step
5. Removing all variants with invalid positions
6. Sorting and indexing of the variants
7. Merging of the genotyped and sorted variants of all samples using *BCFtools merge*
8. Application of a germline filter using *Delly filter* to remove variants that are flagged as likely somatic

Steps 5 and 6 had to be introduced due to a known bug in Delly [`[https://github.com/dellytools/delly/issues/106]`] that caused some variants to have '0' assigned as a position, making the required sorting and indexing of the BCF files impossible. Removal of those variants allowed the continuation of the workflow.

PopDel profile (1.2.2) was applied with default options on each sample to create the PopDel read pair profiles. Joint calling with *PopDel call* (1.2.2) was performed on all autosomes and the X chromosome in parallel using PopDel's *'-r'* option.

Smoove was applied using the approach recommended by its authors in the same way as described for the simulated data in Subsection 4.3.2, with the following exceptions: Only the autosomes and the X chromosome were considered and all non-deletion variants were removed after the single-sample calling.

4.5.2 Comparison Protocol

Like for the comparison on the GIAB data, only deletions between 500 and 10,000 bp were considered. Since no high-confidence regions like for the GIAB data were applied for filtering, variants having any overlap with centromeric regions were determined and removed using *BEDtools intersect.* The BED-file of centromeric regions for GRCh38 was obtained via the UCSC table browser (group: *Mapping and Sequencing*, track: *Centromeres*).

For the analysis of the variant counts, the variant call sets were subject to genotype quality (GQ) filtering based on thresholds that were determined using the Mendelian inheritance error rate on the PKC described later in Section 4.6. Briefly, the thresholds for the GQ were selected as generous as possible such that the MIER dropped below 0.3 %. Those thresholds for the GQ were 26 (Popdel), 28 (Delly) and 78 (Smoove). All deletions with a GQ below the respective threshold were removed from the call sets in order to focus the analyses on variants of high quality.

4.5.3 Results

Deletion Count. Figure 4.15 shows the observed distributions of detected deletions for the three compared SV callers across the three continental groups.

PopDel (Figure 4.15a) detects on average (arithmetic mean) 1,309 deletions per sample of the PDC. 969 of the deletions detected by PopDel are genotyped as heterozygous and 340 as homozygous. PopDel detects significantly more deletions in African (AFR) samples on average than in East Asian (EAS) or European (EUR) samples, as a two-sided t-test with a resulting p-value $< 2.2 \cdot 10^{-16}$ confirms. This finding is consistent with the expected distribution of variants described in Subsection 4.1.4.

Delly (Figure 4.15b) detects on average 1,300 deletions per sample of the PDC. 1,011 of Delly's deletions are genotyped as heterozygous and 289 as homozygous. The expected higher abundance of deletions in AFR samples can also be observed for Delly.

Smoove (Figure 4.15c) detects on average 1,233 deletions per sample of the PDC. 981 of Smoove's deletions are genotyped as heterozygous and 252 as homozygous. The expected higher abundance of deletions in AFR samples can also be observed for Smoove.

All compared SV callers detect a similar number of deletions per genome and per population and show the same trends. As one would expect, the number of heterozygous calls is lower

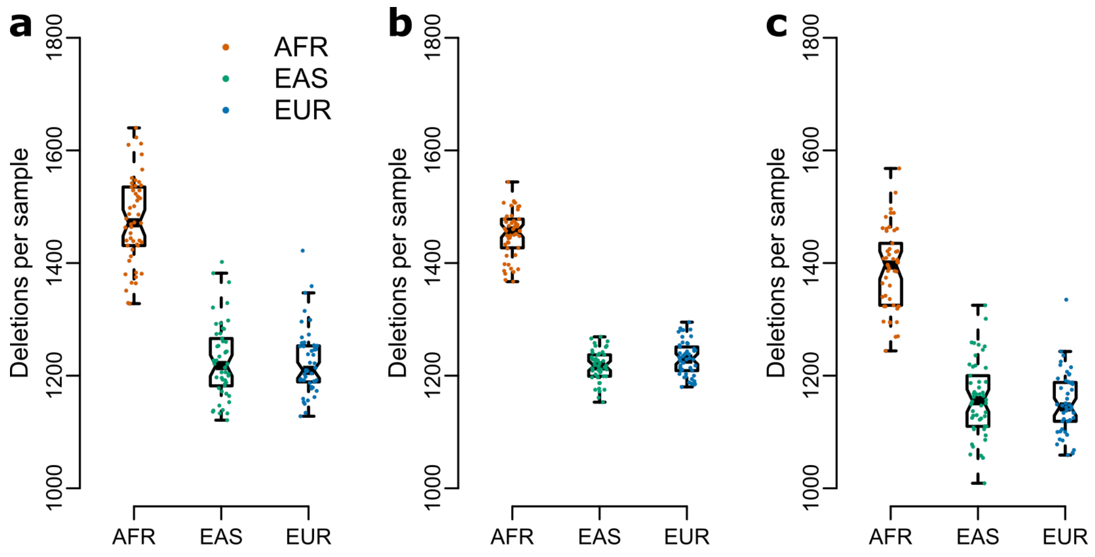

Figure 4.15: Deletion counts per population in the Polaris Diversity Cohort. Each data point indicates the number of deletions detected in one genome. Center lines of the box plots indicate the median for the respective population. Boxes limit the upper and lower quartiles. The whiskers denote the 1.5x interquartile ranges. Vertically non-overlapping notches in the boxes indicate a strong evidence that the groups' medians differ. Colors and abbreviations indicate the continental groups of the samples: AFR, African; EAS, East Asian; EUR, European. **a** PopDel deletion counts. **b** Delly deletion counts. **c** Smoove deletion counts.

than the number of homozygous calls. Notably, the number of heterozygous deletions detected by PopDel is lower than the number of heterozygous calls given by Delly and Smoove, while its number of homozygous calls is higher. Together with the observation that the ratio of heterozygous variants divided by homozygous variants is rather $\frac{1}{3}$ than the theoretically expected $\frac{1}{4}$, this can be indicative of a bias towards genotyping samples as homozygous instead of heterozygous in PopDel. Further examinations of the genotyping performance on the trios of the Polaris Kids Cohort are presented in Section 4.6.

Principal Component Analysis. When calculating the PCA of the deletion allele counts produced by PopDel, Delly and Smoove as described in Subsection 4.1.4, the results for all three SV callers are very similar, as shown in Figure 4.16. There is a clear separation of AFR samples from EAS and EUR samples driven by the first principal component and a further clear separation of the EAS samples from the EUR samples driven by the second principal component. This leads to a very clear clustering of the samples by their continental group, indicating that the deletions calls of all three SV callers reflect the differences between the three populations of the cohort and showing how populations structure can be inferred from deletion alleles counts alone.

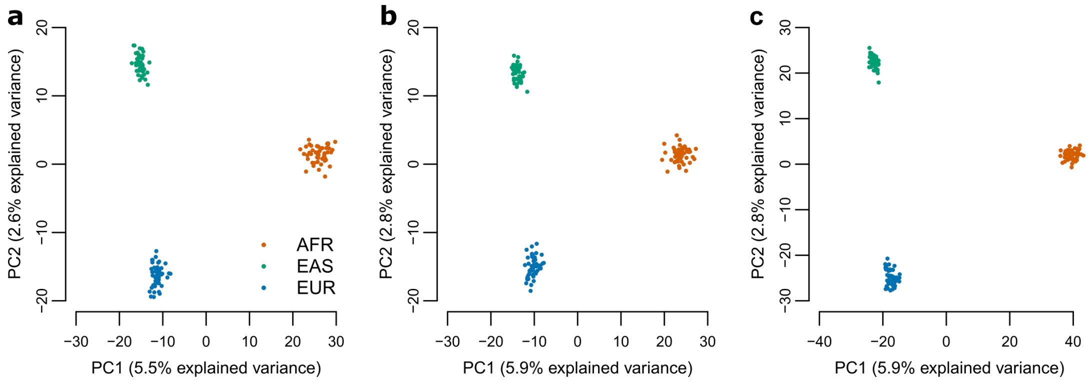

Figure 4.16: PCA based on deletion allele counts on the PDC. Colors and abbreviations indicate the continental groups of the samples: AFR, African; EAS, East Asian; EUR, European. **a** PCA based on PopDel deletion allele counts. **b** PCA based on Delly deletion allele counts. **c** PCA based on Smoove deletion allele counts.

Table 4.1: Running time on 150 WGS samples of the Polaris Diversity Cohort.

	Wallclock time	CPU time
Delly	389:43 h	371:27 h
PopDel	56:17 h	111:12 h
Smoove	87:58 h	103:21 h

Delly and Smoove also report SVs other than deletions. These non-deletion variants were removed after the single-sample calling phases of the respective workflows (see Subsection 4.5.1) to reduce the impact on the running time.

Running Time. Running time on the PDC was measured for the complete workflows of the SV callers. The BAM files of the samples were used as the starting input and the VCF files as the final output. The reported running time of all subprocesses was summed up to get the final result. PopDel's calling process was run in parallel for all analyzed chromosomes. Therefore, the running time for each chromosome was added. The resulting wallclock times and CPU times are displayed in Table 4.1. The runs were performed on a high-performance computing (HPC) cluster and may have been subject to fluctuations in compute cluster performance due to changes in concurrent work load of the HPC cluster.

PopDel completes the analysis of the 150 samples of the PDC within 2 days and 8 hours of wallclock time, requiring 4 days and 15 hours of total CPU time. This result is similar to that of Smoove, which takes 3 days and 16 hours of wallclock time, or 4 days and 7 hours of total CPU time. Consequently, both tools are several times faster than Delly, which requires 16 days and 9 hours of wallclock time (15 days and 11 hours of CPU time). The discrepancy in Delly's wallclock and CPU time might be explained by I/O bottlenecks that force Delly's process to wait for the data to be read from the drive and written into the main memory or vice versa.

Overall, the results show that PopDel performs very well in terms of running time. It allows the joint calling to be performed in short time even for large sample numbers, especially when one considers the easy and effective parallelization on the sample level (for the profiling process) or on the chromosome level (for the calling process). Owing to PopDel's low memory consumption demonstrated in the Paragraph Running Time and Memory Consumption of Subsection 4.3.3.1, this trivial parallelization is often feasible - even for high sample numbers. This can reduce the wallclock time to a few hours instead of days on most compute clusters.

4.6 Polaris Kids Cohort

While the PDC provides the possibility to inspect population structure, the Polaris HiSeqX Kids Cohort enables the inspection of inheritance patterns. It expands the PDC by 50 genomes of children whose parents are part of the PDC, thereby providing 50 family trios. Since one of the child genomes of the PKC (HG03170) was not available for download at the time of this thesis, all results presented here are based on the 49 complete trios.

4.6.1 Variant Calling Protocol

Mapping of the FASTQ files of the PKC genomes to the human reference genome GRCh38 was performed using the same tools and parameters as previously described for the PDC (see Subsection 4.5.1). In the subsequent SV calling, all 147 WGS samples of the 49 family trios were processed together. No pedigree information was provided to the SV callers. All SV callers were provided the same shuffled sample order to avoid any potential influence of the sample order. Variant calling and genotyping were performed using Delly, PopDel (1.2.2) and Smoove with the same workflows as for the PDC (see Subsection 4.5.1).

4.6.2 Comparison Protocol

For the comparisons of the predicted genotypes of the PKC only the autosomes were considered. This was done because in general only the genomes of biological females have two copies of the X chromosome, invalidating the assumption that a variant allele can be present in zero, one or two copies for the gonosomes of biological males. All call sets were filtered to only contain deletions of size 500 to 10,000 that did not have any overlap with centromeric regions (see 4.5.2). To avoid an inflation of the applied metrics, calls were further filtered based on their adherence to the HWE (see Subsection 4.1.3).

Mendelian Inheritance Error Rate. The call sets were subjected to a progressively strict filtering based on the genotype quality (GQ). A variant was considered to be present in a trio if at least one member of the trio had a heterozygous or homozygous variant genotype. A variant was removed for the whole trio if at least one member of the trio had a GQ below the currently tested threshold. Variants with undetermined genotypes for any member of a trio were also removed for that trio. For the assessment of the MIER the three genotypes of each trio were checked for their consistency with the Mendelian rules of inheritance as described in Subsection 4.1.3. If a variant violated those rules (see Equation 4.8) it was considered as *inconsistent* or an *error*.

Transmission Rate. The transmission rate was calculated using only deletions that were exclusive to single trios. This was done to assess the quality of genotypes for rare variants. Calculations were performed as described in Subsection 4.1.3, Equation 4.10.

***De Novo* Variants.** *De novo* variants describe variants that arise in the genome of the offspring without being present in the parental genomes and as such are relatively rare events. A study by Kloosterman et al. on 258 family trios reported *de novo* mutation rates of 2.94 indels (up to a size of 20 bp) per generation and 0.16 SVs (bigger than 20 bp) per generation[46]. Despite their lower rate of occurrence they affect more nucleotides (91 times as much) and even more coding bases (52 times as much) than indels do. When breaking down the mutation rates by SV type and size, one expects 1.33 *de novo* deletions of size 500-10,000 in the PKC.

To check for the presence of *de novo* deletions in the PKC among the SVs detected by PopDel, the GQ threshold was set to 50. Then, deletions with genotype combinations where the child carried an allele not present in any of the parents were extracted.

4.6.3 Results

Mendelian Inheritance Error Rate. Figure 4.17a demonstrates that Delly, PopDel and Smoove all achieve very low values for the MIER. When increasing the GQ threshold, both the number of consistent deletions per trio as well as the MIER decrease. This underlines the meaningful implementations of the GQ as a quality metric. All call sets can be filtered to achieve MIER values below 0.3 %. The threshold of 0.3 % MIER has been suggested in an early version of the GIAB benchmark paper by Zook et al. (2020)[92]. The most relaxed GQ thresholds that come with an MIER below 0.3 % are 26 (PopDel), 28 (Delly) and 78 (Smoove). This shows that the GQ values of the different tools, while all measuring the certainty in the predicted genotypes, are not directly comparable between different SV callers, as their calculations and scaling can vary.

PopDel exhibits a higher number of consistent deletions per trio than the other SV callers for the same MIER. This indicates an overall high quality of the genotypes calculated by PopDel. When filtering for a MIER just below 0.3 %, PopDel detects 1,177 consistent

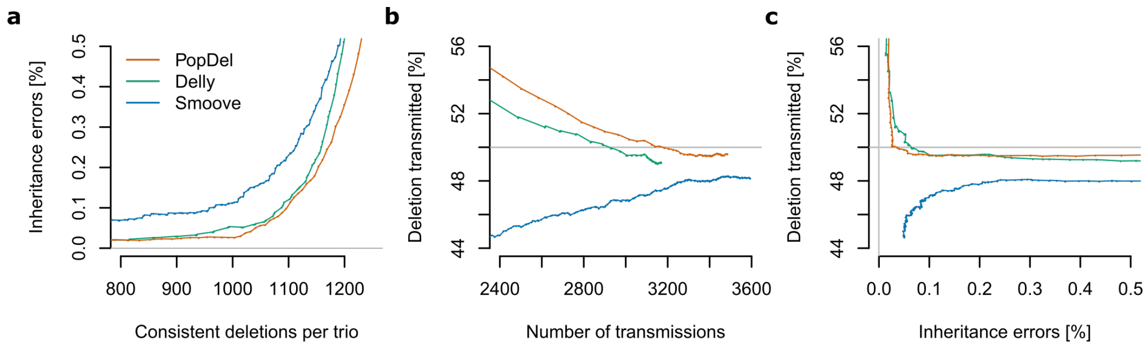

Figure 4.17: Genotyping evaluation on 49 trios of the PKC. Cell sets were filtered for increasing GQ thresholds. Grey lines indicate the theoretical optima. **a** MIER by number of consistent deletions per trio. **b** Transmission rate by the number of deletions that are unique to one trio and one parent. **c** Transmission rate by MIER.

deletions per trio on average (arithmetic mean). Delly and Smoove report 1,161 and 1,128 report consistent deletions respectively.

Transmission Rate. Figure 4.17 shows the transmission rate for deletions that were exclusive to single trios. When limiting the range of the number of transmissions to 2,349 - 3,595, which is the range in which all deletions detected by Smoove lie, PopDel's transmission rate lies between 49.47 % and 54.2 % with a median of 49.58 %. This is 0.41 %-points below the theoretical optimum. Delly's transmission rate in this range lies between 49.02 % - 52.81 % with a median of 49.52 %, being 0.48 %-points below the optimum. Smoove's transmission rate is generally lower, with values ranging from 44.66 % to 48.30 % and a median of 47.58 %, which is 2.42 %-points below the optimum. This is indicative of a slight undertransmission in Smoove's genotypes. On the other hand, it can be observe that Smoove's transmission rate is the most robust against overfiltering based on the GQs. As shown in Figure 4.18c, increasing the GQ threshold for the call set created by Smoove has a much smaller effect on the overall transmission rate than it has for PopDel and Delly. While Smoove's transmission rate always lies between 44.66 % and 48.29 %, PopDel's transmission rate shows a strong increase after a GQ threshold of about 50, going up to 92.27 %. Delly shows a stable transmission rate up to a GQ threshold of 70, after which it increases up to 71.43 %. This demonstrates how one has to be careful when increasing these thresholds more and more, hoping to only keep the presumed 'best' SVs.

These upper boundaries for the GQ filtering are not absolute in any way but heavily depend on the data. For example, the GQ reported by PopDel is calculated from the difference of the PHRED-scaled genotype likelihoods of the best and second best genotype. It thereby depends on the number of read pairs supporting the SV in question. This in turn depends on the overall coverage of the sample. Given a higher coverage sample, the GQ of genotypes with much evidence from the sample in question can be higher, shifting

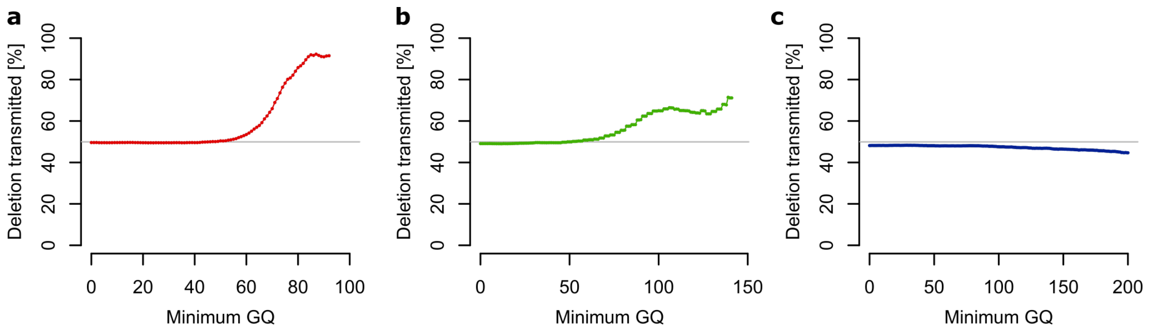

Figure 4.18: Transmission rate by GQ threshold on 49 trios of the PKC. Grey lines indicate the theoretical optimum of 50 % transmission rate. **a** Transmission rate for PopDel. **b** Transmission rate for Delly. **c** Transmission rate for Smoove.

the point of potential overfiltering to higher GQ values.

***De novo* Variants.** Filtering PopDel's deletion calls for *de novo* variants as described in Subsection 4.6.2 yielded a set of 12 *de novo* deletion candidates. The alignments of the family trios at the locations of the candidates were manually inspected using the Integrative Genomics Viewer[75](IGV). This inspection resulted in three likely real *de novo* deletions:

- A 8901 bp deletion at chr6:93,035,858–93,044,759 in HG01763.
- A 984 bp deletion at chr6:27,132,732–27,133,716 in an exon of the H2BC11 gene of HG01683.
- A 769 bp deletion at chr7:105,505,500–105,506,269 in an exon of the PUS7 gene of HG00615.

The deletion in HG01763 could further be confirmed and phased thanks to the presence of 25 SNVs at the locus of the deletion. As shown in Figure 4.19, both parents (HG01761, father and HG01762, mother) are heterozygous carriers of various SNVs within the locus of the deletion. Their daughter only exhibits homozygous SNVs in this locus, thereby confirming that she has one haplotype affected by the deletion and one unaffected haplotype that exhibits the same SNVs as present in one of the paternal haplotypes. Consequently, the deletion haplotype must have been inherited from the mother. The SNVs used for the phasing of the deletion are marked with small black arrows in Figure 4.19.

Since all individuals of the PKC originate from the resources of the G1k project, all samples are reported to be clinically healthy[17]. Therefore, non of the detected *de novo* variants is expected to be of medical relevance. This detection of 1-3 *de novo* deletions is well in line with the findings of Kloosterman et al.[46] and demonstrates how PopDel can be used to detect *de novo* deletions in larger cohorts.

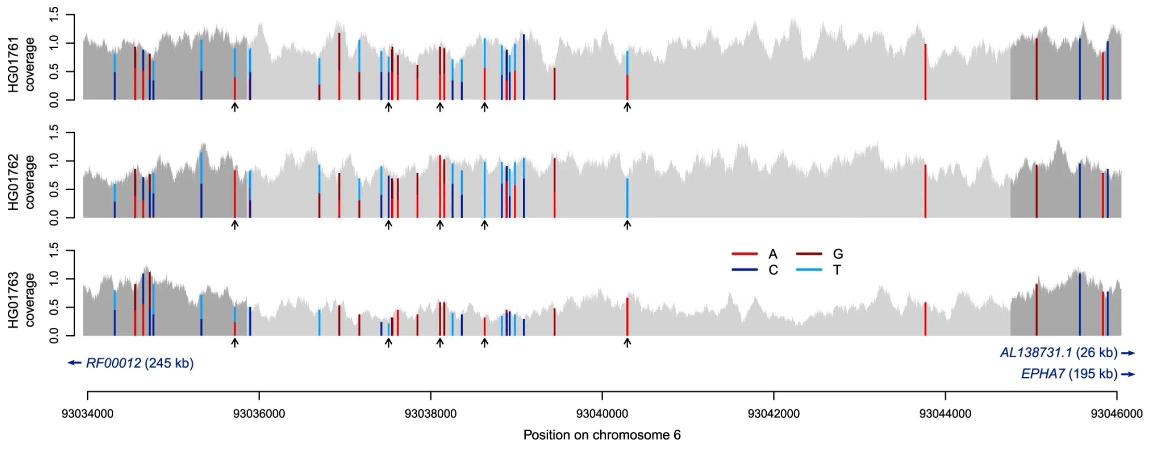

Figure 4.19: *De novo* deletion found by PopDel. The light gray interval highlights the location of the *de novo* deletion found in the child (HG01763, bottom). The father (HG01761, top row) and mother (HG01762, middle row) do not carry the deletion and are heterozygous carriers of various SNV at the locus of the *de novo* deletion. Colored bars indicate SNVs. Arrows mark SNVs that were used for the phasing of the variant.

4.7 Simons Genome Diversity Project

The Simons Genome Diversity Project [62] (SGDP) data set consists of the public WGS data of 300 individuals. As does the PDC, this data set aims at providing a resource for gaining insight into human diversity and population structures [62]. While the PDC focuses on providing a larger number of samples per continental group, the SGDP covers many more populations. The 300 samples of the SGDP are distributed across 142 populations from seven continental regions: Africa, America, Central Asia & Siberia, East Asia, Oceania, South Asia and West Eurasia.

Similar to the analysis on the PDC, PopDel was run on the SGDP to assess what population structures can be revealed only using the count of detected deletion alleles.

4.7.1 Variant Calling Protocol

The reads of the 300 samples of the SGDP were downloaded and mapped to GRCh38 using the same approach as described for the PDC in Subsection 4.5.1. The SV calling of deletions of size 500-10,000 on the autosomes using PopDel (1.2.2) was performed with a maximum coverage of 120 and the '–representative-contigs' option. This option lets PopDel avoid loading redundant header information and is therefore useful for saving memory when analyzing larger cohorts.

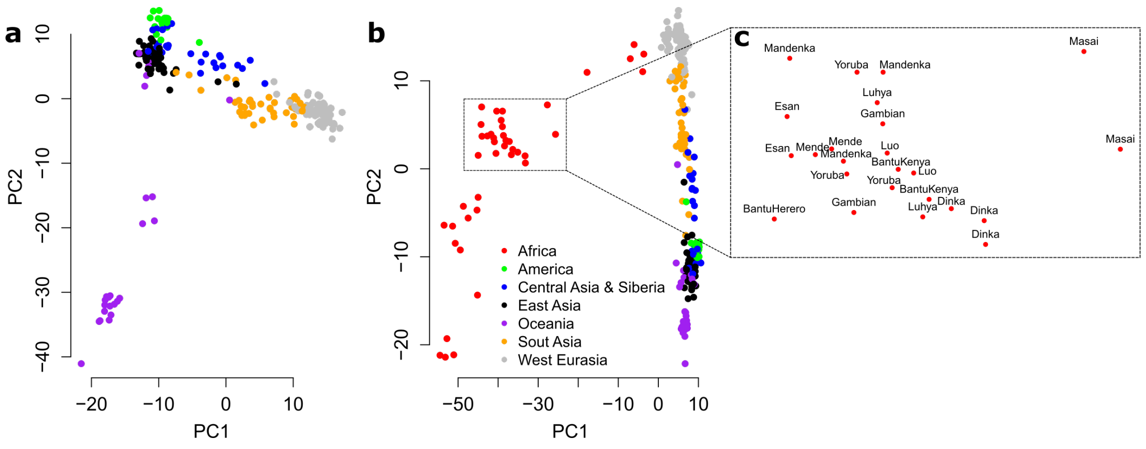

Figure 4.20: PCA based on deletion counts on the SGDP data. The color of the data points encode for the samples reported continental origin. **a** PCA without samples of African origin. **b** PCA of all SGDP samples. **c** Detailed view of a cluster of 24 samples of African origin reveals subclustering by population. The data points were labeled with their reported population.

4.7.2 Results

PCA. The PCA was calculated based on the count of deletion alleles per sample and SV as described in Subsection 4.7.2. The data points of the PCA plot in Figure 4.20b were colored by the reported continental region of origin of the samples. A good clustering by continental origin can be observed. The first principal component separates the African samples from the other continental regions. This is well in line with the observations on the PCA of the PDC in Figure 4.16. The second principal component separates the non-African samples from each other. The samples of Central Asian & Siberian, East Asian and Native American origin show no clear separation in this visualization due to a lack of dimensions. When excluding the African samples from the analysis (see Figure 4.20a), the separation becomes sharper.

The PCA on the deletion allele counts can reveal even finer population structures. When zooming further into the visualization (Figure 4.20c), one can see that the samples that originate from the same population tend to be located in close vicinity of each other in the PCA plot.

t-SNE. Figure 4.21a shows the t-SNE of the deletion allele counts for all continental groups. The continental clusters are visible more clearly than in the PCA plot. The detailed view on the cluster that contains almost all samples of African origin (Figure 4.21b) demonstrates that the subclustering by population has also been preserved. The five African samples that are not part of the African cluster are of Mozabite (2), Saharawi (2) and Somali (1) lineage. They are placed closer to the cluster of West Eurasian samples.

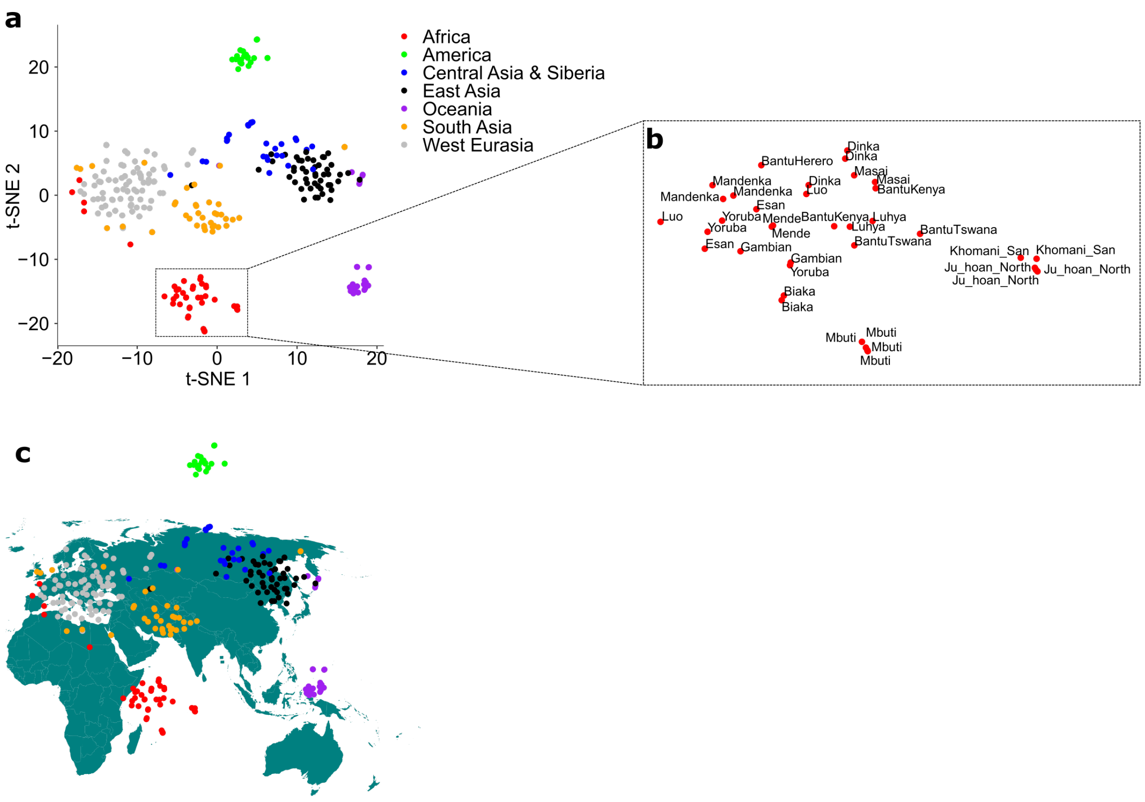

Figure 4.21: t-SNE based on deletion counts on the SGDP data. The color of the data points encode for the samples reported continental origin. **a** t-SNE of all SGDP samples. **b** Detailed view of the cluster containing most samples of African origin reveals subclustering by population. The data points were labeled with their reported population. **c** Overlay of the t-SNE and a partial world map shows how the clusters coincide with their geographic regions of origin. The t-SNE was placed over the world map without distortions.

These are the same five African samples that are placed closest to the West Eurasian Cluster in the PCA plot in Figure 4.20b. The Mozabite and Saharawari samples a placed closer together in the t-SNE, which matches their relative geographical vicinity of the two populations (western and northern Sahara), while the Somali sample, whose population resides in an area on the horn of Africa, is placed further away.

Figure 4.21c shows the t-SNE placed over a world map. This demonstrates how the continental clusters coincide (with some degree of tolerance) with their actual geographical origin. The placement of the American cluster close to Siberia is also very accurate, considering that Alaska and Siberia were once connected via the Bering land bridge that allowed multiple migration events between the continents[24;28;88].

The results for the SGDP show how the PCA or statistical methods like t-SNE on deletion allele counts can be used to get insight into even fine-grained (sub)population structures, despite the deletion allele counts representing only a very small piece of information on the variation in the samples. The geographic vicinity of populations and ancient migration routes that both play a key role in the genetic makeup of populations can also be revealed.

4.8 49,962 Icelandic Genomes

During development, an early version of PopDel (V0.1-alpha-c74a6c0) was tested on WGS data of 49,962 Icelanders. The data was provided by deCODE genetics in Iceland where the analysis was performed. The aim was to examine PopDel's performance on very big data sets and compare its results with those of previous studies.

Furthermore, the Icelandic genomes included 6,794 family trios that were analyzed with PopDel (1.0.6) to gain insights on transmission rate and MIER.

In a third application on the Icelandic data, PopDel was applied in a targeted fashion on the genomic vicinity of the LDLR gene. This lead to the detection of a previously unknown variant of medical interest in the LDLR gene. For details, refer to Section 1.8 discussing the publication that resulted from this finding.

4.8.1 Variant Calling Protocol

PopDel was applied on the Icelandic data set using mostly default parameters and the minimum deletion length being set to 500. The joint deletion calling was performed in multiple regions of the chromosomes in parallel using the '-r' option.

4.8.2 Results

Detection of Overlapping Deletions. As described in Subsection 3.3.3, PopDel applies sample-specific genotype weights (Equation 3.13). The weights guarantee that deletions that occur only in a minority of the samples in a cohort are not missed because most samples do not carry the respective deletion. Together with PopDel's multiple initializations of initial deletion lengths in the window-wise deletion detection (Algorithm 3.1) this allows PopDel to detect different deletions that are overlapping each other. This also holds if one of the deletions has a much lower allele frequency than the other. Figure 4.22 shows an example of such a case where a smaller deletion that was only present in 2 out of 49,962 samples was completely overlapped by a more common larger deletion. Both carriers of the the small deletion had a heterozygous genotype. This resulted in an allele frequency of only $2.02 \cdot 10^{-4}$ for the small deletion. Despite the low allele frequency and the overlapping larger deletion the small deletion was correctly detected and genotyped by PopDel.

Mendelian Inheritance Error Rate The MIER was calculated based on the PopDel deletion genotypes on the 6,794 family trios of the Icelandic data. A filter requiring a

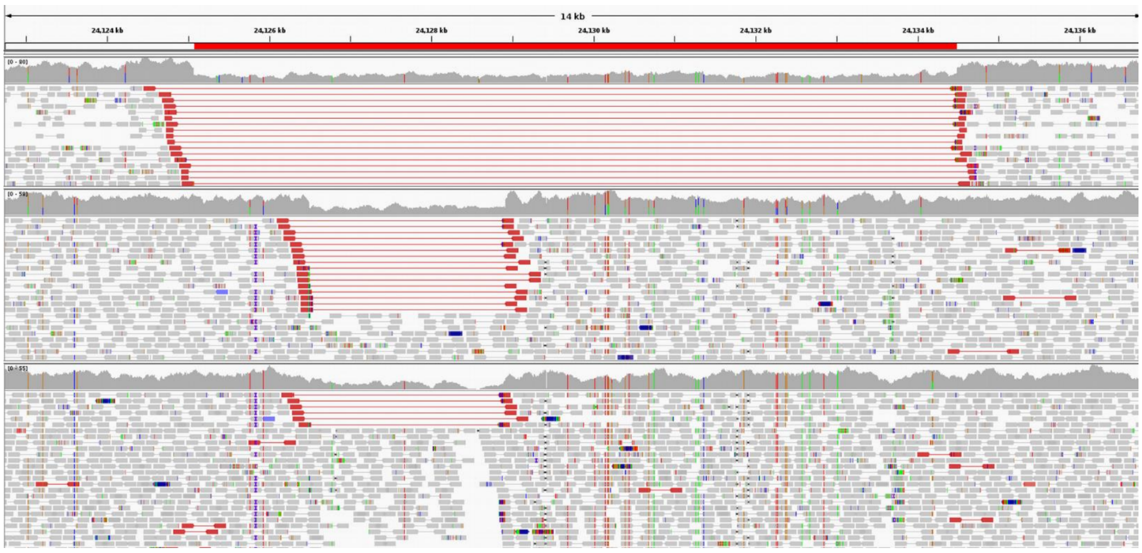

Figure 4.22: Overlapping deletions detected by PopDel in 49,962 Icelandic WGS samples (IGV screenshot). Read pairs supporting a deletion are colored red. The larger deletion (top) had a common allele frequency in the cohort, while the smaller deletion (bottom two samples) was only present in two samples.

GQ of at least 26 in all members of a trio for the analyzed deletions was applied. This resulted in an average of 1,963 deletions per trio that are consistent with the Mendelian laws of inheritance. The resulting error rate was 1.4 %, making 98.6 % of the genotype combinations compatible with the Mendelian rules.

Transmission Rate. The transmission rate on the 6,794 family trios was calculated on all detected deletions that were exclusive to a single trio. One parent was required to be a heterozygous carrier and the other parent had to be a non-carrier of the variant. The same GQ threshold of 26 as for the calculation of the MIER was applied. This resulted in a transmission rate of 49.2 % in the remaining 4,256 deletions. A two-sided binomial test showed no significant difference from the expected 50 % transmission rate (p-value = 0.32).

Chapter 5

Conclusion and Outlook

This final chapter recapitulates and summarizes the contributions of this thesis. Further, it discusses future applications and improvements to the approaches of this thesis.

5.1 Summary of Contributions

This thesis presents and implements the new joint calling approach PopDel and demonstrates how PopDel scales to tens of thousands of genomes. The thesis further introduces the PopDel read pair profile format for the efficient storage and retrieval of information from mapped paired-end NGS data. This thesis also presents a comprehensive multi-sample SV calling and genotyping benchmark that compares various state-of-the-art SV callers. Lastly, the thesis contributed to the detection of a novel deletion variant of medical interest in the LDLR gene, to a comprehensive benchmark study of many SV callers and to the evaluation of the new NRS insertions calling tool PopIns2.

The growing availability and amounts of NGS data poses big challenges and opportunities. One of the main challenges lies in efficiently processing the vast amounts of data. A big opportunity of these growing amounts of data is the expansion of our knowledge about structural variation and its consequences for the human health. This expanded knowledge will ultimately lead to new therapeutic methods for the treatment of severe disease. This thesis provides new approaches and tools for the detection and genotyping of SVs in population-scale data by presenting the PopDel approach. PopDel is implemented as a freely available and accessible program in the C++ programming language. PopDel efficiently scales to tens of thousands of WGS samples as shown by the analysis of almost 50,000 Icelandic genomes (see Section 4.8). PopDel's joint calling and genotyping of deletions compares favorably to other established SV callers in terms of running time, memory consumption and quality of the results. This quality holds on simulated and real data for both single samples or larger cohorts, as the benchmarks of Chapter 4 demonstrate. PopDel achieves the demonstrated quality while being fully unaware of the actual sequence of the NGS data and only relying on the positional information of the read pairs. The positional information of the read pairs of all samples in the cohort drives the joint calling and genotyping of SVs in PopDel's likelihood ratio model. This statistical model does not rely on rigid thresholds, but flexibly bases its predictions on the individual distributions of insert sizes in the analyzed genomes.

In addition to the joint calling and genotyping of deletions, this thesis presents prototypes for the joint detection and genotyping of duplications and inversions in PopDel. The prototypes are not yet fully developed, but already demonstrate excellent detection rates and good genotyping performance on simulated data. Further, the first steps for the extension of PopDel towards the processing of translocations have already been undertaken.

Many tools rely on the extraction of read pairs that are later used as evidence for the detection of SVs. This thesis advances this approach by the introduction of the PopDel read pair profiles. Instead of storing read pairs in a raw format like FASTQ, only the most relevant information for the SV-calling is extracted and stored in a binary format. The binary format drastically reduces the disk space requirements for storing the data for later use. The structure and the self-contained index of the read pair profiles allows for efficient searching of the read pair profiles. The profiles enable efficient streaming and targeted analyses of single genes as demonstrated by the detection of the new LDLR variant described in Section 1.8.

The benchmarks presented in Chapter 4 are another contribution of this thesis. Most benchmarks for SV callers focus on the performance in a single sample scenario. The benchmarks of this thesis put an emphasis on large cohorts for the evaluation of SV callers. The benchmarks show how the different tools scale to growing sample numbers and how the size of a cohort influences the quality of the results. Further, the benchmarks benefit from family-scale or population-scale metrics. The call sets that resulted from the deletion benchmarks are publicly available along with all scripts necessary for the reproduction of the results. The scripts, call sets and approaches of the benchmarks can be used in future efforts for the benchmarking of SV callers.

The analyses of the Polaris Diversity Cohort and the data of the Simons Genome Diversity Project (Sections 4.5 and 4.7) have demonstrated the suitability of the PopDel approach for the exploration of population structures. The analyses underline the power of PCA and UMAP in the context of population genomics by showing how both methods yield conclusive results despite being only applied on the count of deletion alleles per sample.

The work in the course of this thesis has further resulted in contributions to other publications in the field of human genetics and SV detection. Application of the PopDel program was directly responsible for the detection and publication of the previously unknown variant of medical interest in the LDLR gene that has been described by Bjornsson et al. (2021)[8]. The work on the benchmarks of Sarwal et al. (2020)[77] provides an additional resource for the assessment of different SVs in different data sets. The methods for the analysis of population-scale data that have been developed during the work on the benchmarks for PopDel have also been directly applied in the benchmarks of PopIns2 in Krannich et al. (2022)[50].

5.2 Future Potential and Applications

As demonstrated in the respective sections, PopDel's deletion calling and genotyping is in a reliable state and competitive with other state-of-the-art SV callers. The detection and genotyping of duplications, inversions and translocations was a later addition to the scope of this thesis as well as to PopDel and is still subject to improvements.

PopDel's model for the genotyping of duplications can be improved by adjusting the read pair distance deviation for the influence of the distance between reads and the duplication breakpoint. This adjustment has already been implemented for inversions (see Subsection 3.3.3.4) and can be transferred to duplications. Further, investigation of the deviation of the observed and the theoretically expected ratio of read pairs that support the variant allele or the reference allele will lead to a better understanding of the signatures of heterozygous and homozygous carriers of duplications. This improved understanding can be used for the implementation of a better, theoretically well-founded genotyping model for duplications in PopDel.

PopDel does currently not characterize all possible breakpoints of inversions. By specifically analyzing each breakpoint of an inversion individually, the breakpoints can be defined more accurately and called individually. This is necessary to account for the effect of micro-deletions at the breakpoints. A possible approach for this improvement lies in separately calling and genotyping the inversion breakpoints suggested by the presence of FF and RR read pairs instead of processing FF and RR read pairs together.

Especially the detection and genotyping of translocation will have to undergo fundamental improvements before it can be considered reliable. First, the high number of false positives must be addressed by revising the current thresholds for read pairs in the different chromosomal and positional clusters that are used for generating the translocation hypothesis. Additional internal tracking of the number of read pairs that map to different chromosomes as PopDel progresses along the genome are another potential solution for identifying regions in which translocations cannot be reliably called. Secondly, the four possible breakpoints of a translocation must be characterized individually instead of the currently implemented conjoint characterization. This individual characterization of breakpoints allows to better assess the exact breakpoints and is necessary to account for the effect of micro-deletions at the breakpoints.

Currently, PopDel does not calculate a joint variant likelihood for SVs other than deletions. By adapting the sample-specific genotype weights that are already being used for the calling of deletions, calculation of a joint variant likelihood becomes possible. This joint genotype likelihoods can then be used in a likelihood ratio test for assessing the statistical significance of a detected non-deletion variant.

While the initial tests on simulated data have been performed as a proof of concept, additional benchmarks on simulated and real data are required for a reliable assessment of PopDel's performance regarding SVs other than deletions. To address the problem of a comprehensive ground truth for duplications, inversions and possibly translocations, the well-studied haploid assemblies of CHM1[26] and CHM13[81] can be used to form an artificial diploid genome for the purpose of SV calling and genotyping benchmarks[35].

In the course of this thesis, the PopDel read pair profile format has already evolved from only storing the locations of the read pairs and their insert sizes to also containing the read pair orientations, the number of soft-clipped bases and read pairs mapping to different chromosomes. Further additions like the integration of coverage information or GC-content for the integration of read depth as an additional signature for the detection of CNVs are potential expansions of PopDel.

PopDel's capability of processing samples on population-scale and generating VCFs that contain the genotypes of tens of thousands of samples offers further opportunities. With enough samples, PopDel could integrate the calculation of metrics like the Hardy-Weinberg-Equilibrium or the linkage between different SVs for an additional annotation of its output. These information could be used to assess the quality of variants or to filter the variants to meet the needs of the specific use case.

Linked-reads are a valuable data source for the detection of SVs but have not been discussed in this thesis. Linked-reads are short-reads but provide long range information that can be used for mapping reads more reliably in repetitive regions of the genome and for the assessment of SVs. By applying PopDel on mapped linked-read data, it should be possible to get more accurate SV calls in those repetitive regions of the genome. An expansion of the read pair profile format to also hold information on the barcodes of the linked-read data that hold the long-range information could further prove valuable for the phasing of detected variants. The detection of large SVs can also benefit from this long-range information. For example, observing read pairs with the same barcode that are further apart than a sampled distribution of molecule sizes suggest, can be seen as a signal of a large deletion. This approach of detecting deletions is similar to the currently implemented deletion calling in PopDel, but would rely on the much larger molecule size on which the linked-reads are based on. The large molecule size makes the molecules more likely to contain deletions.

With the expected increase in population-scale studies and the ongoing decrease of sequencing costs, there will be a growing number of studies and data sets that can benefit from joint SV calling like implemented in PopDel. One of those data sets is already now a promising point of future research involving PopDel: the new release of the UK Biobank. Collaborators at deCODE genetics have applied PopDel successfully on 150,000 WGS samples of the new UK Biobank release and created a call set of many deletions. The coming analysis of this call set promises to yield some interesting new deletions as additions to catalogs of SVs and for the use in GWAS, thereby helping to deepen our understanding of structural variation and its role in human health.

Bibliography

[1] Abel, H. J., Larson, D. E., Regier, A. A., Chiang, C., Das, I., Kanchi, K. L., Layer, R. M., Neale, B. M., Salerno, W. J., Reeves, C., et al. (2020). Mapping and characterization of structural variation in 17,795 human genomes. *Nature*, 583(7814):83–89.

[2] Alberts, B., Johnson, A., Lewis, J., Morgan, D., Raff, M., Roberts, K., and Walter, P. (1995). *Molekularbiologie der Zelle*, volume 3. VCh Weinheim.

[3] Alkan, C., Coe, B. P., and Eichler, E. E. (2011). Genome structural variation discovery and genotyping. *Nature reviews genetics*, 12(5):363–376.

[4] Antonarakis, S. E., Rossiter, J., Young, M., Horst, J., De Moerloose, P., Sommer, S., Ketterling, R., Kazazian, H. J., Negrier, C., and Vinciguerra, C. (1995). Factor viii gene inversions in severe hemophilia a: results of an international consortium study.

[5] Ball, M. P., Thakuria, J. V., Zaranek, A. W., Clegg, T., Rosenbaum, A. M., Wu, X., Angrist, M., Bhak, J., Bobe, J., Callow, M. J., et al. (2012). A public resource facilitating clinical use of genomes. *Proceedings of the National Academy of Sciences*, 109(30):11920–11927.

[6] Bao, S., Jiang, R., Kwan, W., Wang, B., Ma, X., and Song, Y.-Q. (2011). Evaluation of next-generation sequencing software in mapping and assembly. *Journal of human genetics*, pages 1–10.

[7] Benjamini, Y. and Speed, T. P. (2012). Summarizing and correcting the gc content bias in high-throughput sequencing. *Nucleic acids research*, 40(10):e72–e72.

[8] Bjornsson, E., Gunnarsdottir, K., Halldorsson, G. H., Sigurdsson, A., Arnadottir, G. A., Jonsson, H., Olafsdottir, E. F., Niehus, S., Kehr, B., Sveinbjörnsson, G., et al. (2021). Lifelong reduction in ldl (low-density lipoprotein) cholesterol due to a gain-of-function mutation in ldlr. *Circulation: Genomic and Precision Medicine*, 14(1):e003029.

[9] Bycroft, C., Freeman, C., Petkova, D., Band, G., Elliott, L. T., Sharp, K., Motyer, A., Vukcevic, D., Delaneau, O., O Connell, J., et al. (2018). The uk biobank resource with deep phenotyping and genomic data. *Nature*, 562(7726):203–209.

[10] Cameron, D. L., Schröder, J., Penington, J. S., Do, H., Molania, R., Dobrovic, A., Speed, T. P., and Papenfuss, A. T. (2017). Gridss: sensitive and specific genomic rearrangement detection using positional de bruijn graph assembly. *Genome research*, 27(12):2050–2060.

[11] Carvalho, C. M. and Lupski, J. R. (2016). Mechanisms underlying structural variant formation in genomic disorders. *Nature Reviews Genetics*, 17(4):224–238.

[12] Chen, J., Li, X., Zhong, H., Meng, Y., and Du, H. (2019). Systematic comparison of germline variant calling pipelines cross multiple next-generation sequencers. *Scientific reports*, 9(1):1–13.

[13] Chen, X., Schulz-Trieglaff, O., Shaw, R., Barnes, B., Schlesinger, F., Källberg, M., Cox, A. J., Kruglyak, S., and Saunders, C. T. (2016). Manta: rapid detection of structural variants and indels for germline and cancer sequencing applications. *Bioinformatics*, 32(8):1220–1222.

[14] Chiang, C., Layer, R. M., Faust, G. G., Lindberg, M. R., Rose, D. B., Garrison, E. P., Marth, G. T., Quinlan, A. R., and Hall, I. M. (2015). Speedseq: ultra-fast personal genome analysis and interpretation. *Nature methods*, 12(10):966–968.

[15] Church, D. M., Schneider, V. A., Graves, T., Auger, K., Cunningham, F., Bouk, N., Chen, H.-C., Agarwala, R., McLaren, W. M., Ritchie, G. R., et al. (2011). Modernizing reference genome assemblies. *PLoS biology*, 9(7):e1001091.

[16] Collins, R. L., Brand, H., Karczewski, K. J., Zhao, X., Alföldi, J., Francioli, L. C., Khera, A. V., Lowther, C., Gauthier, L. D., Wang, H., et al. (2020). A structural variation reference for medical and population genetics. *Nature*, 581(7809):444–451.

[17] Consortium, . G. P. et al. (2015). A global reference for human genetic variation. *Nature*, 526(7571):68.

[18] Danecek, P., Auton, A., Abecasis, G., Albers, C. A., Banks, E., DePristo, M. A., Handsaker, R. E., Lunter, G., Marth, G. T., Sherry, S. T., et al. (2011). The variant call format and vcftools. *Bioinformatics*, 27(15):2156–2158.

[19] Danecek, P., Bonfield, J. K., Liddle, J., Marshall, J., Ohan, V., Pollard, M. O., Whitwham, A., Keane, T., McCarthy, S. A., Davies, R. M., et al. (2021). Twelve years of samtools and bcftools. *Gigascience*, 10(2):giab008.

[20] Drezen, E., Rizk, G., Chikhi, R., Deltel, C., Lemaitre, C., Peterlongo, P., and Lavenier, D. (2014). Gatb: genome assembly & analysis tool box. *Bioinformatics*, 30(20):2959–2961.

[21] Durbin, R., Eddy, S. R., Krogh, A., and Mitchison, G. (1998). *Biological sequence analysis: probabilistic models of proteins and nucleic acids*. Cambridge university press.

[22] Eggertsson, H. P., Jonsson, H., Kristmundsdottir, S., Hjartarson, E., Kehr, B., Masson, G., Zink, F., Hjorleifsson, K. E., Jonasdottir, A., Jonasdottir, A., et al. (2017). Graphtyper enables population-scale genotyping using pangenome graphs. *Nature genetics*, 49(11):1654–1660.

[23] Eizenga, J. M., Novak, A. M., Sibbesen, J. A., Heumos, S., Ghaffaari, A., Hickey, G., Chang, X., Seaman, J. D., Rounthwaite, R., Ebler, J., et al. (2020). Pangenome graphs. *Annual review of genomics and human genetics*, 21:139–162.

[24] Elias, S. A., Short, S. K., Nelson, C. H., and Birks, H. H. (1996). Life and times of the bering land bridge. *Nature*, 382(6586):60–63.

[25] Escaramís, G., Docampo, E., and Rabionet, R. (2015). A decade of structural variants: description, history and methods to detect structural variation. *Briefings in functional genomics*, 14(5):305–314.

[26] Fan, J.-B., Surti, U., Taillon-Miller, P., Hsie, L., Kennedy, G. C., Hoffner, L., Ryder, T., Mutch, D. G., and Kwok, P.-Y. (2002). Paternal origins of complete hydatidiform moles proven by whole genome single-nucleotide polymorphism haplotyping. *Genomics*, 79(1):58–62.

[27] Ferguson, D. O., Sekiguchi, J. M., Chang, S., Frank, K. M., Gao, Y., DePinho, R. A., and Alt, F. W. (2000). The nonhomologous end-joining pathway of dna repair is required for genomic stability and the suppression of translocations. *Proceedings of the National Academy of Sciences*, 97(12):6630–6633.

[28] Flegontov, P., Altınışık, N. E., Changmai, P., Rohland, N., Mallick, S., Adamski, N., Bolnick, D. A., Broomandkhoshbacht, N., Candilio, F., Culleton, B. J., et al. (2019). Palaeo-eskimo genetic ancestry and the peopling of chukotka and north america. *Nature*, 570(7760):236–240.

[29] Francioli, L. C., Menelaou, A., Pulit, S. L., Van Dijk, F., Palamara, P. F., Elbers, C. C., Neerincx, P. B., Ye, K., Guryev, V., Kloosterman, W. P., et al. (2014). Whole-genome sequence variation, population structure and demographic history of the dutch population. *Nature genetics*, 46(8):818–825.

[30] Guan, P. and Sung, W.-K. (2016). Structural variation detection using next-generation sequencing data: a comparative technical review. *Methods*, 102:36–49.

[31] Gudbjartsson, D. F., Helgason, H., Gudjonsson, S. A., Zink, F., Oddson, A., Gylfason, A., Besenbacher, S., Magnusson, G., Halldorsson, B. V., Hjartarson, E., et al. (2015). Large-scale whole-genome sequencing of the icelandic population. *Nature genetics*, 47(5):435–444.

[32] Handsaker, R. E., Korn, J. M., Nemesh, J., and McCarroll, S. A. (2011). Discovery and genotyping of genome structural polymorphism by sequencing on a population scale. *Nature genetics*, 43(3):269–276.

[33] Haraldsdottir, S., Rafnar, T., Frankel, W. L., Einarsdottir, S., Sigurdsson, A., Hampel, H., Snaebjornsson, P., Masson, G., Weng, D., Arngrimsson, R., et al. (2017). Comprehensive population-wide analysis of lynch syndrome in iceland reveals founder mutations in msh6 and pms2. *Nature communications*, 8(1):1–11.

[34] Hardy, G. H. (1908). Mendelian proportions in a mixed population. *Science*, 28(706):49–50.

[35] Hastie, A. R., Lam, E. T., Pang, A. W. C., Zhang, X., Andrews, W., Lee, J., Liang, T. Y., Wang, J., Zhou, X., Zhu, Z., et al. (2017). Rapid automated large structural variation detection in a diploid genome by nanochannel based next-generation mapping. *BioRxiv*, page 102764.

[36] Hinton, G. E. and Roweis, S. (2002). Stochastic neighbor embedding. *Advances in neural information processing systems*, 15.

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

[38] Huang, W., Li, L., Myers, J. R., and Marth, G. T. (2012). Art: a next-generation sequencing read simulator. *Bioinformatics*, 28(4):593–594.

[39] Jun, G., Sedlazeck, F., Zhu, Q., English, A., Metcalf, G., Kang, H. M., (HGSVC), H. G. S. V. C., Lee, C., Gibbs, R., and Boerwinkle, E. (2021). mucnv: genotyping structural variants for population-level sequencing. *Bioinformatics*, 37(14):2055–2057.

[40] Kavak, P., Lin, Y.-Y., Numanagić, I., Asghari, H., Güngör, T., Alkan, C., and Hach, F. (2017). Discovery and genotyping of novel sequence insertions in many sequenced individuals. *Bioinformatics*, 33(14):i161–i169.

[41] Kehr, B., Helgadottir, A., Melsted, P., Jonsson, H., Helgason, H., Jonasdottir, A., Jonasdottir, A., Sigurdsson, A., Gylfason, A., Halldorsson, G. H., et al. (2017). Diversity in non-repetitive human sequences not found in the reference genome. *Nature Genetics*, 49(4):588–593.

[42] Kehr, B., Melsted, P., and Halldórsson, B. V. (2016). Popins: population-scale detection of novel sequence insertions. *Bioinformatics*, 32(7):961–967.

[43] Khurana, T. S., Prendergast, R. A., Alameddine, H. S., Tome, F., Fardeau, M., Arahata, K., Sugita, H., and Kunkel, L. M. (1995). Absence of extraocular muscle pathology in duchenne's muscular dystrophy: role for calcium homeostasis in extraocular muscle sparing. *The Journal of experimental medicine*, 182(2):467–475.

[44] Kim, K., Seong, M.-W., Chung, W.-H., Park, S. S., Leem, S., Park, W., Kim, J., Lee, K., Park, R. W., and Kim, N. (2015). Effect of next-generation exome sequencing depth for discovery of diagnostic variants. *Genomics Inform*, 13(2):31–9.

[45] Kim, S., Scheffler, K., Halpern, A. L., Bekritsky, M. A., Noh, E., Källberg, M., Chen, X., Kim, Y., Beyter, D., Krusche, P., et al. (2018). Strelka2: fast and accurate calling of germline and somatic variants. *Nature methods*, 15(8):591–594.

[46] Kloosterman, W. P., Francioli, L. C., Hormozdiari, F., Marschall, T., Hehir-Kwa, J. Y., Abdellaoui, A., Lameijer, E.-W., Moed, M. H., Koval, V., Renkens, I., et al. (2015). Characteristics of de novo structural changes in the human genome. *Genome research*, 25(6):792–801.

[47] Koboldt, D. C. (2020). Best practices for variant calling in clinical sequencing. *Genome Medicine*, 12(1):1–13.

[48] Koboldt, D. C., Zhang, Q., Larson, D. E., Shen, D., McLellan, M. D., Lin, L., Miller, C. A., Mardis, E. R., Ding, L., and Wilson, R. K. (2012). Varscan 2: somatic mutation and copy number alteration discovery in cancer by exome sequencing. *Genome research*, 22(3):568–576.

[49] Köster, J. and Rahmann, S. (2012). Snakemake - a scalable bioinformatics workflow engine. *Bioinformatics*, 28(19):2520–2522.

[50] Krannich, T., White, W. T. J., Niehus, S., Holley, G., Halldórsson, B. V., and Kehr, B. (2022). Population-scale detection of non-reference sequence variants using colored de bruijn graphs. *Bioinformatics*, 38(3):604–611.

[51] Langmead, B. and Salzberg, S. L. (2012). Fast gapped-read alignment with bowtie 2. *Nature methods*, 9(4):357–359.

[52] Langmead, B., Trapnell, C., Pop, M., and Salzberg, S. L. (2009). Ultrafast and memory-efficient alignment of short dna sequences to the human genome. *Genome biology*, 10(3):1–10.

[53] Larson, D. E., Abel, H. J., Chiang, C., Badve, A., Das, I., Eldred, J. M., Layer, R. M., and Hall, I. M. (2019). svtools: population-scale analysis of structural variation. *Bioinformatics*, 35(22):4782–4787.

[54] Layer, R. M., Chiang, C., Quinlan, A. R., and Hall, I. M. (2014). Lumpy: a probabilistic framework for structural variant discovery. *Genome biology*, 15(6):1–19.

[55] Li, H. (2011). A statistical framework for snp calling, mutation discovery, association mapping and population genetical parameter estimation from sequencing data. *Bioinformatics*, 27(21):2987–2993.

[56] Li, H. (2013). Aligning sequence reads, clone sequences and assembly contigs with bwa-mem. *arXiv preprint arXiv:1303.3997*.

[57] Li, H. and Durbin, R. (2009). Fast and accurate short read alignment with burrows-wheeler transform. *Bioinformatics*, 25(14):1754–1760.

[58] Li, H., Handsaker, B., Wysoker, A., Fennell, T., Ruan, J., Homer, N., Marth, G., Abecasis, G., and Durbin, R. (2009). The sequence alignment/map format and samtools. *Bioinformatics*, 25(16):2078–2079.

[59] Li, H., Ruan, J., and Durbin, R. (2008). Mapping short dna sequencing reads and calling variants using mapping quality scores. *Genome research*, 18(11):1851–1858.

[60] Logsdon, G. A., Vollger, M. R., and Eichler, E. E. (2020). Long-read human genome sequencing and its applications. *Nature Reviews Genetics*, 21(10):597–614.

[61] Mak, A. C., White, M. J., Eckalbar, W. L., Szpiech, Z. A., Oh, S. S., Pino-Yanes, M., Hu, D., Goddard, P., Huntsman, S., Galanter, J., et al. (2018). Whole-genome sequencing of pharmacogenetic drug response in racially diverse children with asthma. *American journal of respiratory and critical care medicine*, 197(12):1552–1564.

[62] Mallick, S., Li, H., Lipson, M., Mathieson, I., Gymrek, M., Racimo, F., Zhao, M., Chennagiri, N., Nordenfelt, S., Tandon, A., et al. (2016). The simons genome diversity project: 300 genomes from 142 diverse populations. *Nature*, 538(7624):201–206.

[63] McKenna, A., Hanna, M., Banks, E., Sivachenko, A., Cibulskis, K., Kernytsky, A., Garimella, K., Altshuler, D., Gabriel, S., Daly, M., et al. (2010). The genome analysis toolkit: a mapreduce framework for analyzing next-generation dna sequencing data. *Genome research*, 20(9):1297–1303.

[64] Mohiyuddin, M., Mu, J. C., Li, J., Bani Asadi, N., Gerstein, M. B., Abyzov, A., Wong, W. H., and Lam, H. Y. (2015). Metasv: an accurate and integrative structural-variant caller for next generation sequencing. *Bioinformatics*, 31(16):2741–2744.

[65] Narayan, A., Berger, B., and Cho, H. (2021). Assessing single-cell transcriptomic variability through density-preserving data visualization. *Nature biotechnology*, 39(6):765–774.

[66] Needleman, S. B. and Wunsch, C. D. (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, 48(3):443–453.

[67] Niehus, S., Jónsson, H., Schönberger, J., Björnsson, E., Beyter, D., Eggertsson, H. P., Sulem, P., Stefánsson, K., Halldórsson, B. V., and Kehr, B. (2021). Popdel identifies medium-size deletions simultaneously in tens of thousands of genomes. *Nature communications*, 12(1):1–10.

[68] Pal, T., Permuth-Wey, J., and Sellers, T. A. (2008). A review of the clinical relevance of mismatch-repair deficiency in ovarian cancer. *Cancer: Interdisciplinary International Journal of the American Cancer Society*, 113(4):733–742.

[69] Pedersen, B. S., Layer, R., and Quinlan, A. R. (2020). smoove: structural-variant calling and genotyping with existing tools.

[70] Poplin, R., Ruano-Rubio, V., DePristo, M. A., Fennell, T. J., Carneiro, M. O., Van der Auwera, G. A., Kling, D. E., Gauthier, L. D., Levy-Moonshine, A., Roazen, D., et al. (2017). Scaling accurate genetic variant discovery to tens of thousands of samples. *BioRxiv*, page 201178.

[71] Quinlan, A. R. and Hall, I. M. (2010). Bedtools: a flexible suite of utilities for comparing genomic features. *Bioinformatics*, 26(6):841–842.

[72] Ramachandran, S., Deshpande, O., Roseman, C. C., Rosenberg, N. A., Feldman, M. W., and Cavalli-Sforza, L. L. (2005). Support from the relationship of genetic and geographic distance in human populations for a serial founder effect originating in africa. *Proceedings of the National Academy of Sciences*, 102(44):15942–15947.

[73] Ramocki, M. B., Peters, S. U., Tavyev, Y. J., Zhang, F., Carvalho, C. M., Schaaf, C. P., Richman, R., Fang, P., Glaze, D. G., Lupski, J. R., et al. (2009). Autism and other neuropsychiatric symptoms are prevalent in individuals with mecp2 duplication syndrome. *Annals of Neurology: Official Journal of the American Neurological Association and the Child Neurology Society*, 66(6):771–782.

[74] Rausch, T., Zichner, T., Schlattl, A., Stütz, A. M., Benes, V., and Korbel, J. O. (2012). Delly: structural variant discovery by integrated paired-end and split-read analysis. *Bioinformatics*, 28(18):i333–i339.

[75] Robinson, J. T., Thorvaldsdóttir, H., Winckler, W., Guttman, M., Lander, E. S., Getz, G., and Mesirov, J. P. (2011). Integrative genomics viewer. *Nature biotechnology*, 29(1):24–26.

[76] Sanger, F., Nicklen, S., and Coulson, A. R. (1977). Dna sequencing with chain-terminating inhibitors. *Proceedings of the national academy of sciences*, 74(12):5463–5467.

[77] Sarwal, V., Niehus, S., Ayyala, R., Chang, S., Lu, A., Darci-Maher, N., Littman, R., Chhugani, K., Soylev, A., Comarova, Z., et al. (2020). A comprehensive benchmarking of wgs-based structural variant callers. *bioRxiv*.

[78] Schneider, V. A., Graves-Lindsay, T., Howe, K., Bouk, N., Chen, H.-C., Kitts, P. A., Murphy, T. D., Pruitt, K. D., Thibaud-Nissen, F., Albracht, D., et al. (2017). Evaluation of grch38 and de novo haploid genome assemblies demonstrates the enduring quality of the reference assembly. *Genome research*, 27(5):849–864.

[79] Sibbesen, J. A., Maretty, L., and Krogh, A. (2018). Accurate genotyping across variant classes and lengths using variant graphs. *Nature genetics*, 50(7):1054–1059.

[80] Smith, T. F., Waterman, M. S., et al. (1981). Identification of common molecular subsequences. *Journal of molecular biology*, 147(1):195–197.

[81] Steinberg, K. M., Schneider, V. A., Graves-Lindsay, T. A., Fulton, R. S., Agarwala, R., Huddleston, J., Shiryev, S. A., Morgulis, A., Surti, U., Warren, W. C., et al. (2014). Single haplotype assembly of the human genome from a hydatidiform mole. *Genome research*, 24(12):2066–2076.

[82] Sudmant, P. H., Rausch, T., Gardner, E. J., Handsaker, R. E., Abyzov, A., Huddleston, J., Zhang, Y., Ye, K., Jun, G., Fritz, M. H.-Y., et al. (2015). An integrated map of structural variation in 2,504 human genomes. *Nature*, 526(7571):75–81.

[83] Sullivan, K. E., McDonald-McGinn, D. M., Driscoll, D. A., Zmijewski, C. M., Ellabban, A. S., Reed, L., Emanuel, B. S., Zackai, E. H., Athreya, B. H., and Keenan, G. (1997). Juvenile rheumatoid arthritis-like polyarthritis in chromosome 22q11. 2 deletion syndrome (digeorge anomalad/velocardiofacial syndrome/conotruncal anomaly face syndrome). *Arthritis & Rheumatism: Official Journal of the American College of Rheumatology*, 40(3):430–436.

[84] UK Biobank (2021). Whole genome sequencing data on 200,000 uk biobank participants made available for research. Online press release. Accessed: 2022-21-03.

[85] Van der Maaten, L. and Hinton, G. (2008). Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

[86] Watson, J. D. and Crick, F. H. (1953). Molecular structure of nucleic acids: a structure for deoxyribose nucleic acid. *Nature*, 171(4356):737–738.

[87] Wilks, S. S. (1938). The large-sample distribution of the likelihood ratio for testing composite hypotheses. *The annals of mathematical statistics*, 9(1):60–62.

[88] Williams, R. C., Steinberg, A. G., Gershowitz, H., Bennett, P. H., Knowler, W. C., Pettitt, D. J., Butler, W., Baird, R., Dowda-Rea, L., Burch, T. A., et al. (1985). Gm allotypes in native americans: evidence for three distinct migrations across the bering land bridge. *American Journal of Physical Anthropology*, 66(1):1–19.

[89] Yuen, R. K., Merico, D., Bookman, M., Howe, J. L., Thiruvahindrapuram, B., Patel, R. V., Whitney, J., Deflaux, N., Bingham, J., Wang, Z., et al. (2017). Whole genome sequencing resource identifies 18 new candidate genes for autism spectrum disorder. *Nature neuroscience*, 20(4):602–611.

[90] Zarate, S., Carroll, A., Mahmoud, M., Krasheninina, O., Jun, G., Salerno, W. J., Schatz, M. C., Boerwinkle, E., Gibbs, R. A., and Sedlazeck, F. J. (2020). Parliament2: Accurate structural variant calling at scale. *GigaScience*, 9(12):giaa145.

[91] Zhang, F., Khajavi, M., Connolly, A. M., Towne, C. F., Batish, S. D., and Lupski, J. R. (2009). The dna replication fostes/mmbir mechanism can generate genomic, genic and exonic complex rearrangements in humans. *Nature genetics*, 41(7):849–853.

[92] Zook, J. M., Hansen, N. F., Olson, N. D., Chapman, L., Mullikin, J. C., Xiao, C., Sherry, S., Koren, S., Phillippy, A. M., Boutros, P. C., et al. (2020). A robust benchmark for detection of germline large deletions and insertions. *Nature biotechnology*, 38(11):1347–1355.

Nomenclature

contig	Contiguous sequence
GWAS	Genome-wide association study
bp	base pairs
CDBG	Colored De Bruijn Graph.
CNV	Copy Number Variant.
CNV	Copy Number Variant
ddNTP	Didesoxynukleosidtriphosphat
DNA	Deoxyribonucleic Acid
FF	Forward-Forward orientation (of a read pair)
FR	Forward-Reverse orientation (of a read pair)
G1k	1000 Genomes Project
GATK HC	GATK Haplotype Caller
GATK RCM	GATK Reference Confidence Model
GoNL	Genome of the Netherlands
GQ	Genotype Quality.
GRCh37	Genome Reference Consortium Human Build 37
HPC	High-Performance Computing

HWE	Hardy-Weinberg-Equilibrium
I/O	Input/Output
IGV	Integrative Genomics Viewer
indels	Insertion or Deletions variants (typically < 50 bp).
LDL	Low-Density Lipoprotein
LDLR	Low-Density Lipoprotein Receptor
LRS	Long-Read Sequencing
Megabases	Millions of bases
MIER	Mendelian Inheritance Error Rate.
MMBIR	Microhomology-mediated break-induced replication
mRNA	Messenger RNA
NAHR	Non-Allelic Homologous Recombination
NGS	Next Generation Sequencing.
NHEJ	Non-Homologous End Joining
NRS	Non Reference Sequence. A piece of DNA sequence that is not observed in the reference genome. When the sample genome is aligned to the reference NRS variants typically manifest as insertions even though the sequence might in fact be ancestral and simply missing/deleted from the reference.
PCA	Principal Component Analysis
PCR	Poylmerase Chain Reaction
PDC	Polaris HiSeqX Diversity Cohort
PKC	Polaris HiSeqX Kids Cohort
RF	Reverse-Forward orientation (of a read pair)

RNA	Ribonucleic Acid
RR	Reverse-Reverse orientation (of a read pair)
BAM	Binary Alignment/Map
SAM	Sequence Alignment/Map
SGDP	Simons Genome Diversity Project
SGDP	Simons Genome Diversity Project
SNVs	Single Nucleotide Variants
SVs	Structural Variants (typically affecting $>$ 50 bp)
VCF	Variant Call Format
WGS	Whole Genome Sequencing

Appendix A

Appendix

A.1 Overview of PopDel Parameters

A.1.1 PopDel Profile

PopDel [OPTIONS] BAM/CRAM-FILE

PopDel [OPTIONS] BAM/CRAM-FILE CHROM:BEGIN-END [CHROM:BEGIN-END ...]

Iterates over (user defined regions of) a BAM or CRAM file in tiling windows of 256 bp. For each window, PopDel promotes all read pairs whose ends pass the quality checks. PopDel saves their insert size deviation form the mean together with their position in '*BAM/CRAM-FILE*.profile', together with the insert sizes distribution of each read group. Only insert sizes up to a maximum length (option --max-deletion-size) are considered.

-r, --reference REF Reference genome version used for the mapping. Not used when using custom sampling intervals (option '-i'). One of 'GRCh37', 'GRCh38', 'hg19', 'hg38' (case-insensitive). Default: GRCh38.

-b, --blacklist FILE BED file of regions which will be ignored during translocation detection.

-d, --max-deletion-size NUM Maximum size of deletions. Default: 10000.

-i, --intervals FILE File with genomic intervals for parameter estimation instead of default intervals (see README). One closed interval per line, formatted as 'CHROM:START-END', 1-based coordinates.

-mrg, --merge-read-groups Merge all read groups of the sample. Only advised if they share the same properties!

-n, --min-read-num NUM Minimum number of read pairs for parameter estimation (per read group) Default: 50000.

-o, --out FILE Output file name. Default: *BAM/CRAM-FILE*.profile.

-f, --flags-set NUM Only use reads with all bits of NUM set in the bam flag. Default: 1.

-F, --flags-unset NUM Only use reads with all bits of NUM unset in the bam flag. Default: 3852.

-ir, --index-region-size NUM Size of the index region intervals. Default: 10000.

-mq, --min-mapping-qual NUM Only use reads with a mapping quality above NUM. Default: 1.

-R, --reference-file FILE FASTA file of the reference genome. Only required for CRAM files whose header entries point to the wrong file.

-s, --min-align-score NUM Only use reads with an alignment score relative to read length above NUM. Default: 0.8.

-u, --min-unclipped NUM Only use reads of which at least NUM bases are not clipped. Default: 50.

-sm, --sample-name STRING Set sample name.Default: @RG SM tag in BAM-file. BAM-file name if tag is not present.

A.1.2 PopDel View

PopDel PROFILE-FILE

Displays a profile file in human readable format.

-c, --clipping Print the number of clipped bases of the read pairs.

-e, --header Write the header.

-E, --onlyHeader Only write the header.

-i, --histograms Write insert size histograms.

-o, --orientation Print orientation of the read pairs.

-r, --region CHR:BEGIN-END Limit view to this genomic region.

-t, --translocations Print the block of translocated records.

-tt, --onlyTranslocations Only print the block of translocated records.

A.1.3 PopDel Call

PopDel [OPTIONS] ROFILE-LIST-FILE PopDel [OPTIONS] PROFILE-FILE1 PROFILE-FILE2 [PROFILE-FILE3 ...]

Performs joint-calling of deletions using a list of profile-files previously created using the 'popdel profile' command. The input profiles are either specified directly as arguments or listed in PROFILE-LIST-FILE (one filename per line).

-A, --active-coverage-file FILE File with lines consisting of "ReadGroup maxCov". If this value is reached no more new reads are loaded for this read group until the coverage drops again. Further, the sample will be excluded from calling in high-coverage windows. A value of 0 disables the filter for the read group.

-a, --active-coverage NUM Maximum number of active read pairs (~coverage). This value is taken for all read groups that are not listed in 'active-coverage-file'. Setting it to 0 disables the filter for all read groups that are not specified in 'active-coverage-file'. In range [0..inf]. Default: 100.

-d, --max-deletion-size NUM Maximum size of deletions. Default: 10000.

l, --min-init-length NUM Minimal deletion length at initialization of iteration. Default: 4 * standard deviation. -m, --min-length NUM Minimal deletion length during iteration. Default: 95th percentile of min-init-lengths.

-o, --out FILE Output file name. Default: popdel.vcf.

-r, --region-of-interest REGION Genomic region 'chr:start-end' (closed interval, 1-based index). Calling is limited to this region. Multiple regions can be defined by using the parameter -r multiple times.

-R, --ROI-file FILE File listing one or more regions of interest, one region per line. See parameter -r.

s, --min-sample-fraction NUM Minimum fraction of samples which is required to have enough data in the window. In range [0.0..1.0]. Default: 0.1.

-x, --del-only Exclusively call deletions. Ignore other SVs.

-b, --buffer-size NUM Number of buffered windows. In range [10000..inf]. Default: 200000.

-c, --min-relative-window-cover NUM Determines which fraction of a deletion has to be covered by significant windows. In range [0.0..2.0]. Default: 0.5.

-e, --representative-contigs Use the contig names and lengths of the first sample for all samples. Reduces memory consumption, but requires all samples to have the same contig names and lenghts.

-f, --pseudocount-fraction NUM The biggest likelihood of the background distribution will be divided by this value to determine the pseudocounts of the histogram. Bigger values boost the sensitivity for HET calls but also increase the chance of missclassifying HOMDEL or HOMREF as HET calls. In range [50..inf]. Default: 500.

-F, --output-failed

-n, --no-regenotyping Outputs every potential variant window without re-genotyping and merging.

-p, --prior-probability NUM Prior probability of a deletion. In range [0.0..0.9999]. Default: 0.0001.

-t, --iterations NUM Number of iterations in EM for length estimation. Default: 15.

-u, --unsmoothed Disable the smoothing of the insert size histogram.

A.2 Default Sampling Regions for the PopDel Profiling

chr1:35000000-36000000

chr2:174000000-175000000

chr3:36500000-37500000

chr4:88000000-89000000

chr5:38000000-39000000

chr6:38000000-39000000

chr7:38000000-39000000

chr8:19000000-20000000

chr9:19000000-20000000

chr10:19000000-20000000

chr11:19000000-20000000

chr12:19000000-20000000

chr13:25000000-26000000

chr14:25000000-26000000

chr15:25000000-26000000

chr16:25000000-26000000

chr17:31000000-32000000

chr18:31000000-32000000

chr19:31000000-32000000

chr20:33000000-34000000

chr21:21000000-22000000

chr22:25000000-26000000

A.3 Parameters for NGS Data Simulation with art_illumina

-ss HS25 -p -l 150 -f 15 -m 300 -s 50 -rs ${RANDOMSEED}

A.4 Tables of Simulation Benchmarks without Genotype Consideration

Table A.1: TP on uniform deletion simulation samples.

Samples	PopDel	Delly	Manta	Smoove	GRIDSS
1	1157	1166	1145	1140	1148
2	1414	1408	1395	1380	1396
3	1530	1517	1509	1491	1515
4	1587	1583	1572	1553	1579
5	1633	1630	1616	1596	1625
6	1652	1645	1633	1611	1643
7	1672	1664	1652	1629	1663
8	1690	1678	1666	1643	1679
9	1696	1683	1671	1648	1685
10	1702	1687	1676	1653	1690
20	1748	1732	1723	1702	1736
30	1766	1748	1739	1718	1749
40	1775	1757	1746	1728	1756
50	1781	1763	1754	1734	1764
60	1783	1765	1755	1737	1767
70	1787	1767	1758	1739	1771
80	1789	1769	1759	1740	1771
90	1790	1770	1761	1741	1770
100	1790	1771	1762	1742	1772
200	1794	1777	1761	1748	1773
300	1794	1782	1761	1750	1777
400	1795	1785	1764	1751	1779
500	1795	1787	1763	1751	NA
600	1793	1790	1764	1753	NA
700	1795	1792	1765	1755	NA
800	1795	1793	1765	1755	NA
900	1795	1794	1762	1755	NA
1000	1794	1793	1762	1756	NA

Table A.2: FP on uniform deletion simulation samples.

Samples	PopDel	Delly	Manta	Smoove	GRIDSS
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	0
4	0	0	0	0	0
5	0	0	0	0	0
6	0	0	0	0	0
7	0	0	0	0	0
8	0	0	1	0	0
9	0	0	1	0	0
10	0	0	1	0	0
20	0	0	0	0	0
30	0	0	0	0	0
40	1	0	1	0	0
50	1	0	1	0	5
60	0	0	1	0	9
70	0	0	1	0	17
80	0	0	1	0	29
90	0	1	1	0	58
100	1	1	1	0	69
200	0	2	4	0	82
300	1	3	4	0	134
400	0	5	4	0	309
500	1	5	3	0	NA
600	2	6	4	0	NA
700	1	8	4	0	NA
800	1	11	4	0	NA
900	1	13	4	0	NA
1000	1	13	4	0	NA

Table A.3: FN on uniform deletion simulation samples.

Samples	PopDel	Delly	Manta	Smoove	GRIDSS
1	159	150	171	176	168
2	185	191	204	219	203
3	189	202	210	228	204
4	204	208	219	238	212
5	205	208	222	242	213
6	203	210	222	244	212
7	202	210	222	245	211
8	201	213	225	248	212
9	200	213	225	248	211
10	198	213	224	247	210
20	203	219	228	249	215
30	197	215	224	245	214
40	197	215	226	244	216
50	198	216	225	245	215
60	199	217	227	245	215
70	197	217	226	245	213
80	198	218	228	247	216
90	198	218	227	247	218
100	199	218	227	247	217
200	199	216	232	245	220
300	202	214	235	246	219
400	203	213	234	247	219
500	203	211	235	247	NA
600	205	208	234	245	NA
700	204	207	234	244	NA
800	204	206	234	244	NA
900	204	205	237	244	NA
1000	206	207	238	244	NA

Table A.4: TP on G1k deletion simulation samples

Samples	PopDel	Delly	Manta	Smoove
1	176	163	178	192
2	290	262	287	303
3	379	343	369	388
4	417	378	410	427
5	456	408	456	470
6	491	441	492	505
7	520	466	528	537
8	541	483	551	559
9	558	502	569	581
10	582	524	592	604
20	773	717	790	803
30	913	855	938	952
40	999	948	1025	1042
50	1085	1040	1108	1127
60	1159	1126	1193	1215
70	1272	1248	1304	1331
80	1339	1313	1371	1400
90	1401	1371	1435	1464
100	1466	1443	1502	1534
200	2048	2029	2088	2138
300	2452	2431	2442	2559
400	2721	2699	2639	2846
500	2962	2944	2856	3111

Table A.5: FP on G1k deletion simulation samples

Samples	PopDel	Delly	Manta	Smoove
1	0	0	0	0
2	0	1	0	0
3	0	1	1	0
4	0	1	1	0
5	0	1	1	0
6	0	1	1	0
7	0	1	1	1
8	0	1	1	1
9	1	1	1	1
10	2	1	1	1
20	2	1	7	0
30	5	1	8	0
40	4	2	9	0
50	4	1	8	0
60	5	3	12	0
70	9	3	13	0
80	8	3	15	0
90	8	4	16	0
100	8	4	16	0
200	13	10	18	0
300	16	13	20	0
400	13	17	24	0
500	15	26	28	0

Table A.6: FN on G1k deletion simulation samples

Samples	PopDel	Delly	Manta	Smoove
1	35	48	33	19
2	42	70	45	29
3	46	82	56	37
4	48	87	55	38
5	47	95	47	33
6	47	97	46	33
7	51	105	43	34
8	54	112	44	36
9	59	115	48	36
10	60	118	50	38
20	86	142	69	56
30	100	158	75	61
40	110	161	84	67
50	112	157	89	70
60	129	162	95	73
70	138	162	106	79
80	140	166	108	79
90	144	174	110	81
100	150	173	114	82
200	202	221	162	112
300	239	260	249	132
400	255	277	337	130
500	284	302	390	135

Table A.7: TP on duplication simulation samples.

Samples	PopDel	Delly	Manta	Smoove
1	1182	1174	1139	1073
2	1441	1435	1387	1329
3	1554	1547	1506	1444
4	1616	1608	1570	1509
5	1662	1652	1614	1555
6	1677	1671	1628	1569
7	1695	1690	1651	1588
8	1713	1706	1666	1604
9	1719	1712	1672	1617
10	1723	1717	1676	1623

Table A.8: FP on duplication simulation samples.

Samples	PopDel	Delly	Manta	Lumpy
1	0	0	0	0
2	0	0	0	0
3	0	0	0	0
4	0	0	0	0
5	0	0	1	0
6	0	0	2	0
7	0	0	2	0
8	0	0	2	0
9	0	0	2	0
10	0	0	2	0

Table A.9: FN on duplication simulation samples.

Samples	PopDel	Delly	Manta	Lumpy
1	134	142	177	243
2	158	164	212	270
3	165	172	213	275
4	175	183	221	282
5	176	186	224	283
6	178	184	227	286
7	179	184	223	286
8	178	185	225	287
9	177	184	224	279
10	177	183	224	277

Table A.10: TP on inversion simulation samples.

Samples	PopDel	Delly	Smoove
1	1206	1208	909
2	1456	1460	1131
3	1566	1573	1255
4	1627	1636	1322
5	1674	1680	1372
6	1689	1696	1403
7	1708	1715	1432
8	1724	1731	1450
9	1730	1737	1460
10	1735	1741	1467

Table A.11: FP on inversion simulation samples.

Samples	PopDel	Delly	Smoove
1	0	0	0
2	0	0	0
3	0	0	0
4	0	0	0
5	0	0	0
6	0	0	0
7	0	0	0
8	0	0	0
9	0	0	0
10	0	0	0

Table A.12: FN on inversion simulation samples.

Samples	PopDel	Delly	Smoove
1	110	108	407
2	143	139	468
3	153	146	464
4	164	155	469
5	164	158	466
6	166	159	452
7	166	159	442
8	167	160	441
9	166	159	436
10	165	159	433

A.5 Tables of Simulation Benchmarks with Genotype Consideration

Table A.13: Delly genotyping evaluation on uniform deletion simulation data. undercall: number of alleles erroneously classified as non-variant. overcall: number of alleles erroneously classified as variant. misscall: undercal + overcall. Total: miscall + correctly classified alleles.

Samples	undercall	overcall	misscall	total
1	219	0	219	2632
2	439	0	439	5288
3	657	0	657	7996
4	861	0	861	10700
5	1069	1	1070	13372
6	1304	1	1305	16050
7	1515	1	1516	18762
8	1735	1	1736	21444
9	1962	1	1963	24088
10	2187	1	2188	26766
20	4338	1	4339	53434
30	6401	1	6402	80022
40	8510	3	8513	106492
50	10546	4	10550	133172
60	12571	4	12575	159722
70	14561	4	14565	186438
80	16743	4	16747	213066
90	18850	5	18855	239608
100	20952	5	20957	266070
200	41631	10	41641	531866
300	61636	20	61656	798722
400	81562	336	81898	1065522
500	101205	714	101919	1330888
600	119772	846	120618	1597062
700	139925	1646	141571	1865548
800	159574	2694	162268	2133754
900	178793	3765	182558	2401086
1000	198413	4198	202611	2667468

Table A.14: Manta genotyping evaluation on uniform deletion simulation data. undercall: number of alleles erroneously classified as non-variant. overcall: number of alleles erroneously classified as variant. misscall: undercal + overcall. Total: miscall + correctly classified alleles.

Samples	undercall	overcall	misscall	total
1	242	2	244	2634
2	470	4	474	5294
3	691	6	697	8006
4	927	7	934	10716
5	1145	9	1154	13392
6	1377	11	1388	16072
7	1597	13	1610	18790
8	1829	18	1847	21478
9	2058	20	2078	24126
10	2286	22	2308	26810
20	4499	35	4534	53496
30	6644	53	6697	80126
40	8987	79	9066	106580
50	11127	95	11222	133288
60	13324	113	13437	159866
70	15523	132	15655	186650
80	17737	152	17889	213304
90	19880	169	20049	239988
100	22073	189	22262	266512
200	45731	678	46409	533200
300	69113	1034	70147	800740
400	91844	1386	93230	1067598
500	114790	1717	116507	1333872
600	137801	2073	139874	1600730
700	159619	2454	162073	1867730
800	183208	3185	186393	2135058
900	207447	3564	211011	2401098
1000	230287	3959	234246	2667480

Table A.15: PopDel genotyping evaluation on uniform deletion simulation data. undercall: number of alleles erroneously classified as non-variant. overcall: number of alleles erroneously classified as variant. misscall: undercal + overcall. Total: miscall + correctly classified alleles.

Samples	undercall	overcall	misscall	total
1	217	1	218	2630
2	410	2	412	5280
3	602	3	605	7972
4	811	3	814	10666
5	1005	3	1008	13326
6	1207	7	1214	15984
7	1393	9	1402	18666
8	1568	10	1578	21322
9	1757	11	1768	23940
10	1956	12	1968	26600
20	3894	18	3912	53088
30	5744	27	5771	79466
40	7580	43	7623	105630
50	9481	58	9539	132144
60	11313	60	11373	158448
70	13021	68	13089	184796
80	14890	75	14965	211114
90	16772	86	16858	237432
100	18649	94	18743	263634
200	37551	184	37735	526904
300	56268	276	56544	790214
400	75227	356	75583	1053784
500	92680	444	93124	1315784
600	112174	524	112698	1579032
700	130079	632	130711	1842264
800	148692	724	149416	2105302
900	167202	796	167998	2367550
1000	185604	886	186490	2630320

Table A.16: Smoove genotyping evaluation on uniform deletion simulation data. undercall: number of alleles erroneously classified as non-variant. overcall: number of alleles erroneously classified as variant. misscall: undercal + overcall. Total: miscall + correctly classified alleles.

Samples	undercall	overcall	misscall	total
1	256	0	256	2632
2	524	0	524	5290
3	780	0	780	7990
4	1046	0	1046	10696
5	1304	0	1304	13364
6	1579	0	1579	16044
7	1843	0	1843	18756
8	2103	0	2103	21442
9	2366	0	2366	24084
10	2634	0	2634	26752
20	5124	0	5124	53384
30	7518	1	7519	79956
40	9985	1	9986	106402
50	12483	1	12484	133042
60	14875	1	14876	159540
70	17342	1	17343	186278
80	19767	1	19768	212830
90	22244	1	22245	239336
100	24695	1	24696	265756
200	48883	2	48885	531516
300	73368	4	73372	797868
400	97763	4	97767	1063926
500	122250	4	122254	1329472
600	145816	5	145821	1594722
700	170014	5	170019	1860402
800	194336	6	194342	2126116
900	218553	8	218561	2391032
1000	242594	11	242605	2656286

Table A.17: Delly genotyping evaluation on G1k deletion simulation data. undercall: number of alleles erroneously classified as non-variant. overcall: number of alleles erroneously classified as variant. misscall: undercal + overcall. Total: miscall + correctly classified alleles.

SampleNum	undercall	overcall	misscall	total
1	69	5	74	430
2	140	13	153	948
3	233	23	256	1468
4	301	27	328	1934
5	371	32	403	2384
6	438	37	475	2834
7	521	44	565	3298
8	609	50	659	3748
9	692	59	751	4224
10	773	74	847	4688
20	1568	161	1729	9456
30	2373	217	2590	14130
40	3069	261	3330	18630
50	3509	387	3896	23174
60	4152	478	4630	27934
70	4617	516	5133	32676
80	4896	589	5485	37238
90	5428	669	6097	41896
100	5832	740	6572	46564
200	10460	1681	12141	93252
300	15429	2632	18061	139948
400	19700	3259	22959	185778
500	22950	4097	27047	232502

Table A.18: Manta genotyping evaluation on G1k deletion simulation data. undercall: number of alleles erroneously classified as non-variant. overcall: number of alleles erroneously classified as variant. misscall: undercal + overcall. Total: miscall + correctly classified alleles.

SampleNum	undercall	overcall	misscall	total
1	43	6	49	428
2	65	12	77	934
3	92	23	115	1448
4	98	26	124	1894
5	112	30	142	2354
6	108	35	143	2796
7	116	41	157	3254
8	110	48	158	3698
9	119	52	171	4152
10	129	58	187	4596
20	275	162	437	9316
30	375	234	609	13928
40	492	320	812	18530
50	491	377	868	22690
60	635	539	1174	27550
70	730	627	1357	32244
80	740	706	1446	36774
90	825	802	1627	41396
100	908	938	1846	46232
200	1813	1755	3568	91320
300	2987	2819	5806	138224
400	3459	3421	6880	182810
500	4092	4521	8613	229020

Table A.19: PopDel genotyping evaluation on G1k deletion simulation data. undercall: number of alleles erroneously classified as non-variant. overcall: number of alleles erroneously classified as variant. misscall: undercal + overcall. Total: miscall + correctly classified alleles.

SampleNum	undercall	overcall	misscall	total
1	45	3	48	426
2	83	11	94	938
3	128	31	159	1474
4	163	35	198	1944
5	190	36	226	2396
6	221	51	272	2862
7	259	56	315	3328
8	287	55	342	3774
9	326	62	388	4226
10	349	70	419	4680
20	772	174	946	9474
30	1090	211	1301	14124
40	1454	256	1710	18708
50	1852	371	2223	23226
60	2299	455	2754	28030
70	2534	469	3003	32744
80	2955	573	3528	37426
90	3482	729	4211	42300
100	4030	803	4833	47084
200	8076	1731	9807	93790
300	12645	2647	15292	141038
400	16737	3442	20179	187344
500	21572	4528	26100	234890

Table A.20: Smoove genotyping evaluation on G1k deletion simulation data. undercall: number of alleles erroneously classified as non-variant. overcall: number of alleles erroneously classified as variant. misscall: undercal + overcall. Total: miscall + correctly classified alleles.

SampleNum	undercall	overcall	misscall	total
1	28	6	34	428
2	50	18	68	938
3	70	31	101	1456
4	79	41	120	1914
5	87	50	137	2368
6	95	61	156	2812
7	106	70	176	3270
8	119	83	202	3724
9	130	90	220	4166
10	145	97	242	4614
20	285	204	489	9268
30	418	323	741	13918
40	554	434	988	18456
50	676	549	1225	22850
60	753	668	1421	27538
70	892	784	1676	32330
80	1012	887	1899	36884
90	1131	996	2127	41474
100	1241	1110	2351	46094
200	2493	2208	4701	91836
300	3960	3484	7444	138328
400	4851	4437	9288	183820
500	5939	5649	11588	229828

Table A.21: Delly genotyping evaluation on duplication simulation data. undercall: number of alleles erroneously classified as non-variant. overcall: number of alleles erroneously classified as variant. misscall: undercal + overcall. Total: miscall + correctly classified alleles.

SampleNum	undercall	overcall	misscall	total
1	795	0	795	2632
2	1612	0	1612	5274
3	2400	0	2400	7964
4	3202	0	3202	10654
5	3997	0	3997	13316
6	4803	0	4803	15970
7	5610	0	5610	18680
8	6400	0	6400	21334
9	7209	0	7209	23960
10	8005	0	8005	26614

Table A.22: Manta genotyping evaluation on duplication simulation data. undercall: number of alleles erroneously classified as non-variant. overcall: number of alleles erroneously classified as variant. misscall: undercal + overcall. Total: miscall + correctly classified alleles.

SampleNum	undercall	overcall	misscall	total
1	824	0	824	2632
2	1682	0	1682	5274
3	2495	0	2495	7970
4	3324	0	3324	10664
5	4141	0	4141	13328
6	4989	0	4989	16000
7	5810	0	5810	18696
8	6643	0	6643	21376
9	7485	0	7485	24010
10	8318	0	8318	26682

Table A.23: PopDel genotyping evaluation on duplication simulation data. undercall: number of alleles erroneously classified as non-variant. overcall: number of alleles erroneously classified as variant. misscall: undercal + overcall. Total: miscall + correctly classified alleles.

SampleNum	undercall	overcall	misscall	total
1	463	74	537	2484
2	941	136	1077	4998
3	1408	203	1611	7552
4	1893	263	2156	10114
5	2350	336	2686	12618
6	2837	394	3231	15172
7	3303	456	3759	17752
8	3756	514	4270	20290
9	4232	589	4821	22762
10	4726	653	5379	25298

Table A.24: Smoove genotyping evaluation on duplication simulation data. undercall: number of alleles erroneously classified as non-variant. overcall: number of alleles erroneously classified as variant. misscall: undercal + overcall. Total: miscall + correctly classified alleles.

SampleNum	undercall	overcall	misscall	total
1	389	78	467	2476
2	728	153	881	4960
3	1065	216	1281	7520
4	1415	286	1701	10072
5	1755	372	2127	12558
6	2128	443	2571	15080
7	2482	515	2997	17636
8	2831	581	3412	20180
9	3141	662	3803	22642
10	3510	726	4236	25182

Table A.25: Delly genotyping evaluation on inversion simulation data. undercall: number of alleles erroneously classified as non-variant. overcall: number of alleles erroneously classified as variant. misscall: undercal + overcall. Total: miscall + correctly classified alleles.

SampleNum	undercall	overcall	misscall	total
1	166	0	166	2632
2	333	0	333	5282
3	500	0	500	7990
4	672	0	672	10692
5	837	0	837	13352
6	1014	0	1014	16026
7	1183	0	1183	18734
8	1338	0	1338	21410
9	1502	0	1502	24048
10	1682	0	1682	26722

Table A.26: PopDel genotyping evaluation on inversion simulation data. undercall: number of alleles erroneously classified as non-variant. overcall: number of alleles erroneously classified as variant. misscall: undercal + overcall. Total: miscall + correctly classified alleles.

SampleNum	undercall	overcall	misscall	total
1	177	90	267	2452
2	359	168	527	4950
3	548	252	800	7484
4	740	336	1076	10022
5	912	421	1333	12512
6	1109	479	1588	15072
7	1301	552	1853	17640
8	1485	612	2097	20196
9	1667	690	2357	22678
10	1857	748	2605	25230

Table A.27: Smoove genotyping evaluation on inversion simulation data. undercall: number of alleles erroneously classified as non-variant. overcall: number of alleles erroneously classified as variant. misscall: undercal + overcall. Total: miscall + correctly classified alleles.

SampleNum	undercall	overcall	misscall	total
1	544	0	544	2632
2	884	0	884	5290
3	1140	0	1140	7998
4	1431	0	1431	10706
5	1729	1	1730	13378
6	1999	1	2000	16060
7	2261	1	2262	18776
8	2520	1	2521	21462
9	2810	1	2811	24106
10	3095	0	3095	26788

A.6 Tables of Running Times and Memory Consumption

Table A.28: Delly running time and memory consumption on uniform deletion simulation data. real: wallclock time. usr: time spend in usr mode. sys: time spend in kernel mode. mem: maximum residual memory.

Samples	real [s]	usr [s]	sys [s]	mem [kB]
1	37.78	37.12	0.55	534456
2	64.94	63.73	0.84	565912
3	90.43	89.00	1.01	566968
4	115.26	113.61	1.46	567708
5	141.05	139.17	1.72	568628
6	166.45	164.17	2.13	571060
7	191.81	189.15	2.50	572848
8	217.69	214.92	2.61	573720
9	242.14	239.04	2.96	574624
10	267.82	263.61	4.00	579808
20	528.77	516.58	8.68	606624
30	784.02	767.99	12.99	615892
40	1041.84	1021.12	17.12	624200
50	1301.47	1276.70	21.63	637348
60	1564.27	1535.66	25.40	645856
70	1848.81	1815.06	29.88	654100
80	2078.57	2042.14	34.07	656096
90	2368.26	2328.77	38.36	676476
100	2606.81	2563.86	42.05	691072
200	7373.85	5657.48	299.69	1591604
300	12528.14	8851.43	696.62	1667668
400	17422.61	11786.26	1180.90	1771604
500	22212.83	14828.46	1646.83	1875212
600	24431.94	15428.14	1809.80	2083628
700	28925.30	18175.25	2347.50	2175476
800	33573.62	20950.58	2963.55	2280160
900	38037.54	23670.79	3503.84	2387408
1000	42290.08	26347.80	3840.05	2582868

Table A.29: GRIDSS running time and memory consumption on uniform deletion simulation data. real: wallclock time. usr: time spend in usr mode. sys: time spend in kernel mode. mem: maximum residual memory.

Samples	real [s]	usr [s]	sys [s]	mem [kB]
1	215.55	291.57	22.34	5364112
10	1768.86	2137.81	59.69	6808836
20	4158.96	4744.19	91.75	8243980
30	7414.23	8277.34	129.07	9905520
40	11537.82	12714.63	174.20	10901888
50	15816.34	17699.58	228.52	11599312
60	20704.71	23268.14	279.83	11826068
70	26032.19	29164.00	325.62	13034024
80	31751.32	35619.08	376.56	14972540
90	37163.67	41688.03	423.25	14937792
100	43099.55	48261.57	481.00	14080576
200	101457.43	114018.30	866.20	32238488
300	157812.09	187521.85	1396.76	63534080
400	215201.32	279437.37	1891.78	96184796

Table A.30: Manta running time and memory consumption on uniform deletion simulation data. real: wallclock time. usr: time spend in usr mode. sys: time spend in kernel mode. mem: maximum residual memory.

Samples	real [s]	usr [s]	sys [s]	mem [kB]
1	80.77	64.72	1.44	28180
2	170.71	145.72	1.97	28716
3	261.12	234.17	2.93	29268
4	351.19	323.45	3.70	29580
5	441.58	417.18	4.35	30148
6	531.51	503.04	4.82	30484
7	621.71	597.45	5.52	31044
8	712.03	685.98	6.11	32936
9	817.59[s]	777.09	6.99	35404
10	892.36	862.09	8.16	38196
20	1854.74	1794.37	17.22	62720
30	2817.53	2731.33	26.30	89044
40	3764.32	3671.23	35.18	113324
50	4741.15	4629.38	44.15	138748
60	5675.67	5535.52	53.64	158008
70	6575.44	6427.61	61.65	183796
80	7523.74	7347.66	73.54	206868
90	8380.44	8193.85	81.34	229236
100	9327.14	9117.92	89.47	251012
200	19861.51	18054.04	238.82	478776
300	30900.69	26681.30	402.83	706652
400	42007.73	35496.35	575.30	933608
500	52104.97	43990.26	738.52	1155872
600	62212.48	52660.12	901.60	1394396
700	73640.35	61731.40	1079.00	1623096
800	83876.54	70554.22	1249.73	1844356
900	95157.45	79531.51	1426.47	2067932
1000	107458.49	89403.08	1614.71	2291768

Table A.31: PopDel Profile running time and memory consumption on uniform deletion simulation data. real: wallclock time. usr: time spend in usr mode. sys: time spend in kernel mode. mem: maximum residual memory.

Samples	real [s]	usr [s]	sys [s]	mem [kB]
1	6.07	15.5	1.49	30968
2	11.87	30.5	2.66	31036
3	17.87	45.42	4.15	31188
4	23.85	60.2	5.57	31188
5	30.11	75.03	7.16	31188
6	36.14	89.98	8.62	31188
7	42.04	104.81	10.14	31188
8	48.05	120.34	11.49	31188
9	53.83	134.83	12.88	31188
10	60.17	149.74	14.24	31188
20	119.94	299.34	28.4	31188
30	179.92	447.78	42.91	31188
40	240.1	598.05	56.81	31188
50	300.17	747.86	71.06	31188
60	360.52	896.92	84.82	31188
70	420.98	1046.63	98.84	31188
80	481.32	1195.97	112.65	31188
90	542.97	1348.1	126.9	31188
100	603.13	1497.39	140.57	31188
200	1202.26	2987.09	279.23	31188
300	1803.98	4483.86	418.63	31188
400	2407.08	5975.04	556.65	31188
500	3013.54	7468.69	696.09	31188
600	3617.85	8958.94	836.66	31188
700	4221.82	10457.46	975.96	31188
800	4827.96	11959.06	1116.42	31188
900	5436.13	13460.65	1258.84	31188
1000	6043.68	14955.49	1399.4	31188

Table A.32: PopDel Call running time and memory consumption on uniform deletion simulation data. real: wallclock time. usr: time spend in usr mode. sys: time spend in kernel mode. mem: maximum residual memory.

Samples	real [s]	usr [s]	sys [s]	mem [kB]
1	4.76	4.73	0.02	13268
2	10.62	10.50	0.12	18344
3	16.12	16.03	0.08	19520
4	21.95	21.79	0.13	20816
5	27.59	27.40	0.18	22228
6	33.10	32.88	0.21	23512
7	39.39	39.15	0.23	24484
8	45.02	44.62	0.28	25548
9	50.62	50.36	0.25	27132
10	56.49	56.20	0.28	27788
20	114.99	114.26	0.68	39460
30	174.07	173.17	0.85	48440
40	236.98	235.68	1.28	72932
50	298.58	296.89	1.57	89556
60	359.71	357.74	1.92	107480
70	421.84	419.47	2.33	118080
80	488.87	486.12	2.70	134832
90	552.43	548.92	3.30	149692
100	621.84	617.78	4.01	169016
200	1357.74	1349.54	8.10	321668
300	2124.22	2111.98	12.08	479440
400	2930.00	2913.54	16.22	629972
500	3707.36	3686.31	20.76	778852
600	4476.18	4451.79	24.08	931028
700	5264.08	5235.92	27.78	1079928
800	6066.68	6034.94	31.26	1224908
900	6845.06	6809.74	34.76	1374140
1000	7628.10	7588.80	38.69	1520464

Table A.33: Smoove (all steps) running time and memory consumption on uniform deletion simulation data. real: wallclock time. usr: time spend in usr mode. sys: time spend in kernel mode. mem: maximum residual memory.

Samples	real [s]	usr [s]	sys [s]	mem [kB]
1	61.97	48.94	2.6	978800
2	174.17	147.36	10.83	978876
3	192.7	221.12	14.36	978876
4	261.79	296.35	19.31	979052
5	333.28	375.66	23.61	979052
6	396.74	454.36	27.95	979052
7	465.41	532.13	32.34	979052
8	535.45	612.98	36.58	979052
9	604.15	691.52	41.26	979052
10	672.82	768.78	45.84	979052
20	1417.72	1603.8	97.78	979052
30	2129.34	2420	144.32	979052
40	2848.22	3241.88	193.21	979052
50	3568.41	4063.82	240.93	979052
60	4318.47	4908.78	293.4	979116
70	5051.87	5764.71	343.43	979116
80	5777.22	6583.06	394.73	979116
90	6507.42	7420.97	443.79	979116
100	7233.02	8246.26	492.08	979116
200	14926.53	16944.67	1029.66	979128
300	22984.45	25926.73	1612.13	979128
400	30820.87	34782.32	2166.08	1095304
500	38723	43702.71	2721.21	1384332
600	46653.23	52667.53	3280	1678336
700	54885.91	61891.96	3873.64	2065132
800	62987.76	71157.29	4453.68	2398504
900	71116.08	80526.74	5034.73	2735192
1000	79152.38	89588.22	5584.06	3072048

Table A.34: Running time and memory consumption for different buffer sizes in PopDel Call (100 uniform deletion simulation samples). real: wallclock time. usr: time spend in usr mode. sys: time spend in kernel mode. mem: maximum residual memory.

Buffer size [30 bp windows]	real [s]	usr [s]	sys [s]	mem [kB]
10000	2801.09	2746.80	54.36	21572
20000	1684.78	1657.26	27.60	31952
30000	1284.71	1266.17	18.61	40372
40000	1103.74	1085.72	18.07	49440
50000	987.03	975.47	11.61	52752
100000	753.31	747.25	6.10	85428
200000	635.70	631.67	4.04	168688
300000	595.33	592.83	2.53	222772
400000	568.90	566.59	2.34	308032
500000	562.53	559.95	2.55	358964
600000	549.38	547.96	1.44	418428
700000	543.97	542.69	1.31	445112
800000	541.16	539.50	1.67	539636
900000	541.08	539.45	1.66	651184
1000000	538.34	536.65	1.72	673452
2000000	522.90	521.57	1.36	1250548
4000000	520.63	518.40	2.26	2151456
8000000	520.05	517.84	2.24	4222052

Table A.35: Delly running time and memory consumption on duplication simulation data. real: wallclock time. usr: time spend in usr mode. sys: time spend in kernel mode. mem: maximum residual memory.

Samples	real [s]	usr [s]	sys [s]	mem [kB]
1	64.71	47.32	7.34	671192
2	122.74	85.38	13.03	704128
3	187.45	119.64	23.73	754500
4	234.45	157.39	28.94	755360
5	282.13	106.71	113.90	756208
6	344.71	155.63	92.82	757072
7	397.33	77.47	217.27	757932
8	451.34	238.23	92.24	760744
9	492.90	312.37	79.03	761468
10	571.62	326.40	110.45	762448

Table A.36: Manta running time and memory consumption on duplication simulation data. real: wallclock time. usr: time spend in usr mode. sys: time spend in kernel mode. mem: maximum residual memory.

Samples	real [s]	usr [s]	sys [s]	mem [kB]
1	153.63	57.47	45.77	28520
2	318.74	153.76	79.75	29084
3	513.97	250.61	133.96	29752
4	679.19	363.76	131.26	30124
5	859.18	480.42	179.33	30648
6	1039.25	558.30	246.18	31292
7	1249.50	702.58	205.62	31640
8	1429.94	834.86	218.72	32328
9	1549.94	923.16	304.45	35364
10	1760.31	964.75	394.34	37772

Table A.37: PopDel Profile running time and memory consumption on duplication simulation data. real: wallclock time. usr: time spend in usr mode. sys: time spend in kernel mode. mem: maximum residual memory.

Samples	real [s]	usr [s]	sys [s]	mem [kB]
1	25.54	16.09	1.82	13344
2	50.44	33.35	4.86	13344
3	75.05	34	21.79	13344
4	100.94	46.19	27.62	13424
5	126.07	64.6	29.92	13424
6	150.97	82.96	32.09	13424
7	175.91	83.52	49.34	13424
8	200.93	102.02	51.46	13424
9	226.41	105.55	65.76	13424
10	252.35	123.11	67.43	13424

Table A.38: PopDel Call running time and memory consumption on duplication simulation data. real: wallclock time. usr: time spend in usr mode. sys: time spend in kernel mode. mem: maximum residual memory.

Samples	real [s]	usr [s]	sys [s]	mem [kB]
1	3.66	2.58	0.38	10564
2	6.54	4.69	0.61	14224
3	9.75	6.93	0.92	15428
4	12.29	4.03	5.90	18240
5	15.14	10.58	1.74	21944
6	18.21	12.91	1.71	24572
7	20.26	15.65	2.14	27152
8	23.99	17.43	2.02	29368
9	26.59	19.04	2.56	32632
10	29.59	21.53	2.47	33199

Table A.39: Smoove (all steps) running time and memory consumption on duplication simulation data. real: wallclock time. usr: time spend in usr mode. sys: time spend in kernel mode. mem: maximum residual memory.

Samples	real [s]	usr [s]	sys [s]	mem [kB]
1	151.1	107.93	48.67	6251328
2	426.04	258.34	126	6605444
3	642.91	367.15	202.82	6967992
4	853.91	494.82	265.86	6967992
5	1070.78	628.69	306.94	6967992
6	1243.42	722.25	402.06	6967992
7	1474.06	836.15	469.79	6967992
8	1682.42	979.71	512.86	6967992
9	1918.8	1088.59	567.91	6967992
10	2138.95	1246.99	602.86	6967992

Table A.40: Delly running time and memory consumption on inversion simulation data. real: wallclock time. usr: time spend in usr mode. sys: time spend in kernel mode. mem: maximum residual memory.

Samples	real [s]	usr [s]	sys [s]	mem [kB]
1	78.73	64.70	3.91	628944
2	144.28	80.57	33.97	662640
3	214.52	131.79	30.72	713308
4	270.31	66.32	140.80	714640
5	337.65	122.65	141.35	715848
6	403.08	221.60	76.91	716904
7	454.65	280.42	84.46	719956
8	526.82	274.10	127.48	721180
9	598.12	222.52	227.05	720776
10	730.21	192.34	329.61	721828

Table A.41: PopDel Profile running time and memory consumption on inversion simulation data. real: wallclock time. usr: time spend in usr mode. sys: time spend in kernel mode. mem: maximum residual memory.

Samples	real [s]	usr [s]	sys [s]	mem [kB]
1	24.64	14.81	1.99	13356
2	47.07	30.98	4.17	13356
3	70.16	35.97	15.17	13500
4	94.23	50.51	17.15	13500
5	117.36	66.67	19.95	13500
6	141.24	81.52	21.4	13500
7	165.25	96.27	23	13500
8	188.48	112.95	25.35	13500
9	211.69	129.68	27.63	13500
10	234.38	146.12	29.75	13500

Table A.42: PopDel Call running time and memory consumption on inversion simulation data. real: wallclock time. usr: time spend in usr mode. sys: time spend in kernel mode. mem: maximum residual memory.

Samples	real [s]	usr [s]	sys [s]	mem [kB]
1	4.44	3.21	0.53	10328
2	8.05	6.22	0.51	13480
3	11.67	9.01	0.74	15440
4	15.11	11.61	1.26	16668
5	18.60	14.33	1.67	18828
6	21.76	17.04	1.83	20452
7	26.59	20.17	2.33	22080
8	29.67	22.10	3.08	24948
9	32.60	25.02	3.26	26552
10	36.84	27.93	3.36	28344

Table A.43: Smoove (all steps) running time and memory consumption on inversion simulation data. real: wallclock time. usr: time spend in usr mode. sys: time spend in kernel mode. mem: maximum residual memory.

samples	real	usr	sys	mem
1	38.62	26.96	7.11	370404
2	211.47	117.57	55.18	370412
3	318.94	153.35	112.97	370412
4	417.56	219.81	121.23	370412
5	579.56	235.88	197.15	370412
6	728.56	305.18	193.47	370412
7	747.25	352.04	252.43	370412
8	886.25	402.9	299.49	370412
9	997.52	467.72	331.52	370412
10	1108.76	518.07	360.11	370412

A.7 Evaluation the Illumina Platinum Genome HG001/NA12878

A.7.1 Variant Calling Protocol

Mapping, processing and SV calling for NA12878 was performed using the same tools and settings as described for HG002 in Section 4.4.1. GRCh38 was used as a reference genome.

A.7.1.1 Call Set Overlaps

Call set overlaps for PopDel, Delly, Manta and Smoove were performed using two different GIAB truth sets: The Illumina short read reference set [1] and a PacBio long read reference call set[2]. The resulting call set overlaps are presented in Figure A.1.

[1] ftp://ftp-trace.ncbi.nlm.nih.gov/giab/ftp/technical/svclassify_Manuscript/Supplementary_Information/Personalis_1000_Genomes_deduplicated_deletions.bed

[2] ftp://ftp-trace.ncbi.nlm.nih.gov/giab/ftp/data/NA12878/NA12878_PacBio_MtSinai/NA12878.sorted.vcf.gz

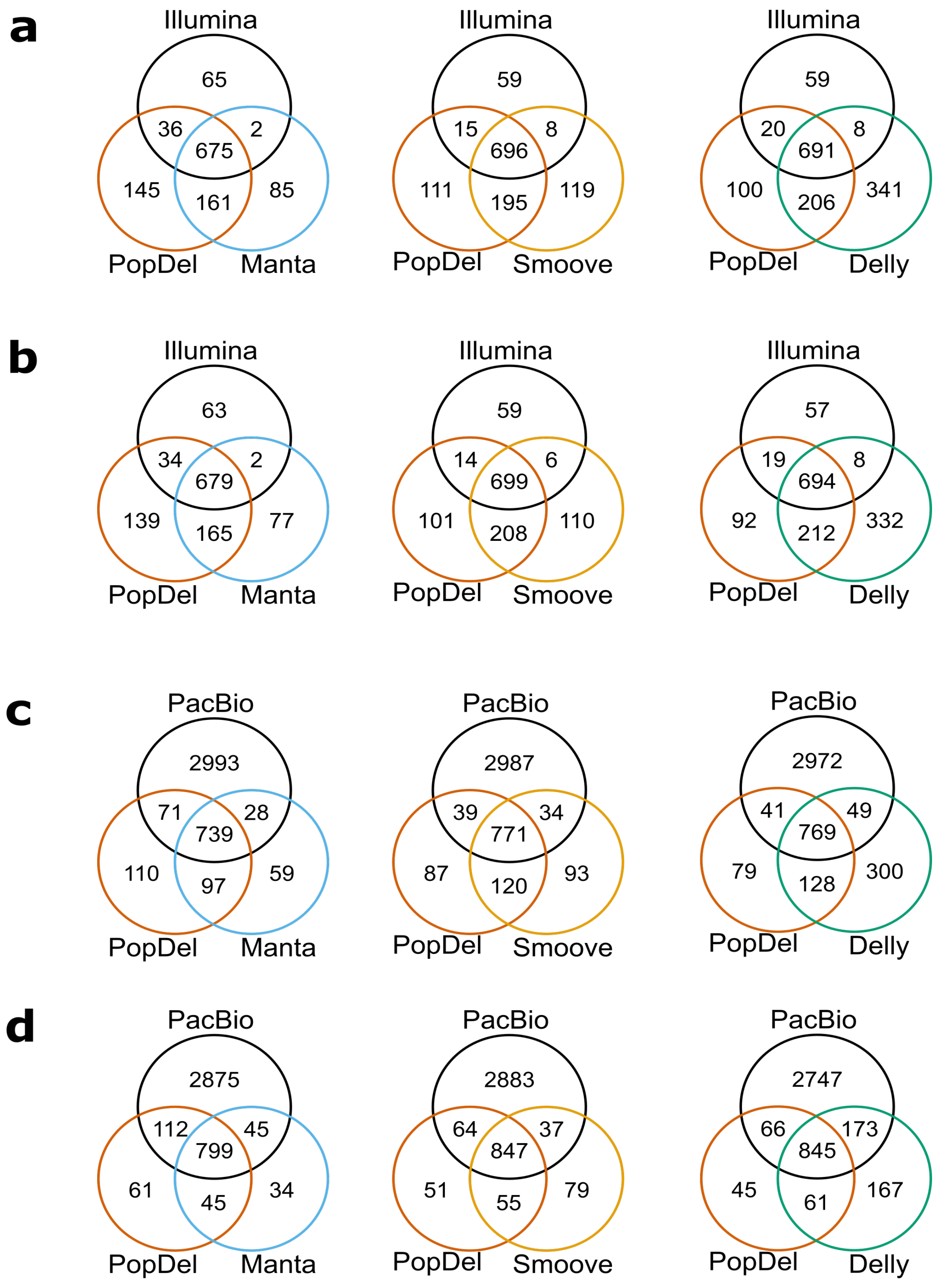

Figure A.1: Venn diagrams of call set overlaps on NA12878. **a**, **b** Comparison with Illumina short read reference set. **c**, **d** Comparison with PacBio long read reference set. **a**, **c** Comparison using a minimum reciprocal overlap of 80%. **b**, **d** Comparison using a minimum reciprocal overlap of 0.1%.

Index