BRNO UNIVERSITY OF TECHNOLOGY

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

FAKULTA ELEKTROTECHNIKY
A KOMUNIKAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF BIOMEDICAL ENGINEERING

ÚSTAV BIOMEDICÍNSKÉHO INŽENÝRSTVÍ

ADVANCED COMPUTATIONAL METHODS FOR CNV DETECTION IN BACTERIAL GENOMES

POKROČILÉ VÝPOČETNÍ METODY PRO DETEKCI CNV V BAKTERIÁLNÍCH GENOMECH

DOCTORAL THESIS

DIZERTAČNÍ PRÁCE

AUTOR PRÁCE AUTHOR	Ing. Robin Jugas
ŠKOLITEL SUPERVISOR	Ing. Helena Škutková, Ph.D.

BRNO 2023

Abstract

The focus in the field of structural variations is mainly focused on human genomes. Thus, detecting copy number variation (CNV) in bacteria is a less developed field. Commonly used CNV detection methods do not consider the features of bacterial circular genomes and generally, there is a space to improve performance metrics. This thesis presents a CNV detection method called CNproScan focused on bacterial genomes. CNproScan implements a hybrid approach combining read depth and read pair signals. It considers all bacteria features and depends only on NGS data. Based on the benchmarking results, the CNproScan achieved very well in various conditions. Using the read pair information, the CNVs are classified into several categories. Also, compared with other methods, CNproScan can detect much shorter CNV events. Because of the necessity of merging not only the various feature signals but also the results of different algorithms, the thesis also introduces a pipeline called ProcaryaSV developed to easily employ five CNV detection tools and merge their results. ProcaryaSV handles the whole procedure from quality check, reads trimming, and alignment to the CNV calling.

Keywords

Next-generation sequencing, Structural variation, Copy number variation, Bacteria

Abstrakt

Hlavní pozornost v oblasti strukturálních variací je zaměřena na lidské genomy. Detekce změny variace počtu kopií (CNV) u bakterií je tedy méně rozvinutou oblastí. Běžně používané metody detekce CNV neberou v úvahu specifika bakteriálních kruhových genomů a obecně existuje prostor pro zlepšení metrik výkonnosti. Tato práce představuje metodu detekce CNV nazvanou CNproScan zaměřenou na bakteriální genomy. CNproScan implementuje hybridní přístup kombinující signály hloubky čtení a párů čtení. Bere v potaz všechny vlastnosti bakterií a využívá pouze sekvenační data. Na základě výsledků ze srovnání dosáhl CNproScan velmi dobrých výsledků v různých podmínkách. Pomocí informací z párových čtení jsou CNV klasifikovány do několika kategorií. Ve srovnání s jinými metodami může CNproScan také detekovat mnohem kratší CNV. Vzhledem k nutnosti slučovat nejen signály různých přístupů, ale také výsledky různých algoritmů, dizertační práce také představuje pipelinu nazvanou ProcaryaSV vyvinutou k detekci CNV s využitim pěti nástrojů a slučování jejich výsledků. ProcaryaSV se stará o celý postup od kontroly kvality čtení, ořezávání konců čtení, zarovnání čtení až k detekci CNV.

Klíčová slova

sekvenování nové generace, strukturální variace, variace počtu kopií, bakterie

Bibliographic citation

JUGAS, Robin. Advanced Computational Methods for CNV Detection in Bacterial Genomes. Brno, 2023. Available also at: https://www.vut.cz/studenti/zav-prace/detail/154368. Doctoral Thesis. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Department of Biomedical Engineering. Supervisor Helena Škutková.

Declaration

Acknowledgment

Here, I would like to thank several people without whom I would not have been able to finish this thesis. Firstly, I would like to thank my supervisor Ing. Helena Škutková, Ph.D. for her long-term mentoring and for leading my doctoral studies. Also, thanks to the UBMI colleagues and namely Ing. Karel Sedlář, Ph.D. who suggested to me to follow the Ph.D. degree. Most importantly, I would like to thank my family for their long-life support in my university studies.

Brno, August 7, 2023 --------------------------------------

Contents

INTRODUCTION

The topic of this thesis is the detection of copy number variations in bacterial genomes. The copy number variations (CNVs) are a subgroup of a large field of structural variations (SVs). The structural variations are largely studied, yet there are still many gaps in the knowledge about them. This is even more factual for structural variations in bacteria. Despite that the first gene amplification was observed in Escherichia coli back then in 1963, this field of research is less developed in bacteria compared to the advancements in human or other eukaryotic genomes.

However, CNVs play an important role in the bacteria. They have a direct impact on protein production. In the long term, this has an impact on evolution and specialization. The short-term adaptive gene duplication can cause antibiotic resistance, which is an emerging issue.

Sequencing is a common way how to study these organisms and became substantially cheap. Two ways of sequencing bacterial genomes are being done. The sequencing of bacterial isolates or whole bacterial communities. The thesis deals with the first one as it enables the detection of structural changes in the genome such as copy number variations.

The lesser attention paid to structural variations in bacteria could be partly caused by technical difficulties detecting small rearrangements with short-read sequencing. Right now, we are at the breaking point between the massively used short-read next-generation sequencing, and the long-read third-generation sequencing. However, the inertia in the field is large and the next-generation sequencers are abundantly present and used in the labs. Furthermore, next-generation sequencing produces high throughput data necessary for copy number detection.

Firstly, I define what structural variations and copy number variations are. The main source of research is papers focused on human genomes, as these are overabundant. However, many features of structural variations are the same for eukaryotic and bacterial genomes. The specifics of bacterial genomes are described in a special subchapter.

In the second chapter, I describe the whole topic of detection of the structural variations. This chapter approaches the topic starting with laboratory aspects and moving to bioinformatical aspects of structural variations detection.

The practical part of the thesis follows in three chapters. Firstly, I pay attention to some underlying theories that could not be placed previously. In the following two chapters, I describe two presented bioinformatical tools.

The first one is a novel algorithm for CNV detection named CNproScan. There were several reasons to create it. First, the majority of tools are aimed at large, mainly human, genomes. They require specific types of inputs and dominantly rely on paired sample-reference samples, e.g., tumor-normal tissues. Also, they are intended to detect large rearrangements, and they are not scaled to small copy-number events. However, large CNVs are rare in prokaryotes. Second, there are not enough detection tools aimed at bacteria genomes, and some of the already published ones are already deprecated. Also, based on the reviews, there is only a small overlap between the results of various tools. A high false positive discovery is a common issue. Third, bacterial pathogens pose still a highly deadly risk. In 2019, they caused 13.6% of all global deaths. Five bacteria – *S. aureus, E. coli, S. pneumoniae, K. pneumoniae, and P. aeruginosa*, were responsible for more than half of all cases. The bacteria pathogens were the second leading cause of death after ischemic heart failure. As mentioned previously, the CNVs can play a role in antibiotic resistance, bacteria adaptation, and specialization. The issue of bacteria drug resistance is present and emerging. Thus, there is a serious need to develop tools aimed at the detection of bacterial CNVs. All these aspects lead to the development of a new tool which was called CNproScan, derived from the words Copy Number prokaryotic Scanning.

The second tool is a pipeline for the alignment and detection of CNVs and SVs, named ProcaryaSV. The reasons to create the ProcaryaSV pipeline were two. It was more convenient to create a reproducible workflow than running the various scripts every time some parameter changed. Secondly, during the literature research, I came across the topic of merging not only the detection approaches but also the standalone detection tools. This idea origins in the results of multiple reviews which show how little CNV and SV overlap across multiple detection tools

The presented tools extend the scope of tools for a microbiologist to study bacterial organisms. While CNproScan detects deletions and duplications, the ProcaryaSV pipeline enables the detection of inversions and insertions by combining multiple detection tools.

1 STRUCTURAL VARIATIONS

While the focus of the thesis is copy number variations, they are inseparably tied to structural variations. There are predominantly mentioned together but the field of CNVs is generally more developed. In the following chapter, the classification of the structural variations from various aspects is expanded. Other related topics are mentioned too, such as the process by which SVs are created and their impact on phenotype. The major source of knowledge comes from human genomes focused research. While there are multiple similarities between eukaryotic and prokaryotic structural variations, there are also differences and these are described in the chapter focused solely on structural variations in bacteria.

1.1 Classification of SVs

The SVs can be classified into several categories. The most common one is classification regarding copy numbers into **balanced** and **unbalanced events**. Another criterion classifies SVs into **single** and **complex SVs** which consist of more underlying simple SVs. The SVs can be classified based on their size as **fine-scale**, **intermediate-scale**, or **large-scale**. SVs can also be categorized based on the process of creation as **cut-and-paste** and **copy-and-paste**. Structural variation is observed as a junction between two breakpoints in the genome. When the sequencing read spans over a breakpoint junction, it leads to discordant features compared to the other read alignment features. This junction is defined by its orientation, space between breakpoints, etc. [1]

The canonical types of SVs are **deletions, insertions, duplications, inversions, and translocations.** The minimum length of such events is not exactly specified. The initial size threshold was 1 kbp, later decreased to 50 bp but nowadays the SVs are all variants that are not single nucleotide variants (SNV). A more accurate definition than by size could be by a mechanism of creation of that SV. Small indels are created by replication slippage, while larger CNVs are created by homology recombination [2].

The inversions and translocations classify as a balanced type of SVs, whereas the rest as unbalanced SVs. The deletions and duplications are also called **copy number variations** (CNVs) especially when they include gene regions. [3]

Another large group is complex structural variations, which consist of multiple canonical SVs organized in many ways. [4]

The basic illustration of various SV types is in Figure 1.1. The upper boxes represent the reference genome, while the lower boxes represent the sample genome situation. As you can realize, the definition of SV is tied to some reference situation. This reference is another genome, another sample of a different location or time, or a pool of samples merged.

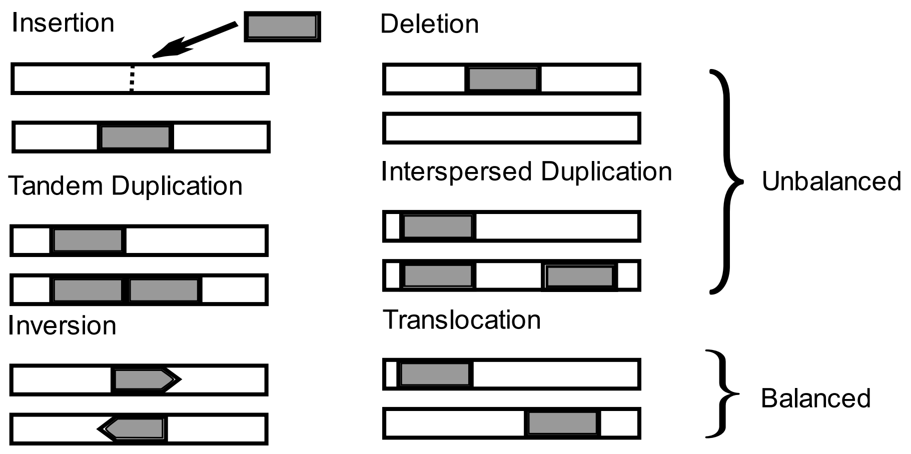


Figure 1.1 – Overview of SV types. Condition between referential and analyzed genome

1.1.1 Unbalanced SVs

Insertion, together with **deletion**, is the most common type of SV in the human genome [5]. It represents the gain of genomic genetic material compared to the reference. [6] Insertions can be further divided based on the source of the genetic sequence into types: mobile element insertions (MEIs), nuclear insertions of the mitochondrial genome (NUMTs), viral insertions (VEIs), and insertions of unspecified sequence [3]. The direct detection of insertions is limited by the next-generation (NGS) approach, which limits the size of detectable insertion to the length of the library fragment. No upper threshold is given for deletions [7]. It is demonstrated by two breakpoints [8].

Deletion is a genomic event where a segment of DNA is cropped from the genome and the adjacent bases fuse next to each other. It creates a single breakpoint and it occurs at the same chromosome [8]. The difference between a small insertion-deletion (indel) event and larger deletion is a matter of definition. However, the most suitable definition depends on the underlying mechanism of creation. The small indel is created by slipped strand mispairing (also as replication slippage), while structural deletion is created more likely by DNA repair mechanisms error [2]. The commonly stated size threshold is often 50 bp, which originates from the features of NGS sequencing.

Repeats or **duplications** are parts of the genome that are duplicated. It can be in tandem or interspersed matter, whether they are adjacent or placed far apart. The majority of them are noncoding DNA, but some are functional genes [9], [10]

Interspersed sequences consist of **long interspersed elements** (LINEs, >300bp), and **short interspersed elements** (SINEs, 100-300bp), which are both types of repetitive noncoding DNA. LINEs are moderately repetitive and exist in less than a hundred of a thousand copies. Such an example can be the L1 (or LINE-1) element which is 7 kbp long and has between 20-50 thousand copies in mammalian genomes. Their assumed origin is retroviruses. SINEs are highly repetitive and exist in hundreds of thousands of copies. The most known example is the Alu element, which is 300 bp long and named after the restriction enzyme involved with it. It exists in 300-500 thousand copies of the human genome and covers approximately 11% of the human genome. [9], [11]

Tandem repeats form several classes: **short tandem repeats** (STR, 1-6 bp, called microsatellites), and **variable number tandem repeats** (VNTR, >7 bp) [10]. They usually form long clusters of tandem repeats without gaps in between (called satellites). Both STR and VNTR are noncoding and belong to the most mutable regions in the genome. [9], [10]

1.1.2 Balanced SVs

Inversions are balanced rearrangements. An inversion is an event when a segment of DNA is inverted (rotated) in its place. It creates two breakpoints and occurs at the same chromosome [8]. Inversions are not associated with a copy number change. They could affect gene expression by breaking coding regions, but they predominantly have no phenotypic effect. However, they can serve as a predisposition to further rearrangements. Inversions can be involved in the centromere region of a chromosome (pericentric inversions), or the rest of the chromosome (paracentric inversions). [6], [12]

Another balanced SV is a **translocation**. Translocation can be of two types: intra-chromosomal, when a segment stays on the same chromosome but is often inverted, and inter-chromosomal when a segment is moved to another chromosome. Both types create a single breakpoint. [8]

1.1.3 Complex SVs

Compared to canonical SVs, the breakpoints of **complex SV** (cxSV) cannot be defined by a single SV event. The scale of complex SV is large, they can involve long-distance rearrangements but also multiple rearrangements occurring at a single locus. Complex SVs are larger than canonical SVs and there are approximately 14 cxSVs in every human genome, while thousands of canonical SVs. Interestingly, the majority of cxSV contains inversion. [4], [13]

Although it is expected that complex SVs are created by a single event, this fact is difficult to prove. Germline (inherited) SVs arise most likely through two mechanisms during replication: fork stalling and template switching and microhomology-mediated break-induced replication. Complex SVs can be created when there are multiple switches on the replication fork. These switches can be even between distant parts of the genome. Somatic mutations are less limited as they do not undergo meiosis, but they undergo various selective pressures and influences. The commonly accepted mechanism for the creation of complex somatic variations is chromoanagenesis, which includes three often separately described events: **chromothripsis**, chromoanasynthesis, and chromoplexy. [13]–[17]

The most commonly mentioned chromothripsis is a single catastrophic event when a chromosome is fragmented and then repaired with many errors. This event is likely caused by external factors, e.g. radiation. The breakpoints following this event are clustered in a non-random way. The copy number exhibits only two values, gain, and loss, but never both simultaneously, and are divided by untouched segments. Also, losses are derived from the same parental chromosome, and heterozygosity is kept at untouched segments. These observations point to chromothripsis rather than a sequential mutation. [4]

Detection of complex SVs is a challenging task, which usually requires manual curation and observation of breakpoints belonging to canonical SVs. The SVs believed to be canonical are first filtered out. Also, SVs in poor-quality regions are omitted. The SVs overlapping with previously detected CNVs in a healthy pool of samples are also omitted [18]. The updated 16 types of cxSV were recently presented in [13] based on an analysis of nearly 700 samples.

1.1.4 Copy Number Variation

Copy number variation (CNV) is usually mentioned as a subtype of SVs. It is a deletion or duplication that involves genes. Other commonly used terms are **copy number gain and loss**. Sometimes, the copy number alteration (CNA) is used concerning somatic-only events, compared to germline CNVs in cancer research. The definition of CNV by size had historical development, tied with technological advancement. As the development enabled the detection of shorter and shorter CNVs, their definition by size also decreased. Commonly stated size is 50 bp to millions of bp. [19]

While CNVs are thought of as deletions and duplications, there is a debate about whether to include **indels**. Indels are much smaller than the lower size threshold of CNV. An important reason why not to include them is a distinct mechanism of creation of both events. The origin of indels is mainly replication based while the origin of CNVs is mainly homologous recombination. Although, the mechanisms can be shared across various events. The restriction of size itself is a matter of debate. [19]

An interesting tract about narrowing the definition of CNVs compared to SV can be found in a review by Pös [19]. The authors cling to the definitions of CNVs similar to the general definition of unbalanced SVs. This describes CNVs as the relative difference in copy numbers of specific DNA sequences among individuals or distinct populations. Or in other words as variation contributing to the copy number change. Then, the terms deletion, insertion, duplication, gain, and loss are meant in a molecular phenotype context of CNVs. The discussion could be also taken about specifying the impacted regions of the genome in terms of variation impact. [19]

Similarly to SVs, CNVs can be **recurrent** and **non-recurrent** (explained further). CNVs can be both inherited or sporadic. The expected lower bound rate of CNV mutations per haploid genome is 3×10^{-2}. [19], [20]

1.2 Mechanisms of Creation

The origin of the creation of structural variants dwells in the genome architecture and mainly in the existence of **segmental duplications** (SDs), a subtype of low-copy repeats (LCRs). These are blocks of DNA in size of 1 kbp to 400 kbp, that occur at multiple places in the genome and have very high levels of sequence identity (homology >90%). The regions flanked by segmental duplications are possible targets of **nonallelic homologous recombination** (NAHR). The analyses showed that they make up around 5 % of the human genome. For the larger picture, up to 50% of the genome consists of repeat sequences. [6], [21]–[24]

Segmental duplications can be located multiple times at a single chromosome (intrachromosomal duplications), or at nonhomologous chromosomes (interchromosomal duplications). Contrary to tandem duplications, they are interspersed across the genome. The multi-genomic placements and high sequence homology are the main substrate for the NAHR mechanism and thus play an important role in the creation of SVs. It can be assumed that SVs are not random events, but their origin is in predisposition to genomic rearrangement due to segmental duplications at a given locus. When visualized, the segmental duplications are often the hotspots (sites of predominant occurrence) of other SVs. [6], [21]–[24]

The parent category of segmental duplications is low-copy repeats. These are over 10kbp long and they have a mosaic architecture consisting of hierarchical clusters of directly and indirectly orientated segments, while SDs have simple architecture. It was observed they also overlap with positions of frequent genomic reorders. Similarly to SDs, both regions have a negative impact on local genome stability. [24]

The genomic rearrangements can be **recurrent** and **non-recurrent**. Recurrent ones have the same size, position, and content in unrelated individuals. Simply said, they occur with a certain frequency in the population. Oppositely, non-recurrent rearrangements have a unique size, position, and content at a given genomic position in unrelated individuals. [24]

The recurrent deletions and duplications have both breakpoints positioned within the directly oriented segmental duplications. The major driver for recurrent events is the NAHR. The non-recurrent have their breakpoints spread out. If there is a segmental duplication, the breakpoints cluster in their neighborhood. The majority of non-recurrent events are driven by **replication-based mechanisms**, i.e. FoSTes/MMBIR (fork stalling and template switching/microhomology-mediated break-induced replication). [22], [24] The most studied event is a **double-stranded break** (DSB), where both strands of DNA are broken. Defective reparation of DSBs is tied with various disorders and is a major driver of cancer. The major source of DSBs is when during replication forks encounter damaged bases which leads to fork collapse. Less frequent exogenous sources are ionizing radiation and chemotherapeutic drugs. In vivo, the frequency of DSB is high and the replication of strands is considered to be discontinuous. In some scenarios, the DSB is programmed by the cell. Like during meiosis when DSB repairs are essential for chromosome segregation. [25], [26]

The two major mechanisms of DSB reparation are two pathways: **homologous recombination** (HR) and **non-homologous end-joining** (NHEJ). The presence of many proteins, e.g. Rad51 in eukaryotes and RecA in prokaryotes for NAHR, is required for the reparation process. [26], [27]

The **homologous recombination** (HR) mechanism requires a homologous sequence serving as a template for DNA reparation. The required length of a homologous template is 50 bp for *E.coli* and up to 30 bp in the human genome. As a template, the sister chromatid is predominantly used followed by the homologous chromosome. The **nonallelic homologous recombination** is a special case of HR when the homology template is in a non-allelic position. Under usual circumstances, the NAHR does not change the genome structure and is accurate. However, structural change can occur if the repaired sequence and the template are located at distant positions. When segmental duplications have a high sequence identity of over 97 % and are located within 10 Mbp, the misalignment of chromosomes can happen and this further mediates the NAHR leading to unequal crossing over. Crossing over between homologous chromosomes can lead to a loss of heterozygosity (LOH) if the chromatids segregate during mitosis. NAHR between directly oriented segmental duplications can result in deletions or reciprocal duplications of the regions in between. NAHR between inverted segmental duplications leads to the creation of inversion. The more complex structure of segmental duplications leads to combinations of these events. The NAHR is behind the majority of recurrent rearrangements. It is active only during the S and G2 phases of cell life. NAHR is used not only to repair DSB but to correct broken replication forks in the process of break-induced replication (BIR), which can also lead to SV. [22], [27]–[29]

Contrary to HR, the **non-homologous end joining** (NHEJ) is independent of the presence of segmental duplications. It also does not require a homologous template to guide the reparation. NHEJ repairs the broken ends of DNA strands by direct resealing. It is the simplest and fastest pathway and the most commonly used, although it can lead to the potential loss of genetic information causing up to 4 bases deletions. It can also insert new genetic information from mitochondria or retrotransposons. It is active during all phases of cell life. [22], [27]–[29]

Similar to NHEJ is **microhomology-mediated end joining** (NMEJ), which requires 5 to 25 bp homology templates, and different proteins. Thus, the deletions possibly caused by NMEJ are longer. [28]

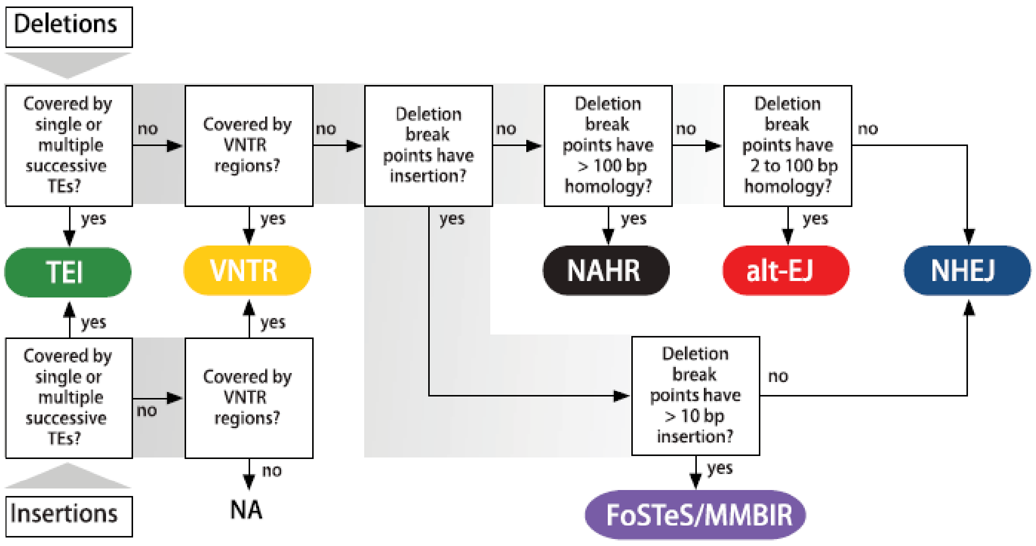
If sister chromatids lose their telomeres during DSB, they will fuse and create a dicentric chromosome. During anaphase, when chromosomes are separated, the dicentric chromosome will break in a random location. The chromosome has again an unprotected end which will merge into a new dicentric chromosome after another replication. This creates large inverted duplications. The process is called the breakage-fusion-bridge cycle. It is commonly observed in human cancer cells. [28]

Another class of mechanisms inducing copy number changes is **non-homologous replicative mechanisms**. These events happen during the replication of the DNA. When there is high sequence identity in expectedly single-stranded DNA during replication, e.g. Okazaki fragments, the fragments in between are deleted or duplicated. This responsible underlying mechanism is replication slippage (or template switching). In *E.coli*, the replication slippage depends on the homology length and distribution and requires an absence of RecA protein. The process is more abundant in prokaryotes cause it is limited by replication fork length and CNVs in the human genome are out of size achievable by replication slippage.

A more complicated form of replication slippage is not limited to a single replication fork but happens between more of them. This mechanism is called **fork stalling and template switching** (FoSTeS). During FoSTeS, the 3' primer end can switch to another single-stranded DNA template in some nearby fork. The FoSTes is considered a non-replicative mechanism based on further evidence, mainly that amplified units have only short microhomology, thus no homologous recombination is likely involved. [28]

Another mechanism is **microhomology-mediated break-induced replication** (MMBIR). It is a special case when important proteins are downregulated due to cellular stress, so break-induced replication (BIR) would likely not be possible. Instead, the 3' end of the collapsed fork will switch to another close single-stranded template-sharing microhomology. This can be the ssDNA of the lagging strand or part of ssDNA under excision. The deletion or duplication is decided on the location of the new fork: upstream causes deletion, and downstream causes duplication. The orientation of a new segment is caused by the strand which is involved, lagging, or leading. The FoSTeS/MMBIR model is considered to be a major player in the creation of CNVs and also in the creation of segmental duplications. [22], [30], [31]

Another source of genome rearrangements is retrotranspositions. Mainly, the L1 sequences play a role in insertions and deletions resulting in smaller events around 5 kbp. [4], [7]



Deletions
Covered by single or multiple successive TEs?
no
Covered by VNTR regions?
no
Deletion break points have insertion?
no
Deletion break points have > 100 bp homology?
no
Deletion break points have 2 to 100 bp homology?
no
yes
TEI
yes
VNTR
yes
yes
NAHR
yes
alt-EJ
NHEJ
yes
yes
Covered by single or multiple successive TEs?
no
Covered by VNTR regions?
Deletion break points have > 10 bp insertion?
no
no
yes
Insertions
NA
FoSTeS/MMBIR

Figure 1.2 – Rules for assigning the mechanism of SV creation, taken from [37]

1.3 Impact of SV

The definition of SVs is not tied or restricted to the gene boundaries. However, the genes are affected by SVs if they overlap. Coding regions of many genes are under the change of copy number through CNVs. Unbalanced SVs can add or remove copies of genes which leads to changes in gene dosage, gene expression, and phenotype depending on a gene type. Several studies observed a correlation between gene copy number and mRNA expression levels. The effect can be both limited-expression or over-expression. However, not all genes under copy number change demonstrate altered expression. There is even a small group of genes with inversely proportional expression to their copy number [32].

The **dosage effect** includes the impact of CNVs on gene expression. This comprises both changed levels of expressions but also modified products of transcription. Deletion of a regulatory element of a gene will lead to a change of expression, potentially complete silencing. The duplication of a regulatory element together with a gene region will lead to an increased gene product. Contrary, the deletion of a regulatory element with a gene region means no product. Insertions, deletions, or inversions overlapping with only part of a gene can lead to the creation of variant gene products through exon shuffling, splice variants, or novel gene fusions. However, the majority of such constellations are nonfunctional unless the open reading frame is functional. A gene fusion is a common type of event. [32]

The **positional effect** includes a change of expression affected by rearrangements outside of a gene. Deletions can uncover recessive alleles by deleting only one allele and affecting phenotype. However, deletions are generally a negative selection. Predicting the phenotype is challenging because of the existence of haploinsufficient genes (the half dose is detrimental) and dosage-sensitive genes (both increasing or decreasing dosage is detrimental). [6], [32]

SVs are commonly studied in cancer research. This requires the classification of SVs as germline (inherited) or somatic (acquired). Two samples are sequenced, a normal sample (usually blood) and a tumorous sample containing the tumor cells. The normal sample is usually sequenced at lower coverage while the tumor sample requires high coverage because of the mixture of cells in the resected sample and other specifics. An alternative approach is to replace the normal sample with a reference dataset of SVs. [33]

SVs also play a role in speciation in many animals through the process of reproductive isolation [34].

One of the recent and valuable tools in cancer research is **fusion genes** detection. The fusion gene is formed by multiple possible mechanisms, i.e. by chromosomal rearrangement or by non-structural aberrations such as cis-splicing (same chromosome) and trans-splicing (different chromosomes) or transcriptional read-through. Thus, we can view it as a direct effect of SV on gene dosage creating new, disrupted, or fused RNA transcripts of two originally independent genes. This later leads to the synthesis of abnormal or chimeric fusion proteins potentially modifying the original function. [35]

1.4 SV Annotation

Structural variation annotation is a process of assigning information to detected SVs. Multiple categories can be assumed based on a research subject. The basic annotation is based on SV-type classification. This is expanded by classifying into more complex SV subtypes, e.g., coding or noncoding rearrangement, gene fusions, gene duplication, or deletion. We can assign the effect on translated protein, and which type this effect will be, such as a change of the structure of the protein and impact on gene expression (dosage effect). There can be also no effect. The change of protein sequence can lead to gain or loss of function. [36]

Three major mechanisms linked to SV creation are homologous recombination, nonreplicative nonhomologous repair, and replication-based mechanisms. [37] By observing genomic features and sequences around breakpoints we can deduce the possible creation mechanism responsible for this SV. The decision tree can be constructed to assign a deletion or insertion of up to six possible mechanisms of creation. The observed features are overlap with the transposable elements (TE) or variable number tandem repeat (VNTR) regions, presence of homology, or small insertion inside or nearby of breakpoints. [7], [37]

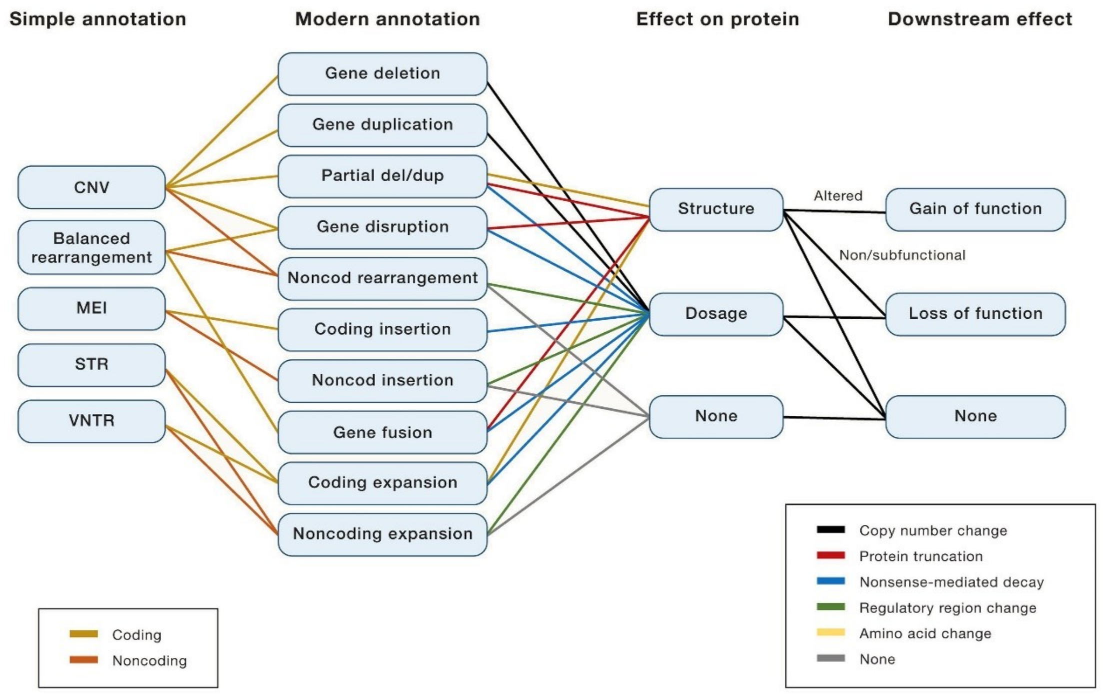


Figure 1.3 – Functional annotation and downstream consequences of SVs, taken from [36]

1.5 Population Studies

After the finished sequencing of the first human genome in 2004 [38] and the international HapMap project focused on the discovery of common SNPs in 2007 [39], several groups focused on sequencing a much higher amount of genomes. An ultimate goal is to capture all kinds of genetic variations in the population. The most influential project was **the 1000 genomes project** (1KGP) which started in 2008. They published their results in several phases but in the end, they sequenced 2,504 human genomes and analyzed thoroughly the human genetic variation including structural variation and CNVs. [40]–[43]

In 1KGP they used short-read Illumina sequencing data at ~7× coverage combined with long-read single-molecule PacBio sequencing. They discovered a combined set of 68 thousand SVs, defined as variants over 50 bp. These were predominantly deletions (42,279), followed by duplications (6,025), CNVs (2,929), inversions (786), mobile elements insertions (16,631), and nuclear mitochondrial insertions (168). Furthermore, the analysis brought useful hindsight into structural variations in the human genomes at various levels. The samples originated from 26 ancestry populations and the SVs could be stratified based on variant allele frequency (VAF) in distinct populations. The functional impact of SV was analyzed too by overlapping with already known functional elements of the genome. They pointed out other than commonly known mechanisms for the creation of SV hotspots and described some of the early complex SVs. However, the low coverage design is limiting in many aspects of discovery SVs. [43]

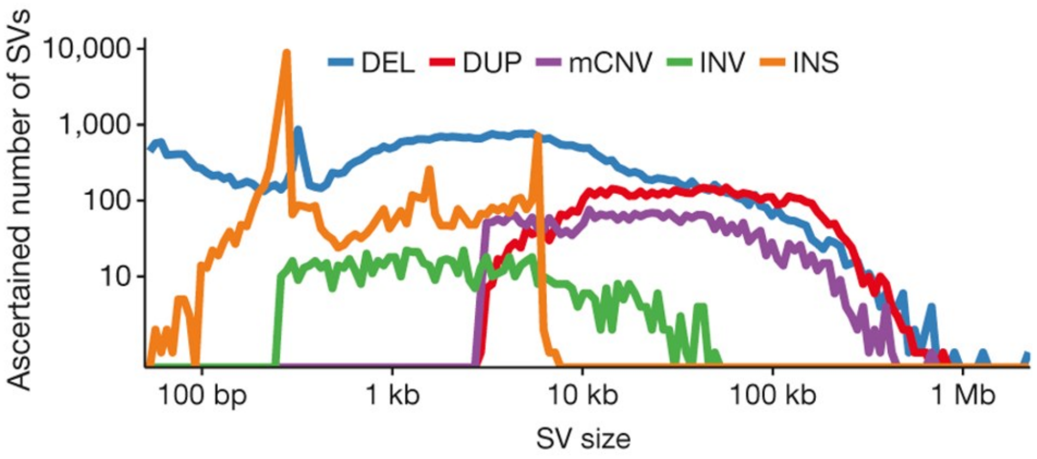


Figure 1.4 – SV types and lengths representation from the 1KGP project, taken from [43]

Another large study was the 2016 sequencing of 10,545 human genomes at ~40× coverage. The study was focused mainly on SNV and the evaluation of detection quality metrics. However, they performed SV and CNV detection on the 200 replicates of the common reference genome NA12878 (Genome in the Bottle, GIAB) and concluded unsatisfactory results of detection from short-reads for clinical practice (precision <77%, recall <36%). The use of orthogonal technologies was recommended by the authors for confirmation. [44]

The largest study of SVs up to that time is the 2018 analysis of 17,795 human genomes. They focused on discovering rare SVs which are usually recent or denovo mutations. They obtained a set of 158 thousand rare SVs, mostly deletions (54.5%) followed by duplications (42.2 %). It was discovered average human individual caries 2.9 rare SVs altering coding regions. These SVs affect on average 4.2 genes. Otherwise, they observed a mean of 4,442 nonrare SVs per genome consisting of predominantly deletions (35%), mobile element insertions (27%), and tandem duplications (11%). The group sequenced samples at higher coverage of over 20× which boosted the sensitivity for rare SVs detection. They applied mapping to two versions of the human reference genome and published a new tool for SV detection in population studies called svtools. [45], [46]

Several other smaller studies were also published. These are often focused on nationally. In 2014 the Genome of the Netherlands published results of genetic variation from 250 family trios (769 samples) at an average coverage of 13×. They discovered a high overlap of found genetic variants with European samples of the HapMap project (98.2%) and with Europen samples of 1KGP (71.1%). However, they also discovered that 93.3% of deletions were novel in comparison with 1KGP. The sequencing of family trios (mother-father-child) enabled the accurate detection of de-novo mutations. [47]

Among other national studies which performed SV detection are the 2017 Swedish population study of 942 genomes [5], the 2015 analysis of 1,070 Japanese genomes [48], the 2015 study of 10 Danish family trios [49], and the 2018 Korean study of 50 genomes [50]. The overview of a multitude of SV reference datasets is listed in [51].

1.6 SV Databases

Several databases of SVs exist and they serve in discovering new structural variants and genotyping. The first database, the **Structural Variation Database**, was part of the 2005 study on segmental duplications and comprehended the 119 detected SDs in 47 samples and previously detected 297 SVs from fosmid paired-end reads [21], [52].

The initial effort to a full-scale database started in 2006 with **The Database of Genomic Variants** (DGV), which comprehended results from mainly array-CGH experiments published in peer-reviewed journals. The majority of records in 2006 was 1,207 CNVs and very few other variants, e.g. 37 inversions. [53]

In 2013, the DGV contained data from 55 studies and more than 2.5 million entries. The database is curated and datasets are checked for accuracy. The studies are accessed from the archival SV databases, the NCBI's **dbVar**, and EBI's **DGVa** (**Database of Genomic Variants archive**), which are not curated. The majority of data are now from next-generation sequencing and some initial results from microarrays with poor resolution and high false-positive rate were removed from the database, pointing out the accuracy of the early results. [54]

Two databases share a data model, the dbVar maintained by the National Center for Biotechnology Information (NCBI) and DGVa maintained by the European Bioinformatics Institute (EBI). Both are uncurated and serve also as repositories of detected structural variants, but not the sequencing data itself. The researchers submit the variants formatted as tab-delimited or VCF files. Both databases support only human samples, the non-human samples are directed to be submitted to the **European Variation Archive** (EVA), which now serves as DGVa replacement and also fur human samples. [55]

The Genome Aggregation Database (gnomAD) is a database from both exome and whole genome high coverage sequences. It comprehends a diversity of large-scale sequencing projects, but the variants are detected by the gnomAD group leading to a unique dataset [56]. The **gnomAD-SV** is a sub-database including only SVs from 14,891 human samples. They annotate 6 canonical SV types and 11 complex SV types. They detected twice as many SVs per genome compared to the 1KGP project, highlighting the detection power of high-coverage data. [57]

The Catalogue Of Somatic Mutations In Cancer (**COSMIC**) is a large curated database of mutations from tumor samples. The database is based on published papers and includes clinically relevant information. It contains over a million of CNVs. [58]

Unlike the human variation databases, no database storing SVs in prokaryotes is known at the moment.

1.7 Structural Variations in Bacteria

Although the first gene amplification was observed in the model organism *Escherichia coli K-12* in 1963 [59], the later major effort regarding structural variants and copy number variants was focused on human genomes or generally eukaryotic organisms. However, that was an underestimation of the importance of prokaryotic genome rearrangements as was later discovered. Bacteria are an omnipresent and essential part of nature. There is an estimation of 5×10^{30} bacteria present on the earth. Also, they belong to the most deadly pathogens and multiple issues related to bacterial pathogens emerged, namely growing antimicrobial resistance. [60]

The sequencing of bacterial samples is done in two ways, by cultivating and sequencing **bacterial isolates** or by sequencing **communities**, e.g., **microbiomes**, by shotgun metagenomic sequencing, or by targeted amplicon sequencing. The focus of this thesis lies in the sequencing of bacterial isolates. [61]

Prokaryotic genomes differ in multiple ways from eukaryotic ones. The genome is composed usually of a single double-stranded DNA formed into a circular shape. There can be additional independent circular genomes called plasmids carrying less important though beneficial genes. In some species, e.g. *Shigella*, the plasmids are responsible for virulence [62]. Because of the small size, the bacterial genome is dense. Genes lack introns and are almost next to each other without a significant gap. Some genes are organized in operons, adjacent genes belonging to the same pathway and expressed together.

Most importantly, bacterial genomes are free of large repetitive regions, yet they contain some repetitive elements. These repetitions then serve as a substrate for genome rearrangements. They can also be incorporated through **horizontal gene transfer** (HGT). It is important to mention that there is a negative relationship between genome stability and repetitive sequences. The bacterial genomes are limited to a finite number of genes they can harbor. They dispose of less-worthy genes to balance new gene gain from HGT. This bacterial continuous gene gain and loss makes them adaptable [63]. [64]–[66]

Generally, rearrangements over 50 bp are considered SVs in Bacteria [61]. The role of SVs in the prokaryotic domain is different compared to eukaryotic genomes. Both evolutionary and phenotypic implications are extensively studied. The prokaryotic genomes are stable between subsequent generations (due to binary fission), but on the evolutionary timeline, they are plastic, shaped by HGT, genome rearrangements, prophages (bacteriophages), and **mobile genetic elements** (MGE). These can all participate in genome rearrangements [60], [67]. Furthermore, the mechanisms of SVs creation are similar to those described in Eukaryotic genomes [65].

When a region is excised and recombined in the opposite direction, it is called **inversion**. Inversions in bacteria are often reversible. Two types are observed, site-specific recombination and large chromosomal inversions. In the genome, the inversion function as an on/off switch depending on the direction. If it is rotated oppositely to the transcription locus, the transcription is turned off. Inversions are the main drivers of structural rearrangements in bacteria [62]. [61]

Duplication in bacteria plays a role in their metabolism, e.g. multiple copies of ribosomal RNA genes are often present. The duplicated genes can be inactivated through pseudogene creation or deletion. There are long-term genomic duplications and short-term adaptive amplification. Also called multicopy duplications or gene accordions. They are created quickly and increase the levels of protein translations. It has been shown that gene amplification in bacteria is predominantly stress-induced as a response to starvation or exposure to drug treatment [68]. Adaptive amplification can also be responsible for antibiotic resistance. The exact mechanism of creation is not known but it is expected to be through initial duplication followed by homologous recombination. [61]

Deletions are believed to play a role in the bacteria's specialization and also to provide quick adaptation. However, because they are under negative selection pressure and because the bacterial genomes are already densely packed, large deletions are rare. The same applies to duplications and insertions. [62]

Insertion is very common in bacteria. It includes the acceptance of DNA sequences from the cell outside via conjugation or natural competency, or within the genome. Mobile genetic elements (MGE) are the main driving force. These include transposases (insertion sequences), integrons, prophages, and transposons. Many MGEs contain genes responsible for antibiotic resistance, the production of toxins, etc. Insertion within the gene can lead to gene inactivation. [61]

The symmetrical design of the genome leads to biased **symmetrical structural variations**. Three forces were described as creating this bias. First, the distance of a gene from the replication origin (oriC) is a large force. More important genes were observed to be close to oriC. Second, there is a difference in replication between the leading and lagging strands. Third, the limitation to having symmetrically sized replichores (halves of a circular chromosome) leads to symmetrical inversions. Symmetrical inter-replichore inversions are the most commonly detected SV in bacteria. [65], [69]–[71]

Structural variations in bacteria can change the distance of a gene from **the replication origin** (oriC) which can have an extensive impact [65]. The SVs and CNVs are part of pathogenesis evolution and antibiotic resistance [72].

Structural variations were proved in *Shigella*, a common cause of diarrheal illness, which is also becoming resistant to multiple antibiotics. 34 SVs were found between different isolate pairs. However, the detailed insight showed that the role of these SVs is largely unknown [73] Similarly, SVs were found in Pseudomonas syringae, a cause of the fungal disease of kiwifruit [74]. A large metagenomic study detected SVs present in the human gut microbiome and associated them with host disease risk factors [75].

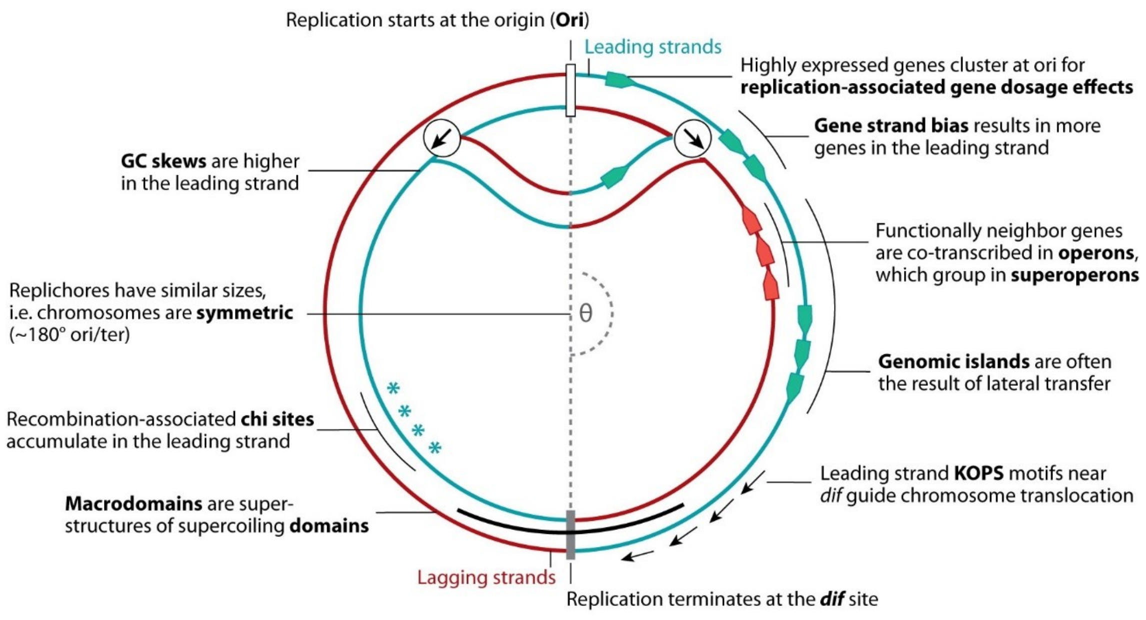

Replication starts at the origin (**Ori**)
Leading strands
Highly expressed genes cluster at ori for **replication-associated gene dosage effects**
Gene strand bias results in more genes in the leading strand
GC skews are higher in the leading strand
Functionally neighbor genes are co-transcribed in **operons**, which group in **superoperons**
Replichores have similar sizes, i.e. chromosomes are **symmetric** (~180° ori/ter)
θ
Genomic islands are often the result of lateral transfer
Recombination-associated **chi sites** accumulate in the leading strand
Leading strand **KOPS** motifs near *dif* guide chromosome translocation
Macrodomains are super-structures of supercoiling **domains**
Lagging strands
Replication terminates at the ***dif*** site

Figure 1.5 – Illustration of replication origin bias, taken from [66]

2 DETECTION OF STRUCTURAL VARIANTS

The following chapter describes the methods of SV detection through time. The earliest methods may be seemed as ancient, but certain algorithms still used nowadays were built upon these methods, e.g., circular binary segmentation. The methods are mentioned chronologically as they were used. The major focus is on next-generation sequencing, which changed the research of DNA by enabling massive and affordable sequencing. The first two subchapters are focused on the principles of these laboratory methods. The following subchapters are then focused on the bioinformatical point of view of SV detection. Firstly, the general principles and terms are described. Later, there is an overview of state-of-the-art detection tools.

2.1 Wet-lab Methods

G-banding, also known as **Giemsa banding**, is a karyotyping method based on unique banding patterns across chromosomes. The chromosomes in the metaphase are digested with trypsin first. Giemsa is composed of methylene blue, azure, and eosin and is specific for the phosphate groups in the DNA strand where it binds through intercalation. Heterochromatic regions, having high AT content are stained darker, while euchromatic, GC-rich and high gene content are lighter. The banded chromosomes are then imaged under a visible light microscope to detect genome rearrangements at a resolution over 3Mbp. It can detect events such as insertions, deletions, and mainly translocations. [76]

Fluorescence in situ hybridization (FISH) uses complementary DNA fragments (15-10kbp) carrying fluorescent labels (fluorophores). After designing complementary probes, the metaphase chromosomes are attached to the slide. The probes are hybridized into complementary sequences and the unattached probes are washed away. The chromosomes are observed under fluorescent microscopes after exciting the fluorescent labels. The FISH can detect the same SVs as G-banding but at a higher resolution of over 100 kbp. It is also more accurate. [76]

Optical mapping is an alternative to sequencing derived from restriction fragment length polymorphism mapping. It is dependent on the nicking restriction enzymes, which cut the DNA strands attached to a slide into fragments (300 kbp to 3 Mbp) at specific positions (creating an optical map of nicks). Each fragment is stained with a fluorescent dye and fragments labels are mapped against the in-silico reference. The missing or additional labels and the distance between them serve to detect structural variations. This can detect deletion (missing label), insertions (extra-label), duplications (repeated labels), translocations (unique nicks), and inversions (reverse nicks). Optical mapping is well-suited for large genomic rearrangements and also useful in repetitive regions of a genome. [51], [76], [77] One of the earliest methods for the detection of CNV is **comparative genomic hybridization** (CGH), originally developed as a method for comparing copy numbers from test and reference samples using fluorescence in situ hybridization. Both samples are differentially fluorescently labeled (by cyanine dyes) and then hybridized competitively to chromosomes. The fluorescence ratio is measured along the chromosome and allowed the detection of regions of loss or gains in the regions of 5-10 Mbp. [78]–[81]

Later, the chromosomes were replaced most recently by **microarrays** of oligonucleotides with known positions within the genome. The measured ratio of the fluorescence intensity between the observed and reference sample at a given location is assumed to be proportional to the relative copy number of that sequence. The equal intensity of both fluorochromes indicates no loss or gain of a segment, while the prevalence of one of the fluorochromes indicates either loss or gain of a segment compared to the reference sample. The array resolution is limited by two aspects: the length of cloned DNA targets and the distances between these targets on the chromosome. [82]–[84]

The major technical challenge of **array-CGH** is generating and measuring hybridization signals. Several aspects influence signal intensity: base composition, amount of repetitive sequences, and amount of DNA available for hybridization. Ideally, the hybridization signals should be linearly proportional to sequence abundance. More accurate results can be achieved by hybridizing a single sample only and then comparing the results with a set of control samples. The complexity of the observed genome has a significant impact on the achieved resolution. Copy numbers from small genomes such as yeast and bacteria are easier to obtain compared with mammalian genomes because each portion of the genome is represented more abundantly. Furthermore, arrays made from BAC clones (~100-200 kbp) provide more intense signals compared to short sequence arrays (cDNAs, PCR products, and oligonucleotides) and can detect single-copy changes. Oppositely, short sequence arrays provide higher resolution but noise levels disable the possibility to detect single-copy changes. [82], [85]

Several algorithms were developed for array-CGH data analysis. The output data are log ratios of normalized intensities from both samples at a given position. The task is to separate regions of low and high log ratios. The initial methods were based on simple smoothing by local average. Based on a false discovery rate the threshold was defined for gains and losses. More thorough algorithms were later applied using the mixture model, Hidden Markov Models, maximum likelihood, regression, and others. [86]

A still commonly used algorithm is **circular binary segmentation** (CBS) [87]. It converted intensity values into regions of the same copy number. A more advanced approach based on Hidden Markov Models was **BreakPtr**, designed for precise breakpoint prediction from high-resolution array-CGH [88].

Similar to array-CGH, the **SNP microarrays** are also based on hybridization and can detect events only in the designed regions. Contrarily, a single sample is hybridized to the microarray, and log-transformed ratios are computed by comparing the fluorescence intensity of every probe with multiple reference hybridization samples. [83]

The original use case is genotyping of the single nucleotide polymorphism (SNP). SNP is a single base change at genome position, mostly with only two different bases (alleles) which both appear significantly in the population. These SNPs are stored in databases, such as dbSNP. The rarer changes are defined as mutations. The target sequences on the microarray are allele-specific oligonucleotides (~15bp) which are specific only for one allele and two probes for both alleles have to be carried by microarray. Since the probes are similar, cross-talk can happen and is usually mitigated by using multiple probes per SNP. [89], [90]

Contrary to array-CGH, the SNP microarrays offer a lower signal-to-noise ratio per probe. On the other hand, the allele-specific oligonucleotides improve CNV detection sensitivity by applying calculated **B allele frequency** (BAF). This metric takes advantage of the fact that two alleles are measured, denoted as A and B. The BAF is then a normalized intensity ratio of both A and B alleles, such that BAF equal to 0 denotes an absence of the B allele (AA or A- genotype), BAF of 1 denotes the absence of the A allele (BB or B- genotype) and BAF of 0.5 denotes the equal presence of both alleles (AB genotype). The BAF can be effectively used to calculate copy numbers from the range 0 to 4 in diploid positions. It also allows the detection of copy-neutral variations such as uniparental disomy, mosaic losses, and gains. [83], [89]

Several methods were developed for CNV detection from SNP microarrays. They are usually designed for specific vendor designs, such as Affymetrix (using differential hybridization on glass surface) or Illumina (using a single-base extension on microbeads). SNP data is usually normalized against a reference population to reduce between-array variation and probe-specific hybridization effects (as cross-talk). However, these procedures assume the same copy number of reference population including positions of known CNV regions. These known side effects lead vendors to design microarrays more suitable for CNV detection. For Illumina arrays, the normalization is done only inside the array. The genotyping is done by clustering for both platforms, where each genotype (AA, BB, AB) represents a cluster in 1D space and the probe is assigned to the genotype by proximity. Both major vendors had their proprietary software for CNV detection. Some other algorithms were also published, such as **QuantiSNP** [91], **PennCNV** [92], and **Birdsuite** as the most commonly used ones. Both applied Hidden Markov Models and assume that observed intensities are related to unobserved copy numbers. The emission distribution is assumed to be Gaussian. It is also assumed that adjacent genomic positions have similar copy number states. Other algorithms have an origin in the array-CGH data analysis, such as circular binary segmentation [87]. The performance evaluation of these algorithms exposes variance in called CNVs across tools and points out the benefits of combining results of multiple algorithms. [90], [93]–[95]

Although the non-sequencing methods might seem to be old-dated, the array-CGH and SNP-microarrays are still commonly used for CNV detection. The main benefits are robust detection of large events, reduced costs compared to sequencing, and processing time.

2.2 Next-generation Sequencing Methods

The massive growth of genomics had not come until the advent of next-generation sequencing (NGS), now mentioned also as second-generation sequencing (SGS). Before this, the common method was the Sanger sequencing by incorporating dideoxynucleotides and following capillary gel electrophoresis. This method is labor-extensive and costly but still considered a gold standard. In the following second generation, four dominant commercial platforms were developed: 454 pyrosequencing in 2005, Illumina/Solexa platform in 2006, SOLiD in 2007, and Ion Torrent in 2010. It is the **Illumina sequencing short-read platform** that became by far the most used sequencing platform till now with a variety of applications. [96]–[99]

The Illumina sequencing device is present in almost all facilities including the Brno University Hospital, with which we cooperated. Also, it provides a large output of data, also mentioned as high throughput sequencing. A sufficient number of sequencing reads are necessary for the read-depth detection method which employs the coverage. Other features of Illumina sequencing are also required such as the reads orientation, length, and insert size distribution. Lastly, there are multitudes of datasets stored in databases such as NCBI Sequence Read Archive (SRA). Thus, we used the Illumina platform as the assumed source of sequencing data.

In Illumina sequencing, **sequencing by synthesis** is in use. After the isolation of DNA, the genomic library is generated by fragmentation and adapters ligation. Size selection can be applied. The adapter consists of a sequence complementary to one of two oligonucleotides on the flow cell, a barcode to identify the sample, and a binding site for primer. The single-stranded fragments are attached to the surface of the flow cell, a glass panel with lanes. The complementary sequence is hybridized by polymerase and the original template is washed away. In the process of bridge amplification (or cluster generation), the hybridized strand is folded over and hybridized to the adjacent flowcell oligonucleotide and polymerase generates a complementary sequence again. The double-stranded bridge is denatured and the whole process is repeated in cycles leading to clusters of approximately 1000 copies of the same sequence [100]. The bridge amplification is needed for a low signal-to-noise ratio. After the bridge amplification is done, the reverse strands are washed away, leaving only forward strands. [101], [102] In the process of sequencing by synthesis, the fluorescently tagged nucleotides are added. The first sequencing primer is hybridized to the forward strand and hybridization is carried from the 5’ to 3’ end. After the incorporation of each nucleotide into the hybridized strand, the clusters are excited by light and the emitted fluorescent signal which is characteristic in wavelength and intensity for a nucleotide is captured. This process repeats and the number of cycles defines the read length. After the cycles are finished, the synthesized read product is washed off. The original strand folds over and creates a bridge. The new strand is hybridized as in the bridge amplification process. The double-stranded bridge is denatured and this time only the reverse strand is kept. The second sequencing primer is hybridized, and the second read product is sequenced.

The details of the process differ based on the sequencing platform, dye chemistry (1,2, or 4 colors), type of flowcell (random or patterned), reads index (single or dual), and library preparation (single-end, paired-end). [103]

Random shotgun sequencing was introduced in 1981 early after the invention of Sanger sequencing [104], [105]. The two strategies for sequencing complete genomes based on the shotgun approach are whole-genome shotgun sequencing and hierarchical shotgun sequencing [104]. Later, an important improvement was introduced as ‘double-barrelled’ shotgun sequencing when both ends of double-stranded clones were sequenced and the information about linking and opposite orientation was used to close the sequencing gaps [106].

In the earlier **hierarchical shotgun sequencing**, the cloned genome is divided sequentially into overlapping segments of known order. These segments are separately sheared into fragments and sequenced [107]. The execution was done firstly by employing yeast artificial chromosomes, later by bacterial artificial chromosome (BAC) with a large insert size (up to 200 kbp) and fingerprinting method [104], [107]. For hierarchical shotgun sequencing, it is necessary to create a molecular and physical map of fragments. In whole genome shotgun sequencing, the cloned genomes are directly sheared into fragments. These unordered fragments are then sequenced again by using cloning vectors [107].

Assembling was easier for the hierarchical shotgun method as apriori information about sequence order is there and was also used in The Human Genome Project. The competing Celere Genomics in their pursuit of a complete human genome employed the whole genome shotgun strategy to complete already published data by the HGP project [104]. This outlined the common future strategy of mapping reads to the already known reference.

The idea of shotgun sequencing started in the first era of sequencing but is still in use. Compared to long reads of Sanger sequencing (approximately 800bp), the next-generation sequencing brought massive parallel sequencing of much shorter reads. The initial read length of the Illumina platform was 35bp, but it raised to 300bp. Illumina platform can utilize three types of reads: single-end, paired-end, and mate-pair.

In **single-end sequencing**, double-stranded DNA fragment has flowcell adapters attached to both ends, but only a single end has sequencing primer ligated. Then sequencing starts from this adapter to the end of the fragment in the 5′-to-3′ direction.

In **paired-end sequencing**, both ends have sequencing primers ligated. Both the read length and the insert size (length of PCR fragment without adapters) are subject to choice. It is the apriori known insert size that gives useful information about the expected position and orientation of the paired read. This information can be used for the detection of structural variants or tagging and removing PCR duplicates. Usually, there is a gap in an unsequenced fragment. However, when the fragment size is smaller than double the read length, there can be an overlap. Both reads then can be merged into large-spanning single reads [108]. In the Illumina platform, paired-end reads have usually short inserts of 200 to 500bp. The graphical description of the terms fragment and insert size is in Figure 2.1. [103]

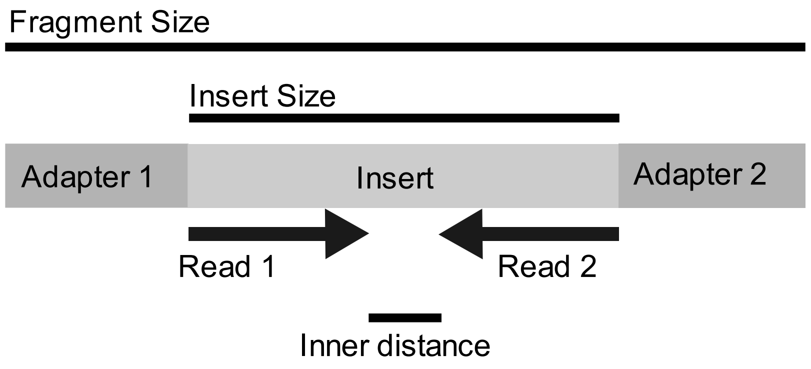


Figure 2.1 – Illustration of insert and fragment size definition

Longer insert sizes in the range of several kilobases can be achieved by applying the mate-pair library preparation method. In the Illumina mate-pair workflow, a biotinylated junction adapter is attached to both ends of fragmented DNA. These tagged fragments are circularized so that both adapters are joined by a biotin junction adapter. Circularized fragments are sheared into smaller fragments, while the fragments carrying the biotin junction adapter are enriched. The fragments without the junction are discarded, however, there is usually some contamination. Illumina adapters are then added to both ends of these fragments and can be sequenced. Compared to the expected forward-to-reverse orientation of paired-end reads, the mate-pairs are in opposite reverse-to-forward orientation as a consequence of circularization, where fragment ends are adjacent. Certain abnormalities of read orientation can occur based on the position of the junction adapter within the fragment. The mate-pair sequencing does not allow reads merging as there is no overlap. [100], [109]

2.3 Detection Approaches for NGS

The next-generation platform still represents the most common way of obtaining genomic data and a plethora of methods and algorithms for the detection of structural variations have been developed for over a decade. Typically, they tend to focus on a particular class of structural variation. Various classifications of detection methods were published - based on a specific feature of NGS data, whether is it a whole-genome or selected targets, i.e., exome sequencing, whether it is a single genome or population approach, whether it searches for somatic or germline variants, whether it is a diploid or haploid genome. However, the most common classification is a division into four methods using features of NGS data. These are read-pair, read-depth, and split-read methods and methods based on de-novo genome assembly. The first three require the data to be aligned to a reference genome, while the assembly approach requires a high-enough coverage to denovo the assembly of the genome. [12], [83]

2.3.1 Read-pair approach

The read-pair approach employs one of the biggest advantages of sequencing – paired-end reads. This approach observes **the position, distance, and orientation of read pairs** in the alignment. The reads which differ from expectations are called '**discordant**'. These discordant reads are mapped closer or further than expected, mapped in inverted orientation, mapped in the incorrect order, or mapped on different chromosomes. The illustration for discordant reads based on the insert size is in Figure 2.2. The majority of SV classes can be detected by this approach. [83], [110]

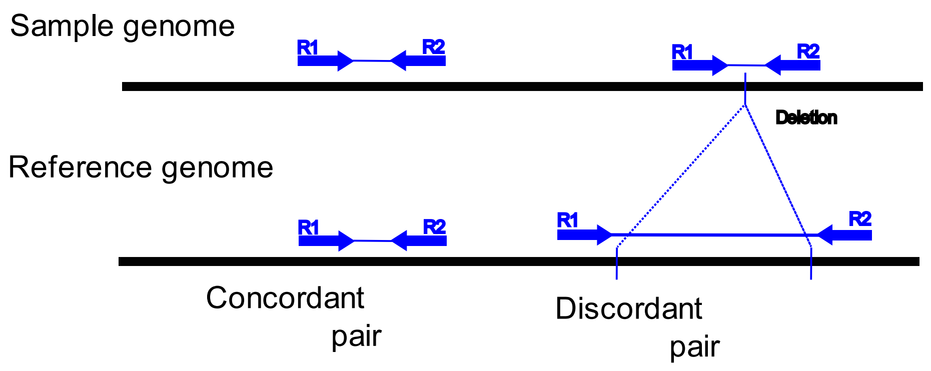


Figure 2.2 – Discordant and concordant reads based on the insert size

Several **signatures** (features of mapped reads) are defined for classes of structural variations. The easiest signatures for detection are basic insertion and deletion. Pair of reads that span over isolated deletion are mapped in the correct orientation of forward-to-reverse, but the insert size between reads is longer than the expected library insert size.

Oppositely if they span over an insertion event, the insert size is decreased from the expected size. The signature of inversion is also clear. Both reads are mapped in the correct order, but the one read spanning over the breakpoint of the inverted region will map with flipped orientation. Two read pairs spanning over both breakpoints of inversion with flipped orientation then form the inversion signature. Expected orientation is usually denoted as +/-, which means the first read is mapped to the forward plus strand and the second mate is mapped to the reverse minus strand. There is a limitation regarding sequencing library parameters. If the inversion is longer than the insert size of the sequencing library, the inversion signature will not be detected. [111]

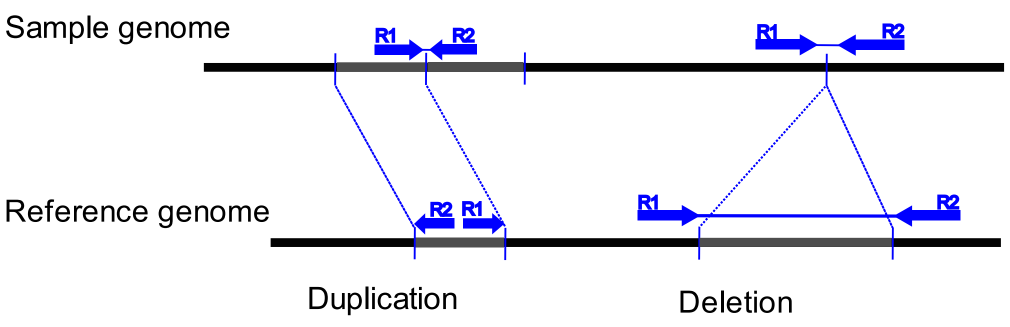


Figure 2.3 – Read-pair signature for deletion and duplication

More **complex linking signatures** can be applied for events located over larger distances. The regions further apart in the reference genome can become closer in the sequenced genome (donor) by genome rearrangement. A mate pair (one of the read pairs) covering the breakpoint in the sequenced genome will map with increased insert size. The two read pairs which are closest to the breakpoint in the donor genome will form a linking signature. The basic deletion is a simple case of this. Cancer fusion genes are another case of linking signature between two remote genes which overlapped in the donor. Insertion which consists of the region presented elsewhere in the genome forms a linked insertion signature. This constitutes of two linking signatures covering two breakpoints of the inserted region. Reads are mapped incorrect orientation but with increased insert size. Linking makes it possible to detect the origin of inserted region contrary to the basic insertion signature. The limiting is the size of the inserted region such that too large regions decrease the probability that the two linking signatures belong together. Tandem duplications of the region are presented in reference forming another linking signature. The mate pair ending in each of the duplications will map in reverse order but with the correct orientation. It is called everted linking signature. It can detect only novel tandem duplications with unchanged copy numbers. Linking signatures can indicate the proximate position of breakpoints, but not the exact one.

If only one read from pair maps to the reference, it forms a hanging insertion signature and we can assume the insertion of a novel genomic region. These hanging reads can be de-novo assembled to discover the insertion region. They are also mentioned somewhere as orphaned reads and can be categorized into the split-read approach. [111]–[114]

The read-pair approach can distinguish between tandem and interspersed duplications. The signatures of reads from tandemly duplicated segments include lower insert size (as reads are mapping closer than expected) and reversed both orientation and order of the reads (upstream read location mapping to the reverse and downstream read location mapping to the forward, i.e. -/+). The interspersed duplication signatures include increased insert size and reads mapping to the opposing strands but with reversed order (+/- and -/+). [115]

Another case of duplication is inverted duplication, which shares signatures with inversion. Contrary, direct duplication (unchanged orientation) shares a signature with deletion. These similarities make the detection challenge. The basic signature for duplication and deletion is in Figure 2.3. [115]

The data distribution of the insert size is expected to be Gaussian [116], [117]. An interesting case represents sequencing with two different insert sizes libraries. This is designed to overcome the limitation of small insert sizes for detecting larger genome rearrangements. [118]

It is necessary to say, that highly accurate detection of various complicated SV types is achieved in combination with the split-read approach as soft-clipping is common in the alignment. [116], [117]

2.3.2 Split-read approach

The split-read approach takes full advantage of mapping properties to the reference genome. It enables single-base resolution. Firstly it was used in the project of human genome indels detection from Sanger sequencing [119]. The signatures are based on an incorrect alignment of mapped reads which is gapped or split. The approach to detect split reads is through **soft clipping**. The soft clip of the read represents a continuous mismatch at the 5’ or 3’ end of the read. The sources of soft clips can be sequencing errors, chimeric reads, reference errors but also structural variants. Another mechanism included in this approach is the **anchor and orphan reads** illustrated in Figure 2.4. This mechanism overlaps with the pair-read approach. [120]

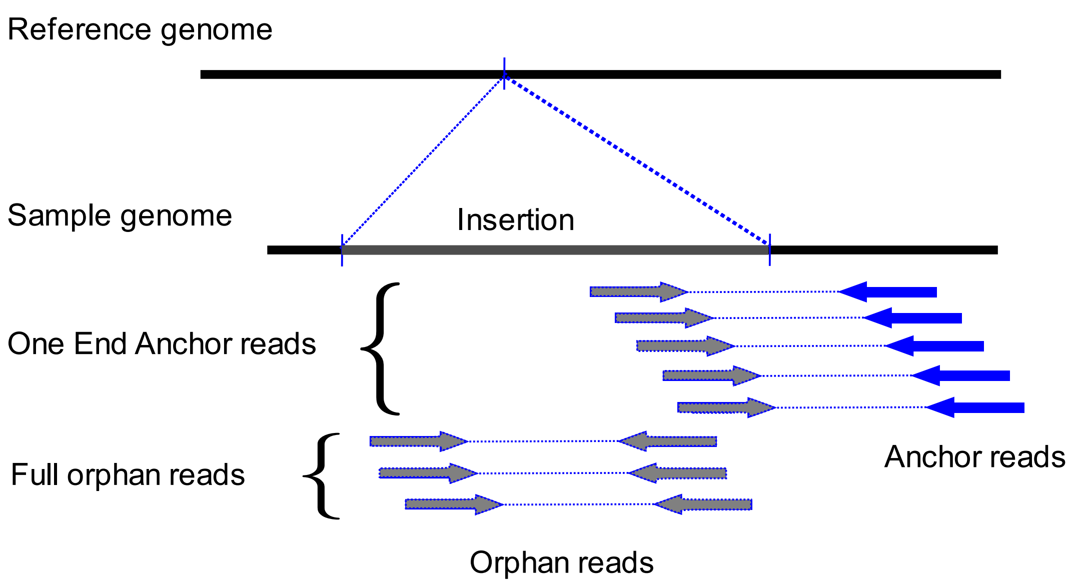


Figure 2.4 – Orphan and anchor reads

Read sequenced over a deletion breakpoint will map with a split mapping signature, where both ends of reads (prefix and suffix) will map to different regions in reference. If its mate pair is uniquely mapped, the split read is masked as a so-called soft-clipped read. This signature is well used by long reads platforms but with short-read data, there can be too much false mapping of read halves. This can be mitigated by limiting the candidate reads or setting conditions. [8], [111]

The decision of what is tagged as soft-clipped and what is tagged as an alignment mismatch depends on the mapping algorithm. The illustration of the soft clipping is in Figure 2.5.

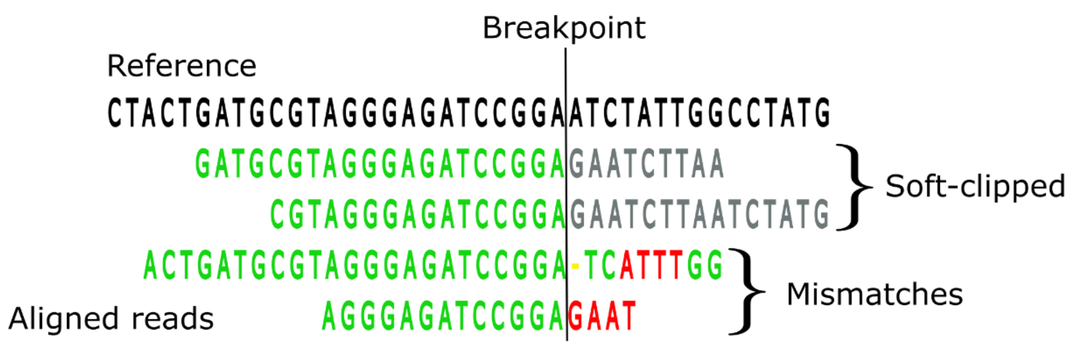

Breakpoint
Reference
CTACTGATGCGTAGGGAGATCCGGAATCTATTGGCCTATG
GATGCGTAGGGAGATCCGGAGAATCTTAA
CGTAGGGAGATCCGGAGAATCTTAATCTATG
Soft-clipped
ACTGATGCGTAGGGAGATCCGGA-TCATTTGG
AGGGAGATCCGGAGAAT
Mismatches
Aligned reads

Figure 2.5 – Soft-clipping illustration

Split mapping insertion signature will form a similar pattern, where short insertion leads to both prefix and suffix mapped very closely but the inserted region will remain unmapped. Both anchored signatures are limited in size for a few bases because the aligners will exclude reads unable to map to the reference by a certain portion of their length. [111]

Under specific circumstances, mobile element insertion (MEI) can be detected [41]. The reads have to be longer than MEI or possibly can be shorter if the breakpoints of MEI are in a unique sequence, the split-read approach can detect the MEI. [83]

The information from the soft-clipped reads can be further used, when these portions are cropped and realigned to the reference genome again, usually within predefine boundaries of breakpoints. This is useful as deletion can be defined when clipped sequences are remapped and they map to the outer side of the deletion breakpoint. If they map reversely, it denotes an inversion signature. If clipped sequences map inside the breakpoints, it is likely a duplication. By observing the position of the remapped soft clip, the tandem, and interspersed duplication can be distinguished too. [83], [121]

Essential for the split-read approach is choose of mapping algorithm and usage of so-called CIGAR strings, where the information about soft-clipping is stored. Among the algorithms that enable soft-clipping are BWA and Bowtie2. [122]–[125]

The most limiting is the computational requirements as the precise search for potential split read mapping is demanding. However, their performance is pleasable even in low coverage conditions. Nice figures of various split-read signatures can be found in [121].

2.3.3 Read-depth approach

The read-depth approach evaluates the **coverage**, i.e. a number of reads that cover a certain position. The terms **read-depth** and coverage are often used interchangeably unless defined specifically. The distribution of coverage is assumed to be Poisson random distribution. In the presence of biases and sequencing errors, the observed coverage distribution differs from the expected Poisson and is wider [126]. The basic hypothesis is that duplicated regions will manifest significantly elevated coverage. Oppositely, the deleted regions will manifest zero or decreased coverage. Thus, only two signatures are created by the read-depth approach illustrated in Figure 2.6. The essential factor for successful detection is appropriate sequencing read-depth because the read-depth approach assumes that read depth is proportional to copy number. The average read counts in regions correlate very well with DNA copy numbers for Illumina and pyrosequencing platforms, while not for SOLID sequencing [127]. [128]–[130]

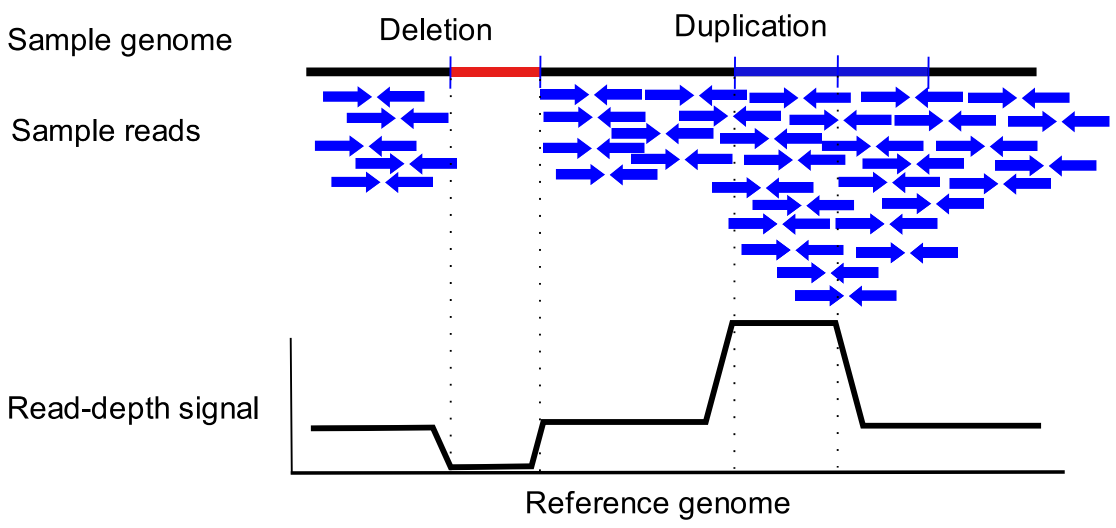


Figure 2.6 – Read-depth signatures and read-depth signal

Generally, the advent of read-depth was easier as the methods applied to CNV detection in array-CGH, e.g. circular binary segmentation, can be used with some modifications on NGS data [127]. However, they are differences: variance in probes is lowest for the normal state (equal copy numbers), and the variance increases for copy number changes. Contrarily, the lowest variance of read-depth is for the deletion state, and it further increases proportionally with increasing copy number [131].

Data processing steps are the following: data preparation (extracting read depth from a pre-filtered set of reads), data normalization (minimizing the influence of sequencing biases), reading read-depth in non-overlapping windows, detection of same copy-number regions (segmentation) and merging them, and estimating the copy-number. [127]

Several studies focused on the topic of genome coverage [132]–[134]. The Lander-Waterman theory used for fingerprinting clones for physical mapping and early shotgun sequencing enabled scientists to estimate parameters such as coverage and the number of gaps as a function of the number of reads [134]. This model assumed a continuous library and was later extended to assume a discontinuous library [133]. An updated model was necessary when paired-end sequencing emerged [132].

The observed values of read depth can be converted into logarithm, log-ratio (for paired or pooled samples), or Z-scores depending on the algorithm [135], [136]. The CNVs can be detected at the visible read-depth level, implicating statistical testing for significant changes from the global average or neighboring regions. Or it can be detected at the read-depth distribution level, observing the significant deviations from the expected parameters of the distribution. [135]

Several **segmentation algorithms** were presented. The earliest, circular binary segmentation (CBS) was developed for array-CGH. Others like mean shift-based, shifting level model, expectation-maximization, and hidden Markov Model can be all used to detect segments sharing the same copy number. [135]

Defining the correct size of the window to aggregate the read depth values can be skipped by focusing only on regions of interest, e.g., gene regions, or by sequencing the designed regions like in whole-exome sequencing. The conceptual segment is then limited to the functional unit. [135], [137]

The results of the read-depth approach are influenced by many external factors. Several **biases** affecting read depth exist, e.g., PCR amplification bias, GC bias, mappability bias, repetitive segments, and artifacts created through multi-mapping reads. Coping with GC and mappability biases is an essential step of the read-depth approach. Multi-mapping reads represent another ambiguity. This phenomenon emerges when a read can be mapped into multiple positions at the same score (the same uniqueness). Managing such a task is fully dependent on the mapping algorithm and knowledge about the algorithm behavior with multi-mapping reads is necessary when applying the read-depth approach. Several scenarios can be applied. First, only uniquely mapped reads are kept, and multi-mapping reads are fully omitted. This leads to the loss of too much information. Second, the multi-mapping reads are positioned randomly. This affects the read depth signal and the following detection. Lastly, the multi-mapping reads are positioned at every possible location simultaneously. The following SV detection requires algorithm design to handle these reads. The difference between multi-mapping reads and uniquely mapped reads can be created by sequencing errors, i.e. otherwise multi-mapping reads are handled as unique ones due to a few mismatches caused by sequencing errors. [11], [131]

2.3.4 De-novo assembly approach

If read length and the amount would be sufficient for a de-novo assembly of the genome, it should be theoretically possible to detect all structural variants including copy numbers, content, and their structure. Such detection would not be based on inference from read signatures but would be directly visible in comparison to the reference, i.e. in self-dot-plot. However, whole-genome sequencing is still costly to perform at parameters that would enable de-novo assembly. Yet, there are several studies where they performed whole-genome sequencing with fosmid clones of multiple samples and detected as many SVs as was technologically possible [7], [138].

The most promising is the application of de-novo assembly in long-read sequencing. Theoretically, all structural variations could be detected by the de-novo assembly. Two main approaches exist – one based on graphs and the second, based on alignments. [139].

Assembly approaches include a whole-genome de-novo assembly and also a local re-assembly to produce contigs which are then compared to the reference genome. The latter is often used in combination with the split-read approach or with the orphaned reads. Thus, the mapping is carried out first. De-novo assembly can be performed over the regions of interest, e.g. regions important for the immune system [140]. This combines the high reliability of the assembly approach with lower costs of sequencing. Generally, the de-novo assembly is limited by required coverage and sequencing costs. The required coverage is about 50× compared to the sufficient 15× coverage required for mapping-based methods. [141], [142]

2.3.5 Signatures

Signatures are features of mapped reads that allow detection from the inferring information. Also, they can be understood as detectable traces that genome rearrangements leave in the alignment. The signatures can be generally divided into three classes: signatures based on pair-end mapping, split-read signatures, and signatures based on the depth of coverage [111].

The usual workflow starts with detecting the signatures and then with calling SVs. The higher sensitivity is achieved by detecting multiple signatures instead of a single one. Sequencing errors can lead to incorrect mapping, while chimeric reads can lead to incorrect information about insert size and orientation. Regarding library insert size, this follows a Gaussian distribution rather than being precise across the whole library [116], [117]. Thus, differentiating between a change of insert size in a signature caused by indel or caused by the tail of insert size distribution creates ambiguity. Read-depth signatures are influenced by sequencing biases caused by a library preparation or sequencing platform [102]. Early use of gain and loss signatures was used in CNV detection based on comparing different species genomes or between healthy and tumor tissue samples. The statistical power of read-depth signature is related directly to the coverage of the whole genome and to the size of CNV. Contrary to pair-end mapping signatures in short-read sequencing platforms, the read-depth signatures can capture very large events with increased confidence as well. However, they are losing at low coverage and short SV events. The breakpoint detection based on read-depth signature is also less accurate. [111]

The signatures supporting the same type of variation can be grouped by clusters for pair-end mapping and split-read signatures or by windowing methods for depth of coverage signatures. The most common clustering method takes into consideration only discordant reads, i.e. those with erroneous mapping. Reads with multiple positions of mapping (multi-mapping reads) are disregarded too, but then the information about repetitive regions is lost. Clusters are then defined at positions where multiple similar signatures occur if the thresholding limit is achieved. Two basic parameters are defined by this strategy: the minimum number of mate pairs to form a cluster and the number of standard deviations to define discordant mate pairs. Both are based on coverage. With increasing coverage, the number of mate pairs and the number of standard deviations can be decreased without harming the specificity. [111], [143] Some methods include multi-mapping reads but they face the decreased sensitivity of too many differing signatures [144]. A rule is defined that multi-mapping mate pairs can be part of a single cluster. A fixed number of standard deviations to define a cluster also represents a challenge in many situations where cluster formation would be prohibited, e.g. short indels. This can be overcome with matching signatures cluster and expected distributions for matches [145]. [111], [143]

The read-depth methods use dividing the reference into windows. Each window must have the same approximate value of coverage but the neighboring windows should have a significant difference from the other windows. Each window is then assumed to represent neutral or copy number change. One of the earliest methods, circular binary segmentation, was initially developed for array-CGH datasets [87]. [111], [143]

Each rearrangement leaves its kind of signature. However, certain signatures are shared between rearrangements. Tandem duplication can create a specific pattern when the second paired read will map before the first read. Or it can be only part of the second read place before and soft-clipped. Contrarily, interspersed duplications can be confused with inversion or deletion patterns (if the duplication is on the same chromosome) or with translocations (if the duplication lies on different chromosomes). [139]

Generally, detecting deletions is easier than duplications. Not all duplications are detectable by read-depth methods. This is likely caused by decreased sensitivity of these methods in distinguishing a copy number increased just a single time. [146]

The combination of read-depth and read-pair approaches could discern between homozygous and heterozygous deletion. For homozygous, where both alleles are missing, reads are mapped farther apart and there is zero coverage. For heterozygous, where only one allele is missing, there is lower coverage compared to baseline, but reads are spanned normally caused they originate from the no-event region. [147]

Signature detection is made difficult cause various reasons. First, sequencing and alignment errors blur the signatures. Contrary to SNV, SV can span over multiple reads which leads to mapping ambiguities. Second, the signatures of multiple SV types can be similar. Tandem duplications and novel insertions may be difficult to separate. Third, genomic rearrangements can overlap or be nested leading to complex SVs. But these complex SV signatures might be overlooked and lead to a preference for simple SVs. [139]

2.3.6 Breakpoints

A breakpoint represents a **novel junction** between the reference genome and a sample. Genome rearrangements can be breakpoint-spanning (discordant pair-end reads) and breakpoint-containing (single split-read). The breakpoint can be understood as two bases, which are next to each other in a sequenced sample but not in the reference, and vice versa. The illustration of deletion-induced breakpoint is in Figure 2.7. [125]

Read-pair and split-read signatures of discordantly mapped reads can be also classified into two classes: direct and indirect cases. Direct case class refers to a situation when both mate pairs indicate the approximate position of two regions linked together in the genome. The breakpoints need to be refined by local assembly or split-read signatures. The direct case approach is used by methods that generally use the clustering approach to detect signatures. Indirect case class refers to a situation when the position of one of the mate pairs is known. The other one is soft-clipped or one-end anchored. [148]

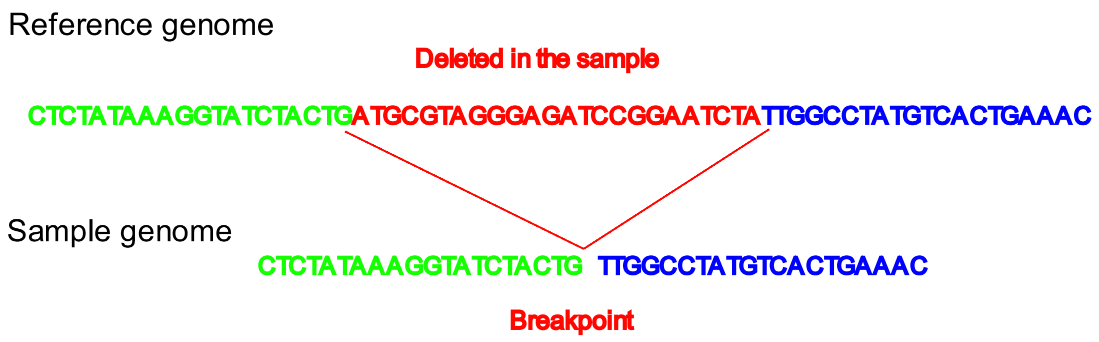

Reference genome
Deleted in the sample
CTCTATAAAGGTATCTACTGATGCGTAGGGAGATCCGGAATCTATTGGCCTATGTCACTGAAAC
Sample genome
CTCTATAAAGGTATCTACTG TTGGCCTATGTCACTGAAAC
Breakpoint

Figure 2.7 – Breakpoint illustration

The detection of exact breakpoints is challenging. The breakpoints are often defined as confidence intervals with the most likely genomic coordinates. Various heuristic clustering strategies are used to assign groups of possible SVs to a single SV event. Close clusters supporting the same SV type are merged into one event. Two clustering strategies are used. First, clustering is based on the distance between the breakpoints. Second, clustering is based on the overlap between the SVs and is predominantly used in population multi-sample studies. Interesting is the use of 2D Euclidean space and geometry (plane sweep algorithm) to deal with the breakpoint overlaps. [149]–[151]

The homology sequences around the breakpoints, which are the substrate for the creation of SV itself, are also complicating the exact breakpoint definition. For such cases, the de-novo assembly approach is useful to uncover the exact genome sequence of the SV. [18]

A special case is exome sequencing. In exome sequencing CNV detection, the breakpoints are not usually detected and whole exomes are considered segments under inspection. Also, only read-depth and split-read methods can be used for exome sequencing. Another special case is restricting breakpoints to be navigated by gene annotation to fit the gene boundaries. In this case, the segments are thought to be gene or intergenic regions. [137], [152], [153]

2.3.7 Copy Numbers

A copy number is an integer that denotes the number of copies of the segment in the sample. This classification can be more challenging if an exact copy number integer is expected, though a majority of the algorithms give only a gain, loss, or copy-neutral classification. Mainly two approaches exist, segmentation and Hidden Markov Model [154]. The basic idea behind copy number estimation through segmentation is that read depth in a given segment should correspond to copy number change. Also, segment coverage values corresponding to the same copy number should cluster together [153]. This process is similar to the Parzen window method for non-parametric density estimation [153]. The HMM-based methods get integer copy numbers from means of their hidden states [155], [156].

The expected linear relation between read-depth and copy-number value was not proven by Pearson correlation for exome sequencing. The normalized read depth by the length of the segment (i.e. exon) was overlapping across the various copy-number states. Thus, the detection of direct copy-number is often impossible. A significantly better correlation was achieved when these ratios were used between the sample and the reference of the known copy number. It was also demonstrated that copy-number prediction is independent of read lengths and mapping algorithms. [157]

The exact copy-number estimation is usually performed by algorithms comparing two or more samples simultaneously. [156], [158]

Also, there is an important difference between CNV detection in diploid and haploid organisms. Haploid organisms (prokaryotes, viruses) have usually a single chromosome, thus, there is only a single copy of a gene. The default copy number is then one. Oppositely, in diploid organisms with paired chromosomes, there are two copies of each gene. The default assumed copy number is then two.

2.3.8 Limitations of using a single approach

It is necessary to mention that NGS methods have difficulties to detect SVS in repetitive regions, thus the detection of microsatellites, transposable elements, heterochromatin, and segmental duplications is challenging. This limitation is not possible to overcome with algorithm design, but rather a combination of other sequencing platforms overcomes this. [159]

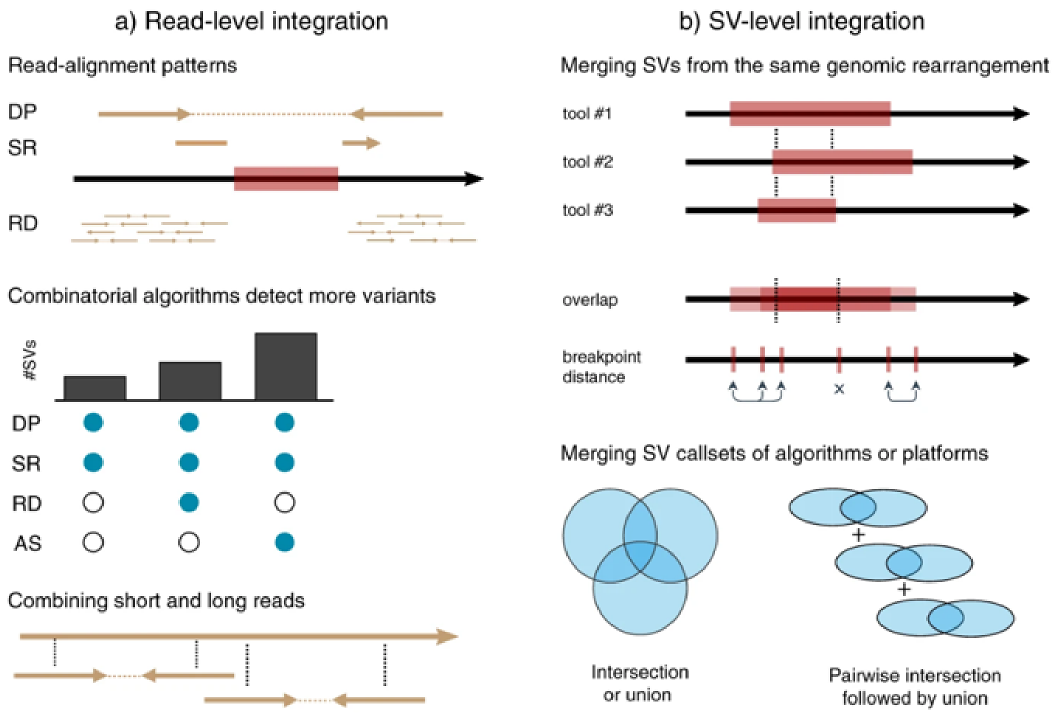
Each detection method itself has limitations. Split-read is the most precise in exact boundaries of SV, but on the other hand, is very limited to the length of the reads and short reads affect accuracy and precisions. Also, it works only in unique regions of the genome. [160]

Read-pair can detect all types of SVs but is not precise in establishing boundaries. The accuracy of read-pair methods depends on the insert size and its distribution. Small

SVs can be skipped in detection with large insert libraries. Similarly to split-read methods, the power is limited in nonunique regions of the genome. [160]

Assembly methods have poor detection power against duplications or repeats and require high coverage. Read-method works well on duplications and can detect the copy numbers as the only method. However, the boundaries resolution is poor. [160]

The limitations of using a single approach are overcome by implementing multiple approaches or even tools together.



a) Read-level integration
Read-alignment patterns
DP
SR
RD
Combinatorial algorithms detect more variants
#SVs
DP
SR
RD
AS
Combining short and long reads
b) SV-level integration
Merging SVs from the same genomic rearrangement
tool #1
tool #2
tool #3
overlap
breakpoint distance
Merging SV callsets of algorithms or platforms
Intersection or union
Pairwise intersection followed by union

Figure 2.8 – Integrating multiple approaches (read-level) or multiple tools (SV-level), taken from [191]

2.3.9 Integrating multiple approaches

Hybrid algorithms were the first to overcome the limitations of distinctive approaches. That is achieved by a combination of more approaches and overlapping their outputs or by increasing the support of SV events by multiple signatures. The breakpoint resolution can be increased by a hybrid approach, which leads to more precise detection of SV boundaries. This is enabled by integrating the split-read approach. The copy number can be calculated by integrating the read-depth approach. The spectrum of detected SVs can also be extended by integrating more approaches. The read-depth method can only detect deletions and duplications and by integrating them with other approaches we can detect a wider spectrum of SV or subtype them. The performance metrics such as sensitivity and specificity can also be improved by a hybrid approach.

Table 2.1 – Overview of methods properties, taken from [141]

Approach	Resolution	Detectable types	Used PE reads
Assembly	base-pair	all	all reads
Split-read	base-pair	all	soft-clipped one anchored split reads
Read-pair	rough	all	discordant reads
Read-depth	very rough	CNVs	all reads

2.4 Overview of Detection Tools

While the principles of detection methods are in the previous subchapter, here I review some of the tools based on these principles. The overview of all further listed tools and their maintenance status and software availability is in **Supplementary Table 1**.

2.4.1 Read-pair tools

Before the arrival of publications describing specific algorithms for SV detection, several studies approached SV detection with prototypes of future algorithms. They focused on the detection of previously unknown SVs and applied various library insert sizes to circumvent the limitations of SV detection.

The earliest studies which used read-pair signatures for SV detection were published already in 2003 [161] and 2005 [52]. In these pioneer studies, they employed genomic libraries of pair-end fosmid sequences with very long insert sizes (called end-sequence profiling - ESP). The same methodology was also used in the later sequencing of eight human genomes which focused on refining the position of discovered SVs and detecting CNVs. They re-assembled the reads with a TIGR assembler. [138]

SVs discovered by previous studies were reassessed by capillary sequencing of fosmid clones with 40 kbp library insert sizes [7]. There is one of the earliest comparisons of SV detected by other studies. While comparing with the 2008 study [138], only partial overlap in detected deletions was found as a possible result of fluctuating clones coverage and sequencing reads quality. They also found an overlap of only 38% with results from another read-depth study [162]. More advanced breakpoint detection was presented. Two regions around breakpoints are extracted from genome assembly and one called junction is extracted from the clone. They are firstly globally aligned and then merged into a single alignment of three sequences. The innermost breakpoint boundaries can be observed from this alignment. [7]

The first NGS study from 2007 used 454 Pyrosequencing of two individuals [112]. The reads were again mapped with MEGABLAST but realigned for higher precision with the Smith-Waterman algorithm. The separation of mate pairs insert sizes was done by thresholding. The cutoff was defined after removing potential outliers and defined for each batch separately. Then, the three standard deviations from the mean (3 kbp) were used as thresholds. At least two paired-end reads were required to establish an SV. The achieved average breakpoint resolution was 644 bp and SVs spanning around 3 kbp and larger were detected. The regions around detected breakpoints were re-sequenced and contigs were assembled. This enabled the examination of breakpoints with other known genome features (segmental duplications, Alu elements, etc). [112]

Two studies from 2008 applied SV detection on the Illumina sequencing data [163], [164]. The first one sequenced two cancer cell lines on the first version of the Illumina GenomeAnalyzer. The authors also performed one of the earliest usages of read-depth data for copy number detection. They observed the number of reads mapped uniquely with correct orientation and insert size within defined size non-overlapping windows. They adapted commonly used circular binary segmentation to estimate copy numbers. [164]

The second study used the methods defined previously [52], [112], [164] such as detecting anomalous insert sizes, incorrect reads orientation, gapped alignment, and read-depth analysis. Both studies only considered a single unique map position.

One of the earliest papers bringing theoretical background to the detection of SV in paired-end data was presented in 2008 by Lee [116]. The author later introduced an algorithm called **MoDIL** for detecting small indels in size between 20-50 bp. It compares the distribution of insert sizes of the whole dataset with local distributions at a given position. The cluster at a given position contains all mate pairs which overlap the given position. If there are no indels, the observed distribution of the cluster will be identical to the distribution of a genome. If there is a homozygous indel, the distribution will be shifted, if there is a heterozygous indel, approximately half of the observed mate pairs distribution will align over a genome distribution and the other half will be shifted. The expected size of indel can be predicted by identifying the parameters of the two distributions (one for each haplotype). The expectation-maximization algorithm is used to find the means of the distributions. It is assumed the insert sizes fit into the Gaussian distribution. [145]

Hormozdiari presented a combinatorial algorithm called **VariationHunter** in 2009 [144]. It was the first algorithm to use reads mapped to multiple positions which were ignored by previous algorithms. Thus, it requires a kind of mapping algorithm which enables multi-mapping, such as mrFAST, SHRIMP, etc. In the first iteration of VariationHunter, they focused on the detection of deletions, insertions, and inversions.

They define concordant and discordant mate pairs, such that concordant reads insert size is in the defined range and their orientation is correct for a given platform, e.g. for Illumina left mate-pair is mapped to the plus strand and right mate-pair to the minus strand. Both reads have to be mapped to the same chromosome. Only discordant reads are taken into account. Each read pair has a set of candidate alignment positions with corresponding insert sizes and orientations. Also, each alignment supports a specific type of SV defined by constraints. The clusters are defined as a set of alignments that support a particular SV at a given position. They define the so-called Maximum Parsimony Structural Variation Problem, which describes the computing of unique alignment for each discordant read pair such that the total number of SVs implied by the alignments is minimized. The modified set cover algorithm is used to solve this problem. The probabilities for each candidate SV are calculated using parameters such as type and length of SV, and the number of reads supporting the given SV. The resulting SVs are then outputted based on the defined cutoff probability threshold. [144]

In **the modified version of VariationHunter** from 2010 [165], the authors implemented support for transposons detection. They defined transposon as part of the genome that is copied to another position with a small dissimilarity (aka copy events). They also assumed the diploid nature of the human genome by adding a conflict resolution of overlapping SVs. The signatures of copy events are more complicated and the authors divided them into two classes – if the transposon is copied in direct or inverted orientation. The modified Maximum Parsimony Structural Variation Problem with added conflict resolution guarantees that no conflicting triplets of SV clusters occur in the final set of SVs. [165]

Korbel et all presented Pair-end Mapper (**PEMer**), an algorithm to detect SVs from all NGS platforms available at the time [166]. For the Illumina platform, the MAQ aligner is used to map reads uniquely and with a mapping score above 20. The mate pairs are then considered outliers if they map with out-of-range insert sizes and discordant orientation. The three-standard deviation rule is used to define the cutoff for insertions and deletions based on the observed insert size distribution. The clusters of reads of defined size are computed in windows separately for long and short events by calculating E and P values. Clusters supporting the same type of SV are merged into a single cluster because clusters are calculated parallelly with different cutoffs and cluster sizes. The authors also created a database of detected breakpoints BreakDB. [166]

BreakDancer published in 2009 detects five types of SVs: deletion, insertion, inversion, and intra- and inter-chromosomal translocations [167]. It also enables pooling multiple samples analysis for population or tumor-normal cancer studies. It consists of two modules – one aimed at five types of SV and the second one aimed at short indels (10-100 bp) usually skipped by the first module. So-called anomalously mapped read pairs (ARPs) are defined as reads mapped with MAQ with a score above 10 and insert size outside the three deviations. The anomalous regions with a statistically elevated number of ARPs compared to the average. An SV is derived from one or more regions if these are interconnected by at least two ARPs. The confidence score is calculated from the Poisson model with the number of ARPs, size of the region, and genome coverage. The most present types of ARPs in the region define the type of SV. Small indels module requires the anomalous regions to have two-sample Kolmogorov-Smirnov test statistics above 2.3 testing for normally mapped reads in the region and the whole genome. [167]

2.4.2 Split-read tools

With the prolongation of the NGS reads length, the split-read methods came to arise. One of the earliest algorithms presented was **Pindel** published in 2009 [113]. It detects breakpoints of large deletions (up to 10 kbp) and insertions (up to 20 bp, restricted by the length of the read). The reads which are analyzed are those which have only one of the mate-pair uniquely mapped. The 3' end of the mapped read serves as an anchor point on the reference genome. From the anchor point, the direction and user-defined distance to restrict the search area for another breakpoint are known. In this area, the Pindel tries to map the unmapped mate-pair dividing it into two (for deletions) or three (for insertions) substrings. Firstly, it uses a pattern growth algorithm to search for unique substrings from the 3' end of the unmapped read within the distance of two insert sizes from the anchor point. Within the distance of the read length plus the defined maximum distance, it searches for the unique substrings from the 5' end of the read. Lastly, it checks if a complete unmapped read can be reconstructed from found unique substrings. This way, the deletions are searched for. The search for insertions is different than the distance area for 5' end substrings is only within the size of the read minus 1 and the read is split into three parts. Every insertion or deletion supported by at least two read pairs is outputted. [113]

Similar to the Pindel is an algorithm called **Splitread** for the detection of indels up to 1 Mbp [168]. It takes multi-mapped reads from mrsFAST (using Hamming distance) into the account and can be used on the exome sequencing. The unmapped reads are decomposed into equally (for balanced splits) or unequally (unbalanced splits) length substrings. For the detection of unique substrings, it uses a seed search approach of balanced splits which narrows the location and length of detected SV. All possible positions of split reads which were within three deviations of the insert size from the anchor point are stored. Clusters are formed from balanced splits. If an unbalanced split supported the balanced one and they have overlapping anchor reads, it is added to a cluster. Split reads can be mapped to multiple clusters supporting different types of SVs. The greedy solution for the weighted set cover problem (similar to the VariationHunter) is applied to find a minimum number of clusters with a maximized cost function, e.g., the number of mappings. [168]

ClipCrop published in 2011 extends the detection abilities of the split-read approach to tandem duplications, translocations, and inversions [121]. The initial set of breakpoints is dug from the CIGAR strings of mapped reads. Only reads with one soft-clipped end are analyzed, reads with both ends soft-clipped are skipped. The authors define L- and R-breakpoint based on which side of the breakpoint is soft-clipped. Breakpoints are clustered within 5-base differences. Soft-clipped fragments over 10 bp are remapped with BWA around the breakpoint within 1000 bp in both directions. The type of SV is inferred from the position of remapped fragments (inside or outside of the breakpoint) and their direction (reverse for inversion). The soft-clipped reads are clustered by SV type and position. Competing SVs are chosen based on a score derived from a number of clipped and anchor reads. [121]

CREST can detect the same types of SVs as ClipCrop and furthermore supports the paired tumor-normal cancer samples [169]. The set of soft-clipped reads is screened breakpoints based on several criteria and are tested for binomial distribution to separate heterozygous events from wild-type reads. The reads fulfilling the conditions serve as a first putative breakpoint. The corresponding breakpoint is detected by repeated mapping and assembly. The distance between alignment to assembled contig and the first breakpoint needs to be within a small distance. For cancer samples, the SV detected in normal samples is skipped from the results. [169]

Another algorithm for SV detection in cancer samples is **Socrates** [120]. It is similar to ClipCrop, but it enables to detection of micro-homologies, untemplated sequences, and gene fusions. The multi-mapped reads are skipped. The soft-clipped reads are realigned. The clusters of anchored reads and corresponding realigned soft clips are formed. Then, the clusters are paired, and those clusters support the SV from both sides. Here, the micro-homologies (1-10 bp identical sequences) can be found on either side of the breakpoints. Also, untemplated sequence (or insertion) can be found between breakpoints of gene fusion. The unmatched clusters are then tested to support already defined clusters, and all soft clips that match another cluster's anchor point with sequence identity over 90% are considered supporting. These pairings can increase sensitivity in exchange for an increase in false positives. [120]

SLOPE is another split-read algorithm that focuses on detecting SV from targeted sequencing [114]. It takes only unmapped sequences, but analyses all reads for single-end sequencing. Then, it carries partial alignment from both ends of the unmapped read and selects the highest-scoring ungapped alignment with up to five mismatches. Only alignments covering more than 30% of reads are kept, however, partial alignment spanning over 90% of reads carries no information and is skipped. All alignments that remained are clustered based on orientation and position in the genome. For all unique reads, the partial alignment is refined by the Smith-Waterman algorithm to allow gaps. The algorithm is called SLOPE cause it plots the slope of median values of breakpoints and their 5' positions. For chimeric junctions, the 5' positions and partial alignment lengths would correlate exactly. Sequencing errors and imperfect alignments will affect the slope. A weighted regression with weights from Smith-Waterman scores prefers longer higher scoring alignments. The clusters of indels and translocations should have a slope of -1 and these are taken as results. [114]

2.4.3 Read-depth tools

The earliest studies which included a read-depth approach were Campbell and Chiang studies on cancer samples [164], [170]. Campbell extracted coverage from uniquely mapped concordant reads and then used modified circular binary segmentation [164].

Chiang applied local change-point analysis using a sliding window and then followed by the merging of events [170].

Generally, the algorithms can be divided into two groups. In the first group, the CNVs are detected by comparing local read count against an average of a genome. The second group consists of algorithms derived from array-CGH methods and CNVs are detected as differences in read counts between two or more samples. [155]

Yoon presented **RDxplorer** in 2009 [131]. There they extracted read depth using non-overlapping windows of size 100 bp. This was justified by an acceptable exchange between detecting small and long CNVs. Also, at the assumed 30X coverage, the distribution of 100bp windows fits the Gaussian normal distribution. This approach is generally shared among many algorithms. The reads are assigned only once at their leftmost position. The GC bias is normalized by comparing coverage values with GC content in a window. For CNV detection, they apply event-wise testing. It detects regions of consecutive windows with a significant change in read depth. For that, the read depth in a window is converted into Z-score (by subtracting the mean of all windows and dividing by the standard deviation). The Z-score is converted into upper- and lower-tail probability. The values in consecutive windows are compared against desired false positive rate divided by the number of all windows of a given size. The extension is carried out iteratively and separately for deletions and insertions. The neighboring clusters of small events (up to 500 bp) are merged. Events with a median of read depth within 0.75 and 1.25 times of dataset mean were omitted. The one-sided Z-test is applied to test significance. [131]

Alkan presented an alternative approach at the level of reads alignment called **mrFAST** [162]. mrFAST is a seed-and-extend method for reads alignment with the main feature of mapping reads into multiple positions. For GC normalization, they applied the local regression LOESS algorithm. Then, they detected regions where at least six out of seven consecutive windows have read depth three standard deviations from the mean. Three deviations correspond to a copy number of 3.5 for the diploid genome. The copy numbers were calculated in 1 kbp windows as a ratio between window read depth and average read depth. The authors focused mainly on large segmental duplications over 20 kbp. The alignment tool is also used in other read-pair algorithms which use multi-mapped reads. [162]

CNV-seq is a method derived from array-CGH and detects CNVs by comparing two samples [158]. The read counts from two samples are detected in a sliding window. They approximate the Poisson distribution of read counts in a window by a Gaussian distribution using the Geary-Hinkley transformation to transform the ratio of read counts into a new variable of approximately Gaussian distribution. Then, they test the probability of these two distributions differing from an equal ratio. The p-values are based on the window size (decreasing with increasing window size) and the number of reads in a window. The theoretical minimal window size giving the best resolution is calculated for the desired ratio. The ratio is also used for thresholding and segmentation. [158]

cn.MOPS brings evaluation of local read count variations by analyzing multiple samples [155]. Similar to other methods, the reads are counted in non-overlapping windows. The reads are GC normalized but also normalized concerning other samples, such that read counts are comparable across samples. The mixture of Poisson models is used, where a separate model is computed at each locus. The model assumes that read counts in a window are distributed across samples according to a mixture of Poisson distributions, where each mixture corresponds to a distinct copy number [155]. The Bayes formula is used to compute the probability that a read count is from a given copy number. This way the cn.MOPS produces integer copy numbers for CNV calls. In the final step, the circular binary segmentation algorithm is used to merge individual calls of the same copy number. [155]

Similar to cn.MOPS is a **method published by Sepúlveda** [171]. They assume the common feature of read depth datasets, the overdispersion of Poisson distribution of read depth values. They assume Poisson distribution for regions of no CNV. Then, they extend this to an overdispersion by either Poisson-Gamma (also negative binomial distribution) or Poisson-Lognormal distributions. The model parameters were done by non-informative prior distributions. E.g., for Possion-Gamma, they used a Gamma prior distribution with parameters α and β. The CNV detection in non-overlapping windows is based on the highest posterior density (HPD) intervals and windows are compared with values of HPD. [171]

CNAseg is an algorithm for CNV detection in tumor-normal samples [172]. It uses a discrete wavelet transformation for smoothing the read counts, which is a useful way to cope with the over-segmentation of HMM method. The read counts are normalized so that both samples have the same read count in windows. The HMM is used for segmentation with Skellam distribution. The k-means clustering is used to approximate the copy numbers, and read counts in every window are clustered between 2 to 7 clusters. The best number of clusters is obtained from the F-statistic (within-cluster sum of squares compared with the between-cluster sum of squares) [172]. The χ^2 statistics is employed for merging the segments where the lowest χ^2 statistic denotes the absence of significant difference between tumor-normal adjacent segment ratios. [172]

Another algorithm using HMM is **JointSLM** [156]. They define a model of the process as the sum of two independent stochastic processes representing read count distribution (approximated to be Gaussian) and white noise (representing alignment errors and coverage fluctuations). The joint distribution of the process has the form of HMM, and it is used to detect CNV in multi-sample windows. [156]

CNVnator focuses on CNV detection in family trios and population sequencing [173]. Alternatively, the authors emphasize the use of reads mapped randomly compared to using only uniquely mapped reads. It employed the mean-shift technique taken from image processing to segmentation. The read depth across the genome can be understood as an image that needs to be processed to define distinct CNV regions. This can be formulated as finding a probability distribution function (PDF) from the read-depth data where this function is an unknown mixture of many distributions corresponding to copy number states. The density maximum in the distribution is the modes of PDF with zero gradients of estimated PDF. In each window, the mean-shift vectors are defined and point to the direction of windows with the most similar read depth. Window breakpoints are detected when two vectors have opposite directions but do not point to each other. The segments are partitioned and merged by an iterative algorithm and thresholding. [173]

ReadDepth is the first algorithm that brought the mappability normalization [174]. It employs modeling a negative-binomial distribution to approximate an overdispersed Poisson distribution of read counts. For segmentation, it uses circular binary segmentation. [174]

CNOGpro overcomes the need for breakpoint detection by applying CNV detection to gene and intergenic regions separately [152]. Furthermore, it focuses on prokaryotic genomes. While counting reads and the GC normalization are done in the usual way of non-overlapping windows, later, the read counts are assigned to individual gene regions defined by a GenBank file. The average of overlapping windows falling into a region is used together with bootstrapping to estimate confidence intervals. The HMM is used to calculate the copy number states in the regions using the Viterbi algorithm. [152]

An alternative to fitting Poisson models is the **Sequana** algorithm [175]. It rather reports all positions that have read depth outside of the overall distribution. The read depth is denoised using a running median. In the ideal case, the read depth fits the Poisson distribution, but in real datasets, the Poisson distribution is a too-narrow cause of overdispersion. The Poisson distribution can be approximated by the Gaussian distribution for large mean parameters. Authors assume for read-depth values higher than 1 the Gaussian distribution. But since this is too restrictive they propose a mixture of models to describe the read-depth underlying nature of central distributions and outliers. They use the expectation-maximization algorithm for the estimation of model parameters. This allows them to define Z-score for every genome position. These Z-scores are then compared with thresholds for duplication and deletion. Double thresholding is used to merge events into larger ones. [175]

CNV-BAC is another prokaryotic-focused algorithm [176]. It was the first to cope with the normalization of the origin of replication bias. The GC normalization is based on BIC-seq2 [177] and CNV detection is based on BIC-seq using the Bayesian information criterion [178]. [176]

2.4.4 Assembly tools

Algorithms depending solely on de-novo assembly are only a few. Furthermore, the distinction between a hybrid algorithm including an assembly method, and a sole assembly algorithm is not absolute.

One of the earliest is **NovelSeq** [179]. It depends on the multi-mapping aligner mrFAST [162]. After the alignment, it separates one-end anchored (only single-end mapped) and orphan reads (reads unable to align under criteria). Orphan reads all de-novo assembled to contigs by EULER-SR or ABySS tools. The contigs are scanned by BLAST for contamination and contigs able to map against the reference are skipped. The one-end anchored reads are clustered into two clusters depending on the strand orientation of the mapped read (+ for forward strand, - for reverse strand). The unmapped reads of clusters are assembled into a single contig for each cluster. The orphan contigs are anchored by merging with one-end anchored contigs by defining it as a maximum-weight bipartite matching problem. Thus, the algorithm enables the detection of novel sequence insertions. [179]

A more complex approach was presented in **Cortex** [180]. It is a de-novo assembler that enables a multisample approach. It also detects structural variants and novel sequence insertions. It employs extended de Bruijn graphs with the coloring of the nodes and edges to represent the distinctive sample in a single graph. [180]

TIGRA [181] depends on previously predicted putative breakpoints and employs de Bruijn graph-based local assembly of reads around the breakpoint boundaries. The assembled contigs then represent alternative alleles. [181]

2.4.5 Hybrid tools

Medvedev was among the first who combined read-pair and read-depth approaches in an algorithm called **CNVer** [182]. It detects discordant read pairs as usual by the three-sigma rule and discordant orientation. It clusters the discordant reads into four clusters regarding the strand and reads the order. Since it focuses on tandem duplications, it deals with sequences already repeated in the reference genome. It maps the reference to itself and iteratively partitions the aligned blocks of at least 100 bp until there are no overlaps of these blocks. The bidirected donor graph is built from the blocks and the clusters. The cost flow function is used to find the optimal walk through the graph and to find the copy counts of the sub-block in the graph. The copy number is based on finding the walks in the graph with higher or lower-scoring functions. [182]

HYDRA [183] combines read-pair and split-read approaches but also employs long reads from capillary sequencing. Discordant reads from Illumina are clustered and targeted as candidate breakpoints. Then, the discordant long reads that overlap with previously called breakpoints are assembled into breakpoint contigs with a phrap aligner. These contigs are aligned to reference with sensitive settings by MEGABLAST and used to identify the breakpoint precisely. [183]

He et all presented a method combining read-pair and read-depth approaches to deal with CNVs in repetitive rich regions [184]. Clusters of discordant reads are used to define the boundaries of a CNV and copy numbers are estimated based on the Possion formula from [111]. The authors present a modified formula to apply in repetitive regions. The branch and bound algorithm is then used to reconstruct CNV in these repetitive regions. [184]

inGAP-sv combines read-pair and read-depth methods [117]. It firstly calculates coverage in 10 bp windows and observes differences to adjacent 1000 bp regions from both sides. It classifies the intervals as homozygous, heterozygous, and uncertain. Later, it employs the read-pair method to classify SVs and assign a quality score. [117]

A novel approach in the time presented **forestSV** [185]. It is a machine-learning approach with a random forest classifier trained on samples from the 1KGP. It can be classified as a hybrid approach because the extracted features consist of both read-depth and pair-read parameters. [185]

GASVpro [186] builds on the previous GASV algorithm [150]. It uses a read-pair approach to detect SVs boundaries and a read-depth approach to decide on hetero/homozygosity. It considers multi-mapping by using the Markov chain Monte Carlo algorithm to sample over the possible alignments. [186]

PRISM [187] combines the read-pair and split-read approach to detect SVs at a base-pair level. It uses the read-pair information to reduce search space for split mapping by the modified Needleman-Wunsch algorithm. [187]

DELLY, one of the commonly used SV callers, also employs read-pair and split-read methods [118]. It also enables a combination of two libraries with different fragment sizes. That usually means a pair-end library with an insert size under 500 bp and a mate-pair library with an insert size over 2000 bp. Discordant reads are clustered using graphs separately for every type of detectable SVs. The nodes of the graph represent the paired ends, edges indicate that both ends support the same SV. The weight of edges denotes the difference between predicted SV sizes. The goal is to traverse the graph so that the distance between reads is greater than a threshold. The discordant reads clusters are considered to contain breakpoints and these regions are used for the split-read method. It first searches for one-end anchored reads within these regions. It can also analyze all other reads for soft-clipping or low-quality read ends alignments. For each read from the region, if the mate read is mapped within two standard deviations of a breakpoint, it is assigned into a set of split reads of a given SV. Instead of dynamic programming alignment, DELLY uses k-mer (k=7) based filtering to identify candidate split-reads and then aligned them to a diagonal. Read-pair method calls are annotated by the number of supporting read pairs and mapping quality. Split-read calls are annotated by the number of split-reads and consensus alignment quality against the reference. Corresponding calls from both methods are merged into the final set of SVs. [118]

A similar approach to DELLY regarding two different insert-sized libraries is used in **PeSV-Fisher** [188]. It uses the read-pair and read-depth methods. Outputs from both methods are merged based on the overlapping. PeSV-Fisher can also work on cancer samples and output only somatic (tumor only present) SV. [188]

Another commonly used SV caller is **LUMPY** [125]. It is unique as it combines read-pair and split-read methods, but optionally accepts a set of called CNVs and/or a set of candidate SVs. It can be understood more as a framework over already established algorithms as the core of LUMPY is based on merging these inputs. As input for the read-pair module is taken an alignment file from BWA or NOVOALIGN. For the split-read module, an alignment file from a split-read mapper such as YAHA, BWA-mem, or BWA-SW is taken. It also enables multi-sample analysis by tracking probability distributions during clustering back to the original samples. The breakpoint is represented as a pair of probability distributions spanning the boundaries of the breakpoint and reflecting whether the selected position represents a real breakpoint. The output of modules is breakpoints boundaries, supporting evidence, and SV type. In the process of clustering and merging breakpoints, the boundaries of a breakpoint are trimmed based on distribution probabilities. [125]

Another algorithm combining three methods is **Pilon** [189]. It employs read-pair, split-read, and local assembly. It is a multipurpose tool that focuses on genome assembly improvement, SNP and indel detection, and also SV detection. It detects SV by detecting discordant reads, soft-clipped reads, and low-coverage regions. These candidate regions are locally re-assembled by the de Bruijn graph. [189]

TARDIS also combines three methods: read-pair, split-read, and read-depth [115], [190]. It builds upon previous works such as mrFAST [162], NovelSeq, and VariationHunter [144], [165]. Signatures of read depth are added to the clusters of discordant reads as additional weights. Both ends of split reads are taken as another case of discordant reads and clustered together. The maximum parsimony approach and set-cover algorithm are used for the solution. [190]

2.4.6 Pipelines

While the hybrid approach combines specific methods, the ensemble or integrative approach combines whole algorithms. It makes use of the fact that there are intersections and unions of the detected SV by multiple algorithmic tools. The algorithm then can assume a simple intersection or union as the best solution or more advanced methods can be used. A pairwise intersection followed by a union is also mentioned [191]. The coordinates of the breakpoints can be averaged across the overlapping region. The decision trees or neural networks were used to decide on the set of true positives for SNP variants [192], [193]. Usually, the union of SV calls and thresholding by a number of callers that called the SV is applied. The earliest mentions of a combination of multiple algorithms were implemented in population studies. [141]

There are several approaches to merging SV calls from multiple callers. Based on VCF files, the BCFtools [124], **SURVIVOR** [194], or **SVDB** [195] package can be used. For BED files, the BEDtools[196] can be used. These tools represent the most simplistic approach.

The biggest study by a number of SV callers used is part of the 1000 Genomes project [41]. The authors used 19 tools based on all approaches and merged their outputs based on the overlap and also used TIGRA [181] for breakpoint discovery. The study is from 2011 and multiple methods have been published so far.

The first independent algorithm published was **SVmerge** [147]. It employs BreakDancer, Pindel, RDXplorer, and two in-house developed tools to detect many SV classes. The initial results of callers are filtered for low-quality SVs and separated by SV type. The merging is done by BEDTools [196] and criteria are defined for an overlap: length (75 bp) and size of remaining parts of SV. All reads and unmapped read-pairs within 1 kbp of SV boundaries are re-assembled and contigs are remapped to the reference with aligner Exonerate. The coverage is calculated by SAMtools[124] pileup. The SVmerge focuses on deletions and distinguishes between homozygous and heterozygous deletion by applying the read depth check. [147]

HugeSeq [197] integrates GATK UnifiedGenotyper [198] and SAMtools [124] for SNP and indel calling, and BreakDancer, Pindel, CNVnator, and BreakSeq for SV calling. The final set is obtained by intersection with BEDTools [196] and SVs called by two and more callers are tagged as high confidence. [197]

iSVP [199] calls deletions based on BreakDancer, Pindel, and GATK Haplotype Caller. The SV calls are categorized by size to achieve precision over 90%, which is fulfilled for SVs that are longer than 100 bp. The calls are merged by BEDTools. [199]

intansv is an R package, developed back in 2014, which integrates seven SV callers, annotation, and visualization inside the R [200]. The callers are BreakDancer, CNVnator, DELLY, Pindel, Svseq2, LUMPY, SoftSearch [201] and others can be added. The findOverlaps function of GenomicRanges is used to merge overlapping SVs. Combined SVs are clustered based on distances between SVs. A cluster of SVs from two or more methods are merged and boundaries are set as a mean of start and end coordinates. [200]

MetaSV carries out both intra- and inter-tool merging of detected SVs [202]. Intra-tool merging suppresses potential duplicated calls which are overlapping. Inter-tool then merges SVs to get unique calls with preference given to tools known to be precise. To detect the breakpoints of SVs, the MetaSV performs local re-assembly with SPADES [203] and contigs are aligned to reference with AGE [204]. The results are annotated and genotyped.

FusorSV uses a data mining approach and fusion model to train and detection of SVs [205]. First, it evaluates the performance of various SV callers on the ground truth dataset of SVs, which are partitioned by type and size. Contrary to simple consensus it assumes the smallest set of callers can be selected based on mutual exclusion, e.g. an SV detected by two callers employing the same approach does not guarantee higher certainty. The pair-wise distance matrix is built from truth dataset results of various SV types and sizes. The Jaccard index is used to find similarly performing callers, which have a lower weight assigned to prefer mutually exclusive callers. The fusion model is used for discovery from eight SV callers. [205]

sv-callers is a Snakemake [206] workflow based on four SV callers [207]. It focuses on human genomes and enables both germline and somatic SV detection. It employs the SURVIVOR (StructURal Variant majorIty VOte) [194] tool for merging SV calls with filtering out low-quality calls with BCFtools [124].

A similar approach based on merging with the SURVIVOR [194] package is employed in **Parliament2** [208]. The final SV calls are additionally annotated with SVtyper [209].

Viola focuses primarily on SV signatures, but in the process, it merges VCF files from four SV callers [210]. Their in-house script first merges the SV calls located within 100 bp (proximity-based merging). Calls called only by single callers are deleted. Then SV calls were merged based on confidence intervals shared by genomic coordinates. [210]

2.4.7 Integrating sample cohorts

During the 1KGP project, the Structural Variation Analysis Group was among the first to employ multiple tools and merge their results. More than that, they had to integrate the structural variations from the hundreds of samples on the population scale. [43]

They were other tools presented that participated in the analysis of 1KGP data, such as **GenomeSTRip** [211]. Another example can be tools for detecting SVs in so-called mother–father–child trios – a combination of father, mother, and child genome sequencing, which is used to detect de-novo mutations associated with rare diseases. Such a tool is **CommonLAW** [212].

In phase 3 of the 1KGP, they sequenced 2,504 human genomes. The reads were mapped by both BWA [122] and mrsFAST [213]. Then, they used a combination of 9 tools for SV detection (BreakDancer, Delly, VariationHunter, CNVnator, Read-Depth, Genome STRiP, Pindel, MELT, Dinumt) based on various approaches and GenomeStrip [211] and other filtering and overlapping to get a final set of 68,818 SVs. [43]

The phase 1 dataset of 1KGP (1,092 samples) was analyzed with a focus on deletions by **BreakSeq2**, an iteration of the previous version [214], [215]. The authors used a combination of 5 CNV detection tools (CNVnator, Delly, Genome STRiP, Pindel, and BreakDancer). The TIGRA-SV was used to reassemble contigs spanning breakpoints, and AGE [204] was used to align contigs to the deleted regions. Multiple filtering was performed to further refine the breakpoints. The main idea of BreakSeq is based on mapping reads to the breakpoint sequence junctions, and was extended by breakpoint genotyping in BreakSeq2. [214]

2.5 Data Preparation for SV detection

After the sequencing, several steps are done before SV detection. The first step quality check of reads followed by trimming. This includes cutting off both sequencing adapters and low-quality bases, which are caused by sequencing errors. [216]

The entrance gate into a wide variety of genomic analyses is **mapping reads** to a reference genome or **de-novo genome assembly**. Matching DNA sequences to a genome is a case of a string-matching problem [217]. Because of both sequencing errors and the true variation of an individual sample, the matching algorithm needs to deal with exact and approximate string matching issues [218], [219]. The initial hashing-based methods were later replaced by methods using the Burrows-Wheeler transformation and Ferragina-Manzini Index [218]. The most commonly used ones are BWA [122], [220], [221], and Bowtie [123]. Both can perform partial alignment called soft-clipping. For specialized detection of rearrangements, other aligners may be useful. This can include aligners dealing specially with multi-mapping reads [162], [213].

Removing the PCR duplicates, i.e., multiple reads originating from a single template, is a commonly performed task. However, this should be used with caution as methods for removing PCR duplicates have difficulties differentiating from natural read duplicates and this step also influences the read-depth, used in the read-depth approach of SV detection. [216], [222]

Whole-genome assembly is a much more complicated task than mapping. Several strategies using Overlap–layout–consensus or De Bruijn graph are used to create continuous stretches of the genome called contigs from overlapping reads. For Illumina NGS, Velvet, and SOAPdenovo are commonly used. [223]–[227]

For the in-silico testing, it is convenient to use artificially created reads. This allows the insertion of various genome rearrangements at predefined positions in the genome. Art and pIRS are such tools simulating Illumina reads. [228], [229]

The files used as the output of structural variation detection tools vary and there is currently no file defined exactly to store SV information. Commonly used is the **BED (browser extensible data)** file [196]. It is a tabular separated file with three required columns: name of a chromosome, start position, and stop position. It is imperative to mention that coordinates are 0-based, instead of the usual 1-based system. There can be nine additional columns, which are more or less defined freely by each tool. Similar to the BED file is BedGraph, with four columns and a required header.

Also, the **VCF (variant call format)** files are used [230], but they are designed with SNV and short indels in mind. The VCF has multiple header lines and multiple columns with content optionally defined in the header. The file is tabularly separated. If VCF is used for reporting SVs, the readability is lowered and while the start coordinate is stored in the POS column, the end coordinate must be placed into a long string in the INFO column together with other information. Also, various SV tools create VCF files based on their own and they are not standardized. [139]

3 OBJECTIVES OF THE THESIS

The purpose of the thesis is to bring novel multidisciplinary approaches to bacterial genome analysis of copy number variations. CNVs play an important role in bacteria in processes of antibiotic resistance, bacteria adaptation, evolution, and specialization. The issue of bacteria drug resistance is present and emerging. Thus, there is a serious need to develop tools aimed at the detection of bacterial CNVs.

Large structural variations are rare in bacteria because of their small and densely packed genomes. Thus, the detection of small CNVs is more important. Special features of bacterial genomes should be taken into consideration and could theoretically improve performance. Multiple bacterial genomes are not annotated. Therefore, the developed method should rely merely on sequencing reads, and a reference.

Developing a standalone method for CNV detection in bacteria is the first objective. Incorporating this method into a pipeline is the second objective. The sub-objectives were set as follows:

1. Develop a novel method for CNV detection (CNproScan)

1.1 Using signal-based computational methods
1.2 Not requiring apriori known genome annotation
1.3 Targeting bacterial genomes
1.4 Statistically evaluated and tested

2. Develop a CNV detection pipeline (ProcaryaSV)

2.1 Targeting bacterial genomes
2.2 Implementing an efficient merging algorithm
2.3 Statistically evaluated and tested
2.4 Enabling the reproducibility and scalability

4 THEORETICAL BACKGROUND

The following chapter describes the theory used in the practical part of the thesis in a more detailed way. Implementing this theory earlier would make previous chapters less coherent because not all approaches would be given the same level of detail.

In the practical part of the thesis, I work with two approaches – read-depth and read-pair. The combination of these two approaches enables to distinguish between the two types of duplications, the tandem and interspersed duplications, and their reverse types. Thus, these two are described more deeply.

Furthermore, I approach the topic of CNV detection from the position of outliers detection. Hence, I describe the theory of outliers and also the related coverage theory. The coverage theory lays the theoretical assumptions about the data distribution of sequencing reads along genomic coordinates.

In the last subchapter, I describe the detailed aspects of the read-pair approach, namely the signatures used to distinguish between tandem and interspersed duplications.

4.1 Read-depth Approach

As mentioned previously, the read-depth approach uses information about the coverage (or read-depth profile) over genomic coordinates. The changes in the coverage point to genomic loss or gain, i.e. CNVs. However, this read-depth profile is skewed by various biases. Therefore, I describe here a few methods of mitigating these biases.

4.1.1 GC Bias

The best-described is bias related to GC-rich and GC-poor regions called GC-bias (or coverage bias), which manifests as deviation from the uniform distribution reads across the genome. It is necessary to cope with GC and other biases when applying methods depending on the coverage, especially the read-depth detection method using a global coverage signal without any reference. However, other detection methods are harmed by low coverage too. Dealing with coverage bias is not always easy as GC-rich regions are placed heterogeneously throughout a genome and they are also correlated with the region function. [231]–[233]

Several technological biases occur at the Illumina platform. Illumina sequencing consists of three steps: library preparation, cluster amplification, and sequencing by synthesis and bias can be introduced at every step. The Illumina library preparation requires the multiplication of genomic material and that includes the PCR, which is a primary source of the under-representation of regions with extremely high or low GC content [160], [231], [233], [234]. This is likely caused by the lower melting temperatures for AT-rich regions and thus their higher fragmentation compared to GC-rich regions

[102]. Considerable biases affecting GC-rich regions are also induced later in the routine. It is known that high cluster densities on the flow cell suppress the GC-rich reads. Also, DNA polymerase plays a role in cluster amplification and sequencing-by-synthesis process. [234], [235]

Firstly, it was observed that the coverage is decreased in GC-poor regions [102]. That was because of the relative rarity of GC-rich regions. Only later it was discovered that the coverage has a unimodal relationship with GC content [146]. It has been described that it is the GC content of the whole fragment, not only the reads, that causes the coverage bias [233]. Also, the GC effect is non-linear and two paired samples can have different GC curves. Thus, coping with GC bias by comparing two samples should be considered. Two approaches are commonly used: binning and smoothing (used in ReadDepth, CNVnator, and others).

The smoothing method is realized commonly by LOESS (locally estimated scatterplot smoothing) algorithm. The table of means of the number of reads belonging to the windows with a given GC content percentage is calculated. The LOESS then removes the local extremes caused by GC bias. [236]

The binning method for single-sample correction is most commonly used in a form defined by Yoon [131]. A similar table to the smoothing method is calculated for every GC content percentage (0,1, 2,…100 %) filled with observed coverage values binned into windows of a selected size. The formula is defined to correct the read count in the i-th window RC_i

$$\overline{RC_\iota} = RC_i \cdot \frac{m}{m_{iGC}}, \tag{4.1}$$

where m is the overall median of coverage of all windows, m_{iGC} is the median coverage value of the same GC content as RC_i. The obligatory part is to define window size and the number of GC content percentages. Selection of window size is chosen accordingly to the following analysis, but it should be longer than the fragment length to produce relevant results. [233]

4.1.2 Mappability bias

Another important source of bias is the mappability bias. This bias is not directly sequencing-based but originates from the presence of repetitive and other low-complexity regions in the genome. The short reads then fall into these repetitive regions without sufficient accuracy leading to the multi-mapping reads problem. It demonstrates itself at two levels – position selection of multi-mapping reads and the following unevenness of coverage. This has a major influence on the coverage signal and downstream analysis similar to coverage bias. The multi-mapping problem is usually solved at the level of reads alignment and is important to know how the alignment algorithm deals with this phenomenon. Several scenarios can happen: multi-mapping reads can be completely ignored, only one alignment position is reported, all alignment positions are reported or some more advanced method is applied. Other solutions are at the sequencing level by employing longer reads or by sequencing longer fragments and thus creating larger insert sizes. Although there is a procedure to deal with the multi-mapping problem by aligner, it is still useful to deal with it at the coverage signal level by applying mappability data. [160], [232]

The mappability bias is solved after the GC bias [237]. Multiple methods have been developed. ReadDepth normalizes mappability by multiplying coverage in bins by the inverse of the mappability in that bin [174]. CNAseg employs discrete wavelet transform to smooth coverage values in regions of low mappability [172]. Another approach is defined by Magi [127] similar to the previous GC bias normalization formula defined as

$$\overline{RC_i} = RC_i \cdot \frac{m}{m_{iMAP}}, \tag{4.2}$$

where m_{iMAP} is the median coverage of windows with the same mappability score as the RC_i window and m is the median coverage of all windows. The key element is the mappability score.

4.1.3 Replication origin bias

A specialty of prokaryotes is their circular genomes and their role in the bacterial cell cycle. In the middle stage of the cycle (C period) the chromosome is replicated before the D period when the cell is divided. DNA replication begins at the origin of replication called *oriC*. *oriC* consists of several conserved repetitions recognized by the DnaA protein and a high AT-rich region (DUE). Then the replication goes both ways around the circumference and it is terminated at the region opposite to *oriC* called *ter*. The majority of bacteria have a single *oriC*, however, there are some with more than a single *oriC*. [238], [239]

During the process of DNA isolation, there are millions of cells in various stages and the sequencing covers all their genomic material. As the replication starts at *oriC* and proceeds further, there is more genomic DNA closer to the *oriC*, that has already been replicated at the moment of DNA isolation. This has an impact on the read-depth profile. [176]

Wu proposed the method to normalize replication origin bias based on smoothing using generalized additive models. They use Gaussian or Poisson model to model the dependence of read-depth in bins on the distance to the replication origin. [176]

4.1.4 Coverage Theory

The general definition of coverage could be stated as: “The theoretical coverage is the average number of times that each nucleotide is expected to be sequenced given a certain number of reads with specified length under the assumption that reads are randomly distributed across the genome” [232]. Also, the coverage can be understood as a redundancy of information to suppress sequencing errors. [232]

The earliest theories regarding coverage were laid back at the beginning of sequencing methods when scientists were interested in calculating how many clones are necessary to have a certain amount of overlaps. The earliest model is called Clarke and Carbon formula (1976) and was later embodied into the Lander-Waterman theory (1988). The coverage was defined as:

$$C = L \times N/G, \tag{4.3}$$

where L is the length of clone insert in bp, N is the number of clones and G is the length of the haploid genome [134]. Importantly, the G can also be understood as the size of the genomic window or bin.

This theory was later extended to filtered DNA sequencing libraries and took into consideration discontinuities and position-based sampling biases [133]. However, a model published a year later was thought for double-strand sequencing. The equation of the expected number of bases that are covered by at least one read has a form:

$$E\langle C\rangle = \gamma - \sum_{x=1}^{\gamma}\left[\left(1-\frac{f(x)}{\pi}\right)\left(1-\frac{r(x)}{\pi-f(x)}\right)\right]^{n}, \tag{4.4}$$

where γ is the genomic target (genome length), $\pi = \lambda - \tau + 1$ is the number of possible insert placements (λ reads length, τ insert size), n number of inserts, f(x) and r(x) are the number of ways forward and reverse reads, respectively, can cover a random position x. [132]

However, for large genomes γ, the expected value of coverage converges to the Clarke and Carbon formula:

$$\frac{E\langle C\rangle}{\gamma} \sim 1 - \left[1-\frac{\lambda}{\gamma}\right]^{2n} \sim 1 - exp\left(-\frac{2n\lambda}{\gamma}\right), \tag{4.5}$$

with n number of inserts and λ reads length. [132]

The general assumption is that reads are randomly and independently sampled with the same probability everywhere across the genome [131]. Under this assumption, it is usually stated that the number of reads mapped into the genomic region follows the Poisson distribution *Pois (λ)*

$$f(x \mid \lambda) = \frac{\lambda^{x}e^{-\lambda}}{x!}, \tag{4.6}$$

with x as the number of reads covering a given site and $\lambda = LN/G$ from Lander-Waterman theory [131]. However, there was overdispersion observed and reported by Yoon [131] and Bentley [163]. The overdispersion is the ratio between mean μ and variance σ^2 and denotes that variance is greater than assumed by the model [171]. The reasons behind this overdispersed Poisson distribution might be the existence of copy number variations, the GC bias (correlation between coverage and GC content), and the mappability bias (correlation between coverage and region mappability) [127]. Removing the CNV regions reduces the index of dispersion for real samples [127]. Alternatively, the overdispersion can originate from another distribution with different parameters.

Often, the Poisson parameter is modeled by a gamma distribution (*Gamma(α, β))* resulting in negative binomial distribution with two parameters *NB(r, p)*. Or, log-normal distribution can be used to model overdispersion (*log-normal(μ, σ)*).

Looking at the whole genome context, we could employ the central limit theorem [131]. The central limit theorem defines a sampling distribution of the mean, based on random samples from a population with a mean μ and variance σ^2, with new parameters $\mu_S = \mu$ and $\sigma_S = \sigma/\sqrt{n}$, where *n* is the number of samples. The sampling distribution then follows the Normal distribution (μ_S, σ_S):

$$f(x \mid \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{(x-\mu)^2}{2\sigma^2}}. \tag{4.7}$$

The central limit theorem validity depends on the sample size *n* and the distribution is normal when the *n* is large enough. The commonly stated size of *n* is 30. Other conditions are that samples are independent and randomly distributed and that the population has a finite variance. The theorem is useful because implies the applicability of tools requiring normal distribution to data under other distributions if certain conditions are fulfilled. For Poisson, it means that *Pois (λ)* with large λ will be approximately normal with both mean and variance equal to λ.

The central limit theorem approximates the cumulative distribution function of Poisson distribution with normal with this formula

$$F_\lambda(x) \approx \phi\big((x + 0.5 - \lambda)/\sqrt{\lambda}\big), \tag{4.8}$$

where ϕ is the CDF of the standard Normal distribution N(μ=0,σ=1) and 0.5 as continuity correction.

More accurate is the non-linear Wilson-Hilferty approximation of argument *x* [240] defined as

$$F_\lambda(x) \approx \phi(c - \mu/\sigma), \tag{4.9}$$

with variables defined hereafter:

$$c = (\lambda/(1 + x))^{1/3}, \tag{4.10}$$

$$\mu = 1 - 1/(9x + 9), \tag{4.11}$$

$$\sigma = 1/3\sqrt{(1 + x)}, \tag{4.12}$$

where λ is the Poisson distribution parameter and x the argument.

Concluding the theory, various tools differ in the view of distribution nature. The Gaussian distribution is used in multiple CNV detection tools, e.g. Magi [127], Sequana [175], and JointSLM [156]. Some employ the Poisson distribution and others multiple models of the mixed distributions (e.g. negative-binomial, beta-binomial). The idea of approximating coverage with normal distribution is embodied in the window approach to read-depth, which divides the genome into distinctive regions. Multiple authors based their algorithms on the assumption that in a window of 100bp with coverage at 30×, the distribution of read counts is well approximated by normal distribution [131], [156].

4.1.5 Outliers Detection

The initial approach to my CNV detection was based on the MATLAB peaks detection function [241]. Later, I approached the CNV detection from the assumption that the CNVs are outliers in the read depth signal. Thus, given the theoretical assumptions, I approached the CNV detection problem as the one-class classification, more specifically as the outlier detection.

The outlier is "an observation which appears to be inconsistent with the remainder of the set of data"[242]. Alternatively, the outlier is "an observation which deviates so much from other observations as to arouse suspicions that it was generated by a different mechanism" [243].

There are two most popular models of outliers generation. The first one is a slippage model. This model assumes that a random sample of *n* observations is mixed with some portion of *r* observations from a different distribution. This further divides into the location-shift model, where r observations originate from distribution $N(\mu+a,\sigma^2)$, and the scale-shift model, where they originate from distribution $N(\mu,b\sigma^2)$. Thus, this model can generate up to *r* true outliers. The second model is a mixture model, where original observations come from distribution G_1, and the outliers come from G_2. All observations then come from a mixture distribution defined as

$$(1-p)G_1 + pG_2, \tag{4.13}$$

where *p* is a constant in the range 0-1. In the mixture model, the number of outliers from G_2 is a random variable depending on probability *p*. If $p=0$ there are no outliers, and vice versa. [242]

Regarding outliers, we can talk about the labeling of potential outliers and outliers identification, to prove if the assumed outliers are real outliers. The most used graphical tool to label outliers is a boxplot. In a large dataset, three-quarters of observations should lie in between Q_1 and Q_3 quartiles and the remaining two quarters under and above Q_1 and Q_3. The central line is often a median, a more robust measure against outliers compared to the mean. The fences are $1.5(Q_3\text{-}Q_1)$ above and under Q_3 and Q_1. The difference between $Q_3\text{-}Q_1$ is called the interquartile range (IQR). Observations beyond these fences are referred to as outliers, depending on the approach. Similarly, the histogram can be used to observe possible outliers. [242], [244]

Z-score is a common way to screen observations for outliers. If *X* is normally distributed as $N(\mu,\sigma^2)$, then $Z=(X-\mu)/\sigma$ is distributed as $N(0,1)$. A popular rule is to label observations with a Z-score higher than 3 as outliers. However, it has been proven that the absolute value of the Z-score from *n* observations is at most $(n-1)/\sqrt{n}$. [242]

This limitation is overcome in modified Z-scores defined as M and using the median absolute deviation defined as MAD:

$$M_i = \frac{0.6745(x_i - \tilde{x})}{MAD}, \tag{4.14}$$

$$MAD = median(|x_i - \tilde{x}|), \tag{4.15}$$

, where the sample mean is replaced by the sample median $\tilde{x}$. The MAD is a variation of average absolute deviation and is even more robust against extremes in the tails. The constant 0.6745 is necessary because $E(MAD) = 0.6745\sigma$ for large n. The observations with the absolute value of modified Z-score M>3.5 are labeled as outliers. The modified Z-score is a robust measure to identify obvious outliers. [242], [244]

In the past, the Z-score statistic was used from the earliest use of the coverage histogram [245] or later in Sequana [175].

Regarding the statistical test for outliers, there are multiple approaches. One tests for a single outlier and then iterates with reduced observation space until the test statistic is not significant. Another approach guesses the number and location of potential outliers *r* and then test if these are true outliers. A more effective approach assumes the number of candidate outliers *r* and then identifies the true outlier in this subset.

To test for exactly one outlier, the extreme studentized deviate (ESD) statistical test is suitable. It is also known as the Grubbs test or maximum normalized residual test. The x_j is identified as an outlier if the value T_s defined in the equation is higher than a formula

$$T_S = \max_i\{|x_i - \bar{x}|/sd \mid i = 1, \dots, n\}, \tag{4.16}$$

$$T_S > \frac{n-1}{\sqrt{n}} \sqrt{\frac{\left(t_{\alpha/2n,n-2}\right)^2}{n-2+\left(t_{\alpha/2n,n-2}\right)^2}}, \tag{4.17}$$

where $t_{\alpha/2n,n-2}$ is a critical value of t distribution with *n-2* degrees of freedom and *α/2n* level of significance. For the one-sided test, the significance level is *α/n*. Then, the outlier x_j is the observation that leads to the largest $|x_i - \bar{x}|/sd$, where $\bar{x}$ is the sample mean and sd is the sample standard deviation. If T_s is lower than the critical value, there is no identified outlier. Alternatively, if we choose the x_j as an outlier, we can repeat the process with the removed observation x_j. The two-sided equation can be modified into two one-sided tests to test whether a minimum or maximum value is an outlier. [242], [244]

The Lr test, also known as the Tietjen-Moore test, enables to test for multiple *r* outliers at once. The test statistic has two one-sided forms for detecting r upper (4.18) or r lower outliers (4.19), and also a two-sided formula (4.20), defined as

$$L_{r\_upper} = \frac{\sum_{i=1}^{n-r}(x_i - \bar{x}_r)^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2} \tag{4.18}$$

$$L_{r\_lower} = \frac{\sum_{i=r+1}^{n}(x_i - \bar{x}_r)^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2} \tag{4.19}$$

$$L_r = \frac{\sum_{i=1}^{n-r}(z_i - \bar{z}_r)^2}{\sum_{i=1}^{n}(z_i - \bar{z})^2} \tag{4.20}$$

where the $\bar{x}$ is a sample mean of the whole dataset and $\bar{x}_r$ is a sample mean of *r* minimal or maximal observations removed depending on the formula used. In the two-sided formula (XX), the absolute residuals of observations are calculated as $r_i = |x_i - \bar{x}|$ and then sorted ascendingly into z_i. Then, the $\bar{x}_r$ is a sample mean with removed *r* largest observations. The value of L_r is between 0 and 1. If L_r is less than a critical value then the *r* outliers are identified as outliers. However, the Lr test is susceptible to false positives when there are false outliers among the *r* candidates. [242], [244]

The most attractive approach is the generalized extreme studentized deviate (GESD) statistical test. This enables testing up to r candidate outliers. It calculates the $Ri = \max(|x_i - \bar{x}|)/sd$ for all candidate outliers *i=(1,..,r)* but in each step *i* it finds the observation that maximizes the $|x_i - \bar{x}|$ and removes this observation from the sample size. The next iteration is on an *n-1* smaller dataset. Followingly, compute the *r* critical values λ_i by the formula

$$\lambda_i = \frac{(n-i)t_{p,n-i-1}}{\sqrt{\left(n-i-1+t_{p,n-i-1}^2\right)(n-i+1)}}, \tag{4.21}$$

where tp,ν is the 100p percentage point from the t distribution with ν degrees of freedom and the p is $p = 1 - [\alpha/2(n - i + 1)]$. The important aspect is selecting the r. Selecting a large r than needed increases computation time, but does not affect the detection of false positives. The r represents the upper bound of the number of outliers. It does not require the location of outliers. Also, it performs very well under conditions where previously presented methods fail. Furthermore, it is suitable for large samples. [242], [244]

4.2 Read-pair approach

The second method used is the read-pair approach. The read-pair approach can independently detect a wide spectrum of SV types. It employs the information from the alignment of pair-end reads such as the genomic position, distance between read pair, and their orientation to each other.

4.2.1 Detected signatures

Soylev recently presented rules to distinguish between two types of duplications, tandem and interspersed, and their direct and inverted (or indirect) forms [115]. The signature of reads from tandemly duplicated segments includes lower insert size (as reads are mapping closer than expected) and reversed both orientation and order of the reads (upstream read location mapping to the reverse and downstream read location mapping to the forward, i.e. -/+).

The interspersed duplication signatures include increased insert size and reads mapping to the opposing strands but with reversed order (+/- and -/+). Another case of duplication is inverted (or indirect) duplication, which shares signatures with inversion. Contrary, direct duplication (unchanged orientation), shares a signature with deletion. Thus, similarities make detection challenging and that is why the read-pair approach is used to validate read-depth results which can distinguish between duplication and deletion clearly. [115]

Interestingly, the combination of read-depth and read-pair approaches could discern between homozygous and heterozygous deletion. For homozygous, where both alleles are missing, reads are mapped farther apart and there is zero coverage. For heterozygous, where only one allele is missing, there is lower coverage compared to baseline, but reads are spanned normally caused they originate from the no-event region. [117]

The signature rules from Soylev are the following [115]. The signature of deletion is defined as reads mapped to the expected strands (+/-), plus strand for the first mate, and minus strand for the second mate. But the insert size is higher than the rest of the sequencing library.

The signature of direct tandem duplication includes both orientations of mapped mate reads, (+/-) and (-/+), but the insert size is lower than expected. If the reads were both mapped to the same strand, (+/+ or -/-), it would point to the indirect duplication.

The signature of interspersed inversed duplication is defined as both reads mapped to the same strand (+/+ or -/-) and with increased insert size.

The signature of interspersed direct duplication is defined as reads mapped to opposite strands in both ways (+/- or -/+) and with increased insert size.

4.2.2 Circular genome correction

The circular genome correction serves to tweak the reality that the reference genome for alignment is linearly represented, while the real bacterial genome is circular. Thus, the distances between read-pairs have to be corrected. However, this correction only affects the extremities, which are caused by the circular to linear conversion, and not the majority of concordant pair-reads. Values of insert size (reported as TLEN in BAM format) which are higher than half the reference genome's length are recalculated so that the genome length and pair-read length are subtracted (4.22). Vice versa, for negative values lower than half the reference genome's length, the genome length, and pair-reads length are added (4.23).

$$TLEN_{new}(+) = TLEN_{old} - 2 \times ReadLength - ReferenceLength \tag{4.22}$$

$$TLEN_{new}(-) = TLEN_{old} + 2 \times ReadLength + ReferenceLength \tag{4.23}$$

The illustration of the reality between circular and linear coordinates in the reference genome is in Figure 4.1. Notice that the real distance in the circular representation does not correspond to the distance in the linear transformation.

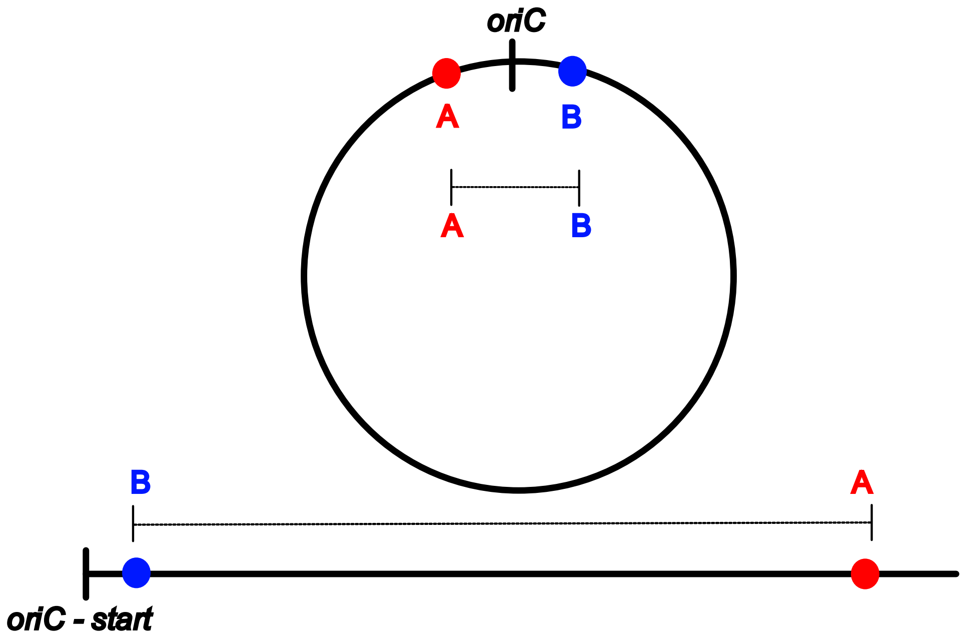


Figure 4.1 – Depiction of circular genome distance

5 CNPROSCAN

5.1 Merging the detection approaches

The earliest tools for CNV detection used only a single approach but soon there were released tools combining multiple approaches. This combination proved to be very useful as each approach is limited in its detection abilities. While for detecting large SVs the pair-end method is the most applicable, the detection of CNVs can not be done without the read-depth method. All methods focused on CNVs employ read-depth information. This information can be extended by other approaches as was presented that methods combining read-depth with split-read or read-pair methods have both sensitivity and specificity improved for small CNVs [160]. The majority of CNV detection tools for human genomes rely on paired samples, and if no control reference is available, pooling even the observed samples together can serve as the reference. Thus, I decided to merge the read-depth approach with read-pair information, because this enables us to narrow the subtype of duplications [115].

5.2 The Algorithm Design

The CNproScan uses the 'sandwich' design with partial blocks stacked vertically, as illustrated in Figure 5.1. The program consists of several main blocks. The first one is coverage normalization, the second is outliers detection to determine CNVs, the third is the application of the read-pair approach and signature rules to narrow the CNV subtype and the last is formatting the output. Each block is here described more from the implementation aspect. [246]

5.2.1 Data preparation

The preparation of sequencing reads before the CNproScan's detection of CNVs is carried out by the usual procedure. It is useful to analyze library quality metrics by FastQC[247] or fastp[248], as these are the most used tools. Fastp is an all-in-one tool for quality control, adapter, and quality score trimming. Other frequent tools for trimming are Trim Galore and cutadapt [249].

Then, the reads are mapped by the aligner. The BWA-MEM was used in the testing [122], [220]. BWA places the multi-mapping equally scored reads randomly but with an alignment score of zero. Other locations are reported in the XA tag defined by the SAM/BAM file format [250].

The samtools package is used to handle the rest of the work [250]. The alignment is sorted and written as a binary BAM file. The coverage signal is obtained by the command "samtools depth" with parameter -a which includes zero coverage positions.

For the optional mappability correction, the required genome mappability file is obtained from GenMap [251]. The settings -K 30 -E 2, meaning the size of unique k-mers and allowed mismatches, was generally used for all analysis.

If the user is interested in the correction of the origin of replication bias, then, the location or multiple locations of *oriC* is necessary. This information can be searched for in the DoriC database [239]. The record from DoriC must match with the corresponding genome reference used for alignment.

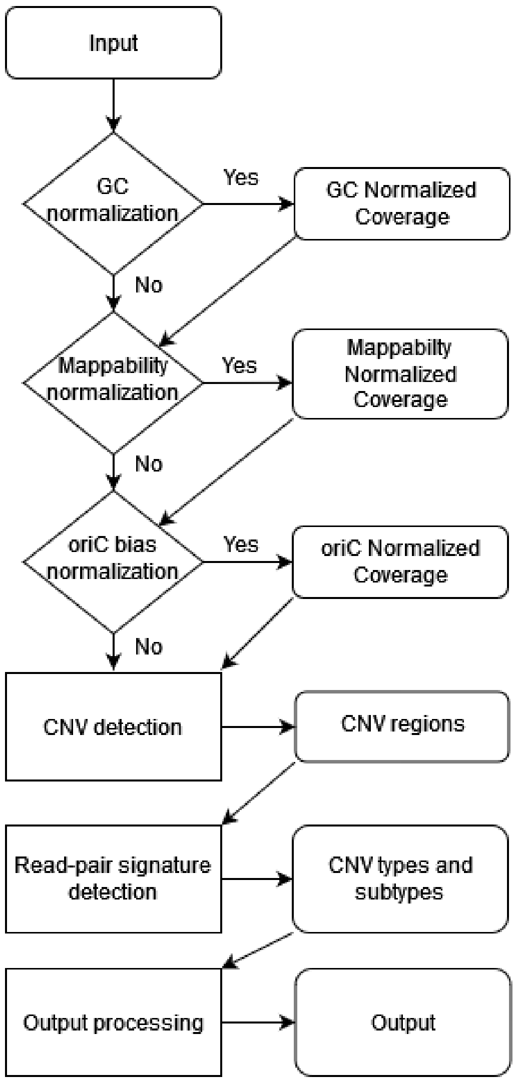


Figure 5.1 – CNproScan workflow

5.2.2 Data used

In the chapter on algorithm design, several real and artificial samples were used in the testing. The real data are also used in the chapter ProcaryaSV. The overview of the datasets used in the discussion of implementation is in Table 5.1.

Table 5.1 – Overview of used real datasets

Organism	**Accessory ID**	**No. of samples**	**Citation**
Staphylococcus aureus	PRJNA497094	92	[252]
Escherichia coli	DRA005229	58	[253]
Lactobacillus casei	PRJNA342061	50	[254]
Klebsiella pneumoniae	PRJNA515630	48	[255]

Alternatively, the artificial dataset of 30 simulated and induced CNVs was used taken from benchmarking the CNOGpro [152], which is described in the chapter Benchmarking on the simulated data.

5.2.3 Main function

The CNproScan was developed initially as a set of MATLAB functions and during the peer review, it was rewritten into R. Both versions share the same methodology. Recently, the R version was updated to version 1.0, and these modifications are presented in a separate chapter. Both versions are hosted in GitHub repositories (Table 5.2). All normalizations are optional, but the GC and the mappability normalization are recommended as commonly used.

Table 5.2 – CNproScan GitHub repositories

R version	https://github.com/robinjugas/CNproScan
MATLAB version	https://github.com/robinjugas/CNproScanMatlab

5.2.4 GC Normalization

The GC normalization is done with the use of the modified Yoon approach (4.1). The normalization requires to use of a sliding window. Benjamini et Speed notes that a window size of at least fragment length should be used [233]. Yoon ties the GC normalization and read-depth approach as it is common with the use of a 100bp window[131]. The CNproScan does not use a window approach for CNV detection, but it relies on a single-base resolution of the coverage signal. However, the normalization is done in a window manner as the single-base approach is computationally demanding and there is no strong rationalization as GC content has to be calculated over a certain region. Originally, I applied the normalization to the single bases but applying it in a window brings a performance bonus and provides the same results.

The table of GC content percentage values is based on 50 bp windows and their GC content. This table is filled with read-depth values of the corresponding GC value. Then, the median values are used as prescribed by Yoon. Several exceptions can happen which are not described by Yoon. The windows of high or low GC content can have zero or very low coverage leading to division by zero later in the formula. Also, if the median coverage of given GC content (m_{iGC}) is around 1, then the normalized values are extremely high and this creates false spikes in the coverage signal. There are three rules to suppress these exceptions without inducing such noise. Furthermore, I added the mean value that can be used instead of the median. First, there is no normalization if the median coverage of the GC is zero or the base coverage is zero. Also, if both the median and mean are less than 1, the coverage is not corrected. Alternatively, if the median is less or equal to 1 and the mean is higher than 1, the mean value is then used instead. The median is used if is higher than 1. These rules avoid division by zero and by a number less than 1 which induces noise.

The dependence of read depth on the GC content is in Figure 5.2. In this figure for *E. coli*, the relation between the two variables can be clearly seen. With increasing GC content, the observed read depth is decreasing. The average GC content in *E. coli* is around 50%. The normalization removed this relation completely. The Spearman correlation coefficient and p-values are in Table 5.3. The visible relation is supported by Spearman's Rho of -0.9 denoting negative relation. The results of the other three samples are in Supplementary Table 2. The same conclusion is valid for *S. aureus* and *L. casei*, however, the *K. pneumoniae* sample has a weaker correlation with Spearman's Rho of -0.3017. For all normalized samples, the correlation was not statically significant, which means the normalization was successful. The plots of differences between raw and normalized coverage signals are in Supplementary Figures B1-B4.

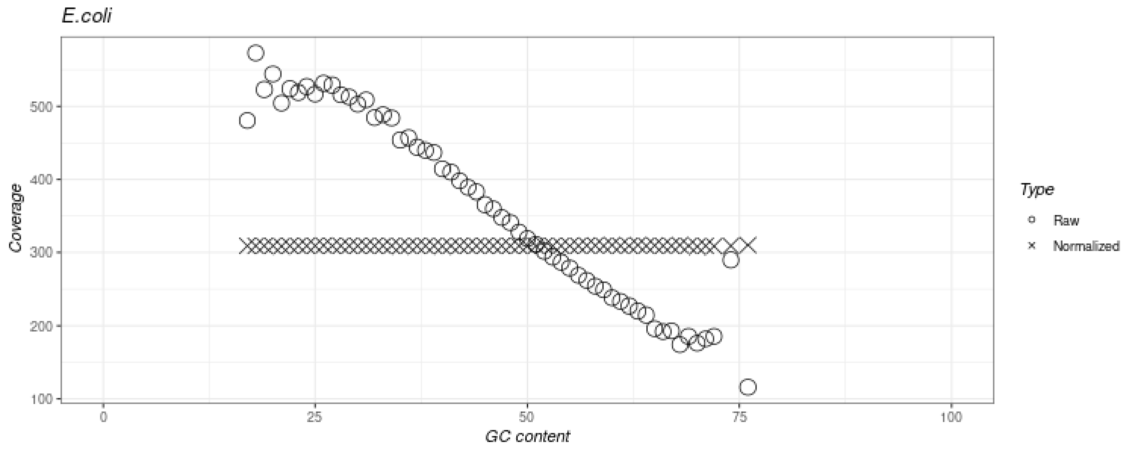


Figure 5.2 – Dependency of read-depth on GC content for raw and normalized values

Table 5.3 – Spearman correlation test for GC normalization (*E. coli*)

Condition	**Rho**	**P-value**	**Significant**
Raw	-0.971	8.395e-37	Yes
Normalized	0.222	0.093	No

5.2.5 Mappability Normalization

The mappability normalization is based on the Magi approach and formula presented earlier (4.2) [127]. The approach is very similar to GC normalization. We use mappability scores calculated from an external tool named GenMap [251]. GenMap focus on the problem of finding the occurrence of a substring with length *k* in the sequence while allowing some errors *e*, when the sequence here is the reference genome sequence. It returns a mappability score, defined as the inverse of the occurrence frequency, of 1 for a unique substring and a mappability score close to 0 for repetitive substrings. GenMap was chosen because it is accessible as a conda package, based on the paper it outperforms previously published competing packages, is exact and non-heuristic, and enables the choice of a number of errors. The *k*=30 and *e*=2 were used by authors to perform analysis on *Klebsiella pn*. Thus, we take over the same settings. [251], [256]

Instead of the GC table, the mappability table is constructed where each row corresponds to a mappability score, and the columns are observed coverage values. The window is not set up, the coverage values are calculated from regions defined in the bedgraph formatted file from the GenMap tool. The normalized coverage value is then computed as previously, each window has its values multiplied by the division of median coverage and median coverage of windows of the same mappability score.

The dependence between the two variables, the read depth and mappability score for *E. coli* is plotted in Figure 5.3 and correlation coefficients are in Table 5.4 and Supplementary Table 3. The raw points are randomly scattered, while the normalized coverage leads to a slightly higher Spearman's Rho, both insignificant. Only the *S.aureus* sample had a statistically significant correlation of coverage and mappability and this correlation was reduced successfully by normalization. The plots of differences between raw and normalized coverage signals are in Supplementary Figures C1-C4.

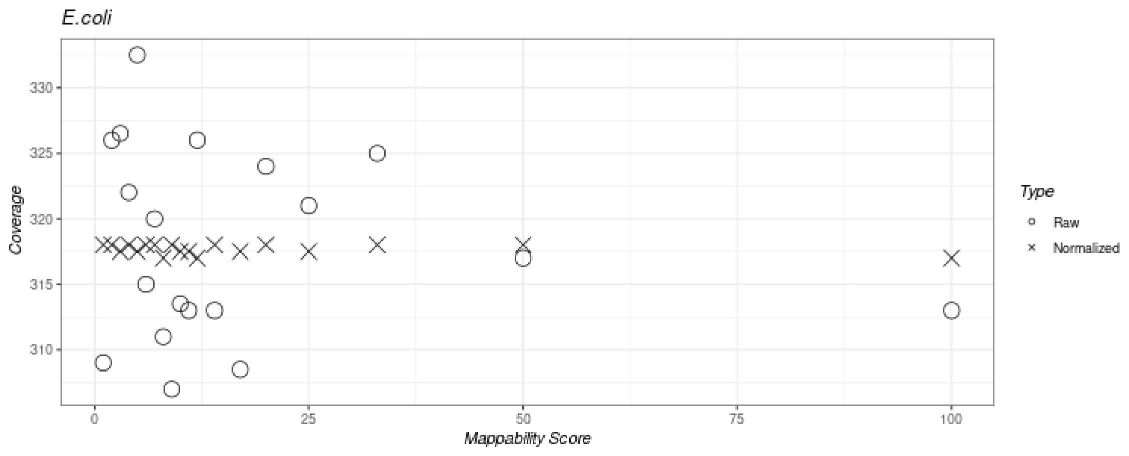


Figure 5.3 – Dependency of read-depth on mappability score for raw and normalized values

Table 5.4 – Spearman correlation test for mappability normalization (*E. coli*)

Condition	**Rho**	**P-value**	**Significant**
Raw	-0.1582	0.5176	No
Normalized	-0.2023	0.4061	No

5.2.6 Replication origin bias normalization

The replication origin bias is normalized by our approach, which is based on the previously presented principles for GC and mappability normalization. The genome is binned into 100bp windows similarly to the GC normalization. Whether the bias is corrected depends on the user and the result of the Spearman correlation test p-value.

Firstly, it is required to remove any outliers. It is because the distance to *oriC* is symmetrical and deletion or duplication on one side of the symmetry could completely deflect the normalization of the regions with the same distance to the replication origin. The outliers are removed using the 1.5 times IQR (interquartile range) rule on both tails. Then, the distance to *oriC* is calculated in windows of 100 bp. The circular genome correction is applied so that the minimum value of all possible constellations is chosen.

The important parameter is the level of rounding. This parameter impacts how many windows are taken together in estimating the median read depth of a certain distance to *oriC*. Rounding to thousands means that approximately ten 100bp windows on each side are taken together, rounding to tens of thousand means a hundred windows are taken together. The table of values of *oriC* distances and median read depths is constructed. Importantly, the Spearman correlation value is computed between the *oriC* distances and estimated read depth medians. The p-value is calculated for the alternative hypothesis that Spearman's correlation coefficient Rho (-1,1) is different from zero. The normalization is further applied if the p-value of this test is less than the alpha value of 0.05. The p-values are computed via the asymptotic approximation, which means that they depended on the number of *oriC* distance values and will likely be less than the alpha level for the lower rounding level. However, it was observed that higher rounding is more robust, and rounding to tens of thousand is applied. The replication origin bias is then normalized by the formula

$$\overline{RC_i} = RC_i \cdot \frac{m}{m_{iORICdist}}, \tag{5.1}$$

where m stands for the median value of read-depth of all windows and $m_{iORICdist}$ for the median value of the windows with the same *oriC* distance.

The information about the genomic position of replication origin is accessible in the DoriC database [239]. If the *oriC* normalization is intended, it is useful to check if there is a record in DoriC for the selected genome reference or choose the different one.

The *oriC* position for *E. coli* was set up at 3923657bp. As in the mappability case, the scatter plots (Figure 5.4) of dependency look similar for *E. coli* and *K. pneumoniae.* The correlations are in Table 5.5, and Supplementary Table 4 for other samples. The *S. aureus* and *L. casei* have both more visible negative correlations with coefficients of -0.82 and -0.93 respectively, while *E.coli* has a correlation coefficient of 0.23 and *K. pneumoniae* -0.25. Fully removing the bias was not always successful, as all *K. pneumoniae, S. aureus, and L.casei* have significant p-values even for the normalized condition. Only *E.coli* was successfully normalized. The plots of differences between raw and normalized coverage signals are in Supplementary Figures D1-D4.

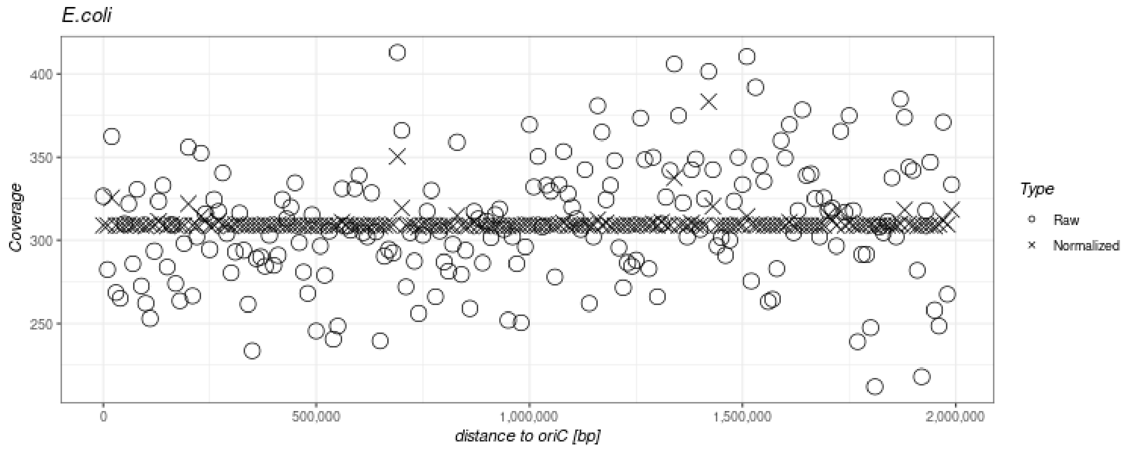


Figure 5.4 – Dependency of read-depth on *oriC* distance for raw and normalized values

Table 5.5 – Spearman correlation test for replication bias normalization (*E. coli*)

Condition	**Rho**	**P-value**	**Significant**
Raw	0.2358	0.0007	Yes
Normalized	0.0902	0.2037	No

5.2.7 Read depth normality distribution

The underlying theory behind the applicability of the outliers detection algorithm is that data are somewhat normally distributed. Here, the Q-Q plot and histograms of both artificial and selected real samples are shown.

Firstly, the Q-Q plots of artificial samples are in Supplementary Figure A1. The four samples are constructed in four various mean coverages (10×, 20×, 100×, 200×) and with or without artificially imputed CNVs. The dataset normality is assumed from the linearity of data along the normal distribution line. Ideally, the dataset and line should overlap. Here, we can see how CNVs create the tails on both sides of otherwise normally distributed coverage. From the coverage value of 100× in the sample without CNV we can see the visible drop created by zero or close to zero values of coverage. This is induced most likely by the reads simulator and also by the fact that the two previous coverage values are very close to zero. For higher coverages, there is almost no upper tail in the no CNV samples. On the opposite side, there are upper tails formed by induced duplications and more visible lower tails induced by deletions. However, a large part of the samples is normally distributed.

Similarly, the histograms are plotted in Supplementary Figure A2, together with a normal distribution bell curve with parameters $N(\mu,\sigma^2)$ taken from the dataset statistics. We can see how for lower coverages the normal distribution does not fit. The sample histograms are too narrow. The histogram is widened by the presence of CNVs as displayed in the sample of 20× coverage with CNVs. The no CNVs samples of 100× and 200× fit normal distribution perfectly. While for samples with induced CNVs, the mean and variance of the normal distribution curve were influenced by them, and the "peak" is too narrow.

A different situation is for real datasets. Every sequencing organism has its specifics. In Q-Q plots in Supplementary Figure A3, there are raw coverage values on the left side, and on the right side, there are coverage values with detected and removed CNV values. The CNproScan was used. Firstly, the *L. casei* was a specific sample with almost no CNVs. This is more prominently visible in the histogram in Supplementary Figure A4. Other samples with CNVs removed fit the normal distribution more. Contrary to *L. casei*, *K. pneumoniae* has a specific shape created by the presence of large deletions and duplications. It is important to note that the selection of reference genome plays a huge role in the coverage profile. The reference used for *K. pneumoniae* is specific because is composed of multiple substrains.

The histograms in Supplementary Figures A2 and A4 show how samples can fit normal distributions both with and without CNVs. The differences are again *L. casei* and *K. pneumoniae.* However, with CNVs removed the *K. pneumoniae* fits normal distribution very well.

5.2.8 Outliers as CNV candidates

The normalized coverage signal is sent into the outliers analysis. Firstly, zero coverage is apriori considered as deletions and labeled separately. This also removes part of the lower tail of coverage distribution. The other upper tail is being removed by outlier detection. For this task, the CNproScan employs the GESD outlier detection algorithm described before. As this algorithm requires the upper bound of the suspected outliers. To serve a robust estimation of this upper bound, the modified Z-score outliers detection is used and values with a modified Z-score above 3.5 are labeled as candidate outliers. This usually leaves a large number of candidates, meaning several thousand and e.g., more than ten thousand candidates for real *Klebsiella pneumoniae* samples.

To reduce the performance drawback of testing thousands of values in a for-cycle, the GESD testing is done in a parallel way. This was possible because the task is possible to parallelize. This is done in the R version with the use of R packages *parallel*, *doParallel,* and *foreach*. Simply done, the whole genome is divided into *n* sections which are tested separately and parallel. The argument *cores* in R main function serve as the definition of the number *n*. After each partial segment is done, the results, which are genomic positions of significantly large coverage values, are merged into a single vector.

In the Matlab version, there is a parallelization of computing multiple samples at once through a script *paralell_run_CNproScan*.

More changes are made in the updated R version 1.0. Outliers from both tails are removed first with 3 times interquartile-range rule (anything beyond this value is labeled as an initial outlier). This tweak reduces the search space for the computationally demanding GESD procedure and also increases slightly the sensitivity.

The results are post-processed. The vector of outliers is sorted and the gaps between outliers are detected using the lagged differences function. Then, depending on the parameter *peakDistanceThreshold*, which is set up to 20bp, the adjacent outliers closer than 20bp are merged into consecutive segments. These serve as a basis for CNV events.

In Figure 5.5 the results of outliers detection are displayed for the artificial genome. The details of the creation of the artificial genome dataset are described later in the chapter Benchmarking on simulated data. The zero coverage values are removed first (in red). The candidate outliers from the modified Z-score method are in blue and multiple of them are overlapped with blue as they were confirmed by the GESD outliers test.

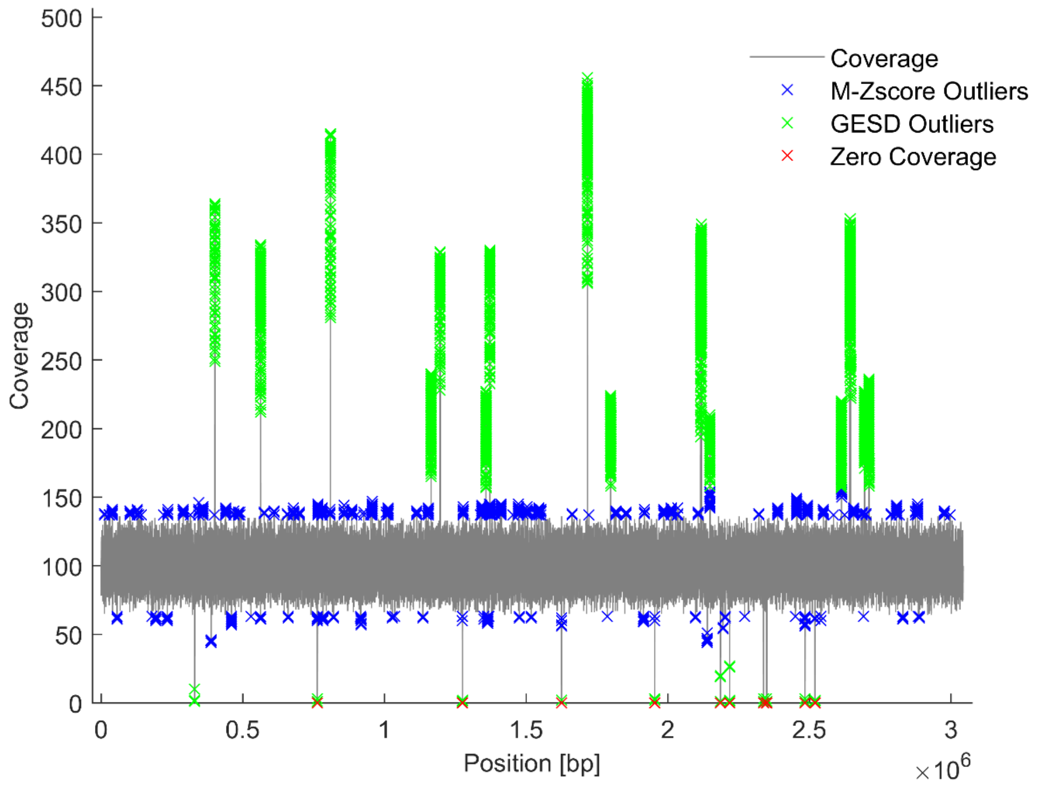


Figure 5.5 – Detecting outliers in artificial genome

5.2.9 Extending CNV boundaries

Because of the nature of outliers detection, only the most significant parts of CNVs are uncovered, i.e. for duplications, only the peaks are labeled yet. To extend the borders of the CNV down to the baseline, the slope of a line is used. The line is given as a coverage region. The slope is calculated as $m = \frac{y_2 - y_1}{x_2 - x_1}$ and the distance between x-axis values $x_2 - x_1$ is defined by a specified step (11 bp, optional). The slope is calculated gradually on both ends of the peak until there is a change in the numerical sign for the value of slope m. If the change of slope is detected x-times (x defined as 5, optional), then it is considered as the CNV's border. The updated version adds the condition of reaching the baseline defined as the average of the coverage. A detail of one CNV with extended boundaries is plotted in Figure 5.6. Noticed how the whole depth of CNV is detected compared to the previous Figure 5.5 (third peak from the end).

5.2.10 Read-pair information

Since CNproScan detects solely CNVs, I choose only a few signatures to use from the read-pair approach. The features of deletion, tandem duplication, and interspersed duplication are targeted. The signatures are as defined in Soylev's work [115].

Contrary to other approaches which merge the two pieces of information, the read-

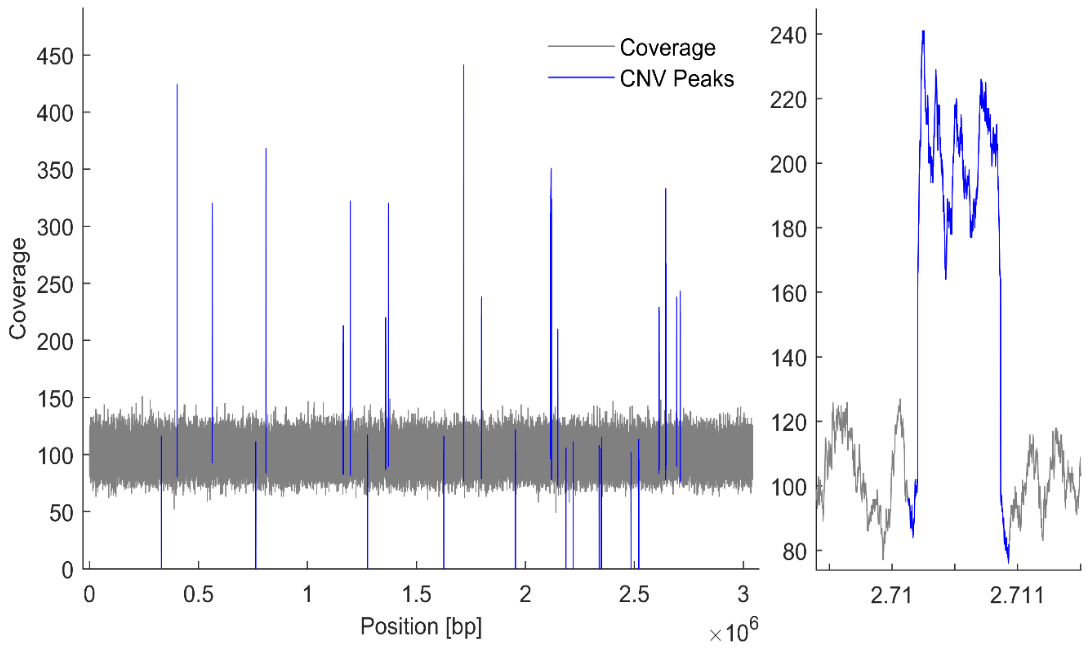


Figure 5.6 – Detail on extending the CNV boundaries

pair approach helps to validate and specify the CNV events detected from the read-depth approach. Both approaches are not equal, rather the read-pair information is subjugated to the main read-depth information. This is because as mentioned earlier, certain signatures are not exclusive, and specifically direct duplication signature is the same as a deletion signature. Because of this reality, the read-pair approach is subjugated to the read-depth approach which can distinguish between duplication and deletion very clearly.

The usual approach is to cluster together neighboring signatures and assign an SV type to them. Here, in CNproScan I search for outliers in the signatures because the distribution of fragment sizes in a library is Gaussian and the detected signature is usually largely distant from the normal state. The outliers are defined on a simple rule of 1.5IQR, which means that insert sizes larger than the sum of the upper Q3 quartile and 1.5 times the interquartile range are labeled as an outlier.

Furthermore, the features are searched only in the regions of already detected CNVs and not genome-wide. This reduces the computational time. The genomic regions are scanned inside the detected CNV boundaries extended by the insert size on both ends. The decision of which SV subtype will be chosen is done by selecting the most prevalent signature inside the region.

The detection of discordant reads uses the fields defined in the SAM/BAM format [230], [250], mainly TLEN (Template length), and bitwise FLAG, which contains information about the read`s relative orientation, etc. In the R version, packages *Rsamtools*, *GenomicRanges*, and *IRanges* were used to access the BAM file structure. In the Matlab version, the *bamread* function from the Bioinformatics Toolbox was used.

As already mentioned, the BAM is scanned only in the regions of detected CNVs. The boundaries are defined as the start and end of CNV's coordinates plus/minus the insert size. The insert size is defined as the median of absolute values of whole BAM reads. Similarly, the interquartile range, the first and third quartiles are defined based on the whole BAM file. The "isize" in Rsamtools (TLEN in BAM definition) is used for these estimations. For median read length, the "qwidth" is used.

The circular genome correction is used as described, and each CNV region defined in the read-depth approach part is scanned for reads defined by specified signature rules. The genome-wide signal of paired read distances is plotted in Figure 5.7 with outliers highlighted in red and blue, denoting simply insertion and deletion of genomic sequence.

The theoretical signature rules were extended because of some observations from the testing and are all listed in Table 5.6.

Table 5.6 – Overview of applicated signature rules

Type	**Subtype**	**Strand orientation**	**Insert Size**
Deletion		+/- (-/+)	Higher
Tandem Duplication	Direct	+/- -/+	Lower
Tandem Duplication	Indirect	+/+ -/-	Lower
Interspersed Duplication	Direct	+/- -/+	Higher
Interspersed Duplication	Indirect	+/+ -/-	Higher

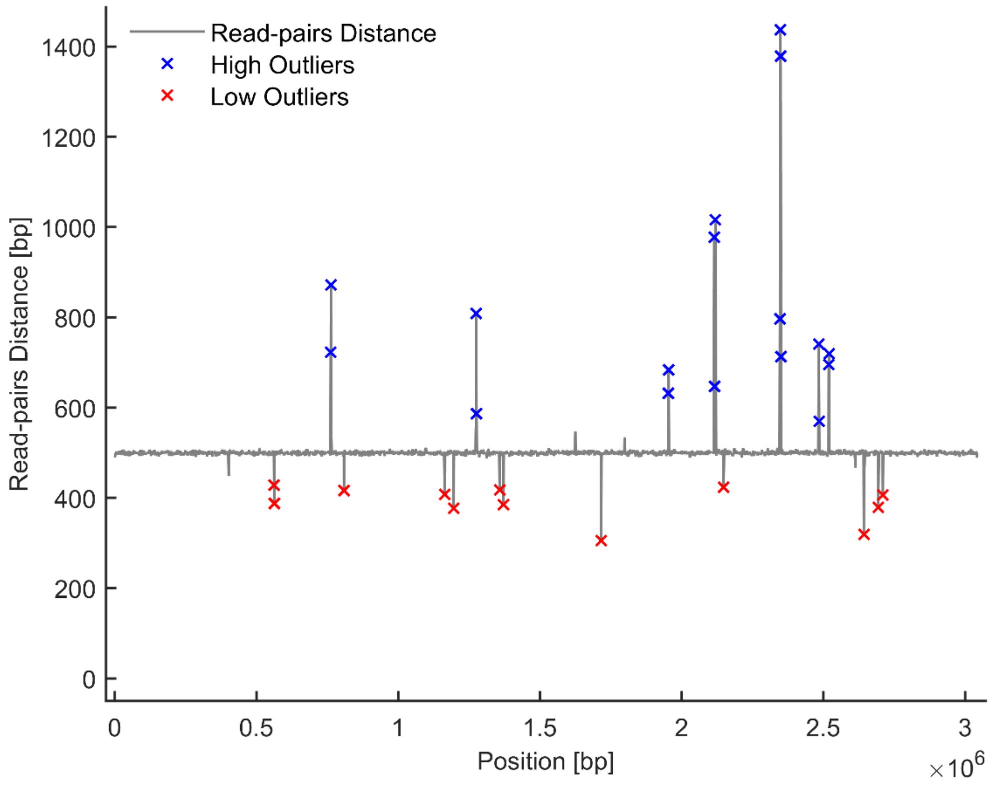


Figure 5.7 – Plot of read-pair distances across genome

Firstly, the usual deletion signature is stated as the +/- position of reads. But observing the IGV outputs it was discovered that deletion produces both reads placement, i.e. -/+ too. Narrowing it to +/- only would decrease, by almost a half, the number of reads which would support this signature.

Secondly, it was observed that large tandem duplication can seem like interspersed duplication. This is caused by mirroring the length of tandem duplication itself into the observed insert size of reads around its breakpoints. This is demonstrated as insert size labeled as upper outliers instead of expected lower outliers. For example, the tandem duplication above 1000 bp will have observed reads insert sizes of similar length, while the median is only 500 bp in the artificial dataset. Thus, the setting for higher outliers has to be much higher. I applied an arbitrary rule of 10 times the IQR for labeling upper outliers for interspersed duplications. The boxplots of three tandem duplications are in Figure 5.8. Notice how statistics of insert sizes correspond with event length. Tandem duplication of 3776, the third boxplot, should theoretically have reads with insert sizes lower than the majority of reads in the sample. The mixed signatures of tandem and interspersed duplication make their distinction more difficult, but this can be mitigated by a much higher threshold. However, this is a possible weakness of the approach.

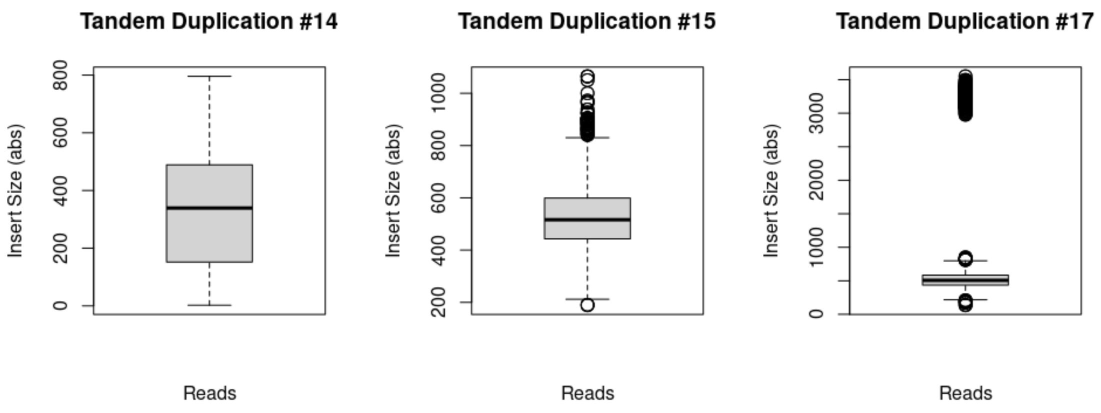


Figure 5.8 – Insert size of tandem duplications with size of 354, 1302, 3776 bp

As already partly mentioned, the insert size of the event copies somewhat the length of the event. This is expected, but not commonly mentioned in the literature. For deletions, this is displayed in Figure 5.9, and for tandem duplications in previous Figure 5.8. Deletion number 6 has originally 828 bp and the boxplot of insert sizes of reads in the given region reaches an approximately similar level of values. The same applies to the deletion of 1677 bp where the third quartile corresponds approximately to the same value.

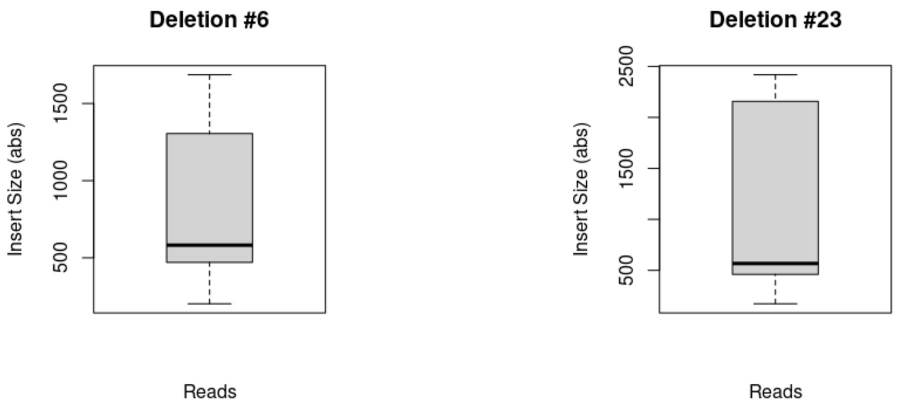


Figure 5.9 – Insert size boxplot of deletions of size of 828 and 1677 bp

Furthermore, it was observed that for small events, the read-pair approach is less usable. This is displayed in the IGV screenshots in Supplementary Figures E1-E2, where two deletions are plotted. While for longer deletion number 6 there are multiple reads creating signature clusters, for smaller deletion number 13 with the length of 134 bp, there are almost none. The smaller gaps in the reference genome are bridged by using split-read signatures (aka soft-clipping) or are likely thrown away depending on the alignment algorithm. The event with no supporting reads for any type of event could be then easily tagged as a possible false positive. This is still being applied but with caution.

For every CNV event, there are counted all reads supporting the above-defined signatures. The main distinction between duplication and deletion is done at the read depth level, the deletion is lower than the average of the coverage signal while duplication is oppositely higher than the average. The distinction between tandem and interspersed and their direct and indirect forms is done by applying the read-pair approach.

5.2.11 Output files

The output consists of a table where rows correspond to the detected CNVs and columns to several reported features, such as CNV coordinates, length, type, and subtype, and read counts of supporting reads for every read-pair signature.

For the MATLAB version, the cell data structure can be written into an Excel spreadsheet. The same applies to the R version and the resulting 'data.frame' variable. Furthermore, the VCF file, defined similarly to CNVnator and LUMPY, is written to a drive.

5.3 Benchmarking on the simulated data

The following chapter is based on the published paper of CNproScan [246] and the results are based on the MATLAB version. A more detailed analysis of multiple aspects is in the next chapter focused on the R version. Since the results of the MATLAB version are very similar to the later R version, I decided to put here both results and make them complementary. Thus, the MATLAB version results are focused on comparison with other state-of-the-art methods, while the R results chapter is focused on various aspects and is compared to the MATLAB version for continuity. The benchmarking with other tools on the real dataset is in the next chapter ProcaryaSV.

5.3.1 Test dataset

The performance of CNproScan was evaluated on the dataset which had been previously used in the testing of the CNOGpro package [152]. This dataset is based on the *S. aureus* genome sequence (in Table 5.7) into which were imputed 30 artificial CNVs with defined genomic coordinates, lengths, and copy-number. These CNVs are listed in Table 5.8. There are 12 deletions and 18 duplications of various lengths, mainly focused on the small events.

Table 5.7 – Testing genome

Sequence	**NCBI Accession**
Staphylococcus aureus subsp. *aureus* TW20	GenBank: NC_017331

The dataset has two parts – one with imputed CNVs and the second one with no CNVs to evaluate the metric of true negatives. These two datasets were constructed with different coverage values – 10×, 20×, 100×, and 200×. The sequencing reads were generated with the ART reads simulator [228] and then processed by the described pipeline.

Table 5.8 – Dataset of 30 artificial CNVs

Number	**Start Position**	**Stop Position**	**Segment length**	**True CNV number**
1	330012	330015	4	0
2	344821	344843	23	2
3	388478	388485	8	0
4	402047	402109	63	4
5	562944	563213	270	3
6	762780	763607	828	0
7	809546	809626	81	4
8	1164412	1164687	276	2
9	1196369	1196578	210	3
10	1275253	1275864	612	0
11	1358393	1358649	257	2
12	1371773	1371988	216	3
13	1625170	1625303	134	0
14	1716617	1716970	354	4
15	1798287	1799588	1302	2
16	1953890	1954411	522	0
17	2115756	2119531	3776	3
18	2148564	2148827	264	2
19	2186039	2186055	17	0
20	2195082	2195085	4	2
21	2219052	2219068	17	0
22	2338314	2338339	26	0
23	2348148	2349824	1677	0
24	2484353	2484868	516	0
25	2519733	2520386	654	0
26	2612863	2613755	893	2
27	2643228	2643743	516	3
28	2645385	2645421	37	3
29	2694225	2694725	501	2
30	2710239	2710898	660	2

The reads were generated with these ART (command art_illumina) settings – fold coverage (-f) set to the tested value (10×, 20×,100×,200×), read length set to 76 bp (-l), paired-end reads (-p), median insert size 500 bp (-m), the standard deviation of fragment length 100bp (-s), and the sequencer profile set to HighSeq (HS25). The same settings were used in the compared methodology.

The performance of CNproScan was compared directly with LUMPY [125], CNVnator [173], Pindel [113], and DELLY [118]. And indirectly with CNOGpro [152], cnv-seq[158], and cn.MOPS [155], where I adopted the previously published results. All tools are summarized in Table 5.9.

Regarding the competing tools settings, we ran LUMPY using lumpy express settings and CNVnator with a 30bp bin size specified. The insert size for Pindel was set as 348 bp. All other settings for all tools were left at default values. To compare our results with the reference tool, we recalculated the CNOGpro results using the same methodology as ours. This recalculation is necessary as CNOGpro makes the calculation using the gene regions. Therefore, we consider it possible to count only the 30 CNV events as true negatives (TN) instead of 5437 gene regions. Other metrics – true positives (TP), false positives (FP), and false negatives (FN) are based on results evaluation.

Table 5.9 – Overview of competing tools

Tool	**Method**	**Version**
LUMPY	RD+PR+SR	0.2.13
CNVnator	RD	0.4.1
Pindel	SR	0.2.5b9
DELLY	RD+PR+SR	0.8.7
CNOGpro	RD	1.1
cnv-seq	RD	1.0
cn.MOPS	RD	3.1
RD – read-depth, PR – pair-read, SR – split-read		

The main focus was on 100× coverage, then the only tools competing well were evaluated for other coverages 10×, 20×, and 200×. The results are evaluated by the metrics of the confusion matrix. The Accuracy, Sensitivity/Recall, Specificity, Precision, and F1 score are all used across the results chapters. Lastly, the results and discussion are taken from CNproScan's published paper [246].

5.3.2 Results for coverage 100×

The most emphasis was put on the 100× coverage. All 8 tools are benchmarked for this value of coverage. It is high enough to provide a sufficient signal-to-noise ratio with easily detectable CNVs.

The complete results with the number of correct and false observations, and performance metrics are in Table 5.10.

A common problem stated in the literature is a high false discovery rate [141], where only Pindel failed significantly with 789 FPs. All tools performed well with a minimum number of false positives.

Focusing on the default 100× coverage (in Table 5.10), the overall accuracy achieved was 93% and was the highest among tools. CNproScan detected 26 TP. Four FN CNVs were short regions under 26 bp in length, consisting of 2 deletions and 2 regions with a copy number of two. There was a single CNV event detected outside the original coordinates, which we consider an FN case.

CNproScan and Pindel were both able to detect shorter CNV events than other methods. Pindel has higher sensitivity as it was able to detect 27 out of 30 CNVs. However, Pindel's high sensitivity has the drawback of a high false positive rate. Pindel detected 371 CNVs, mainly deletions, in the empty reference dataset. Furthermore, there were another 418 FPs in the dataset with CNVs. A high false discovery rate in CNV detection is a common problem stated in the literature [141], however, only Pindel suffered from this.

Table 5.10 – Results for coverage 100×

	CNproScan	**CNOGpro**	**Cnv-seq**	**cn.MOPS**	**LUMPY**	**CNVnator**	**DELLY2**	**Pindel**
TP	26	22	14	7	13	21	22	27
FP	0	0	0	0	2	0	0	789
FN	4	8	16	23	17	9	8	3
TN	30	30	30	30	30	30	30	30
Accuracy	93.3	86.7	73.3	61.7	69.4	85.0	86.7	6.7
Sensitivity	86.7	73.3	46.7	23.3	43.3	70.0	73.3	90.0
Precision	100.0	100.0	100.0	100.0	86.7	100.0	100.0	3.3
Specificity	100.0	100.0	100.0	100.0	93.8	100.0	100.0	3.7
F1 score	92.9	84.6	63.6	37.8	57.8	82.4	84.6	6.4

Other tools detected fewer CNVs. Sorted from the lowest number of TPs, there was cn.MOPS, LUMPY, cnv-seq, CNVnator, and equal CNOGpro and DELLY2. Since they all detected zero or a very low number of FPs, other metrics are influenced by the number of TP and FN. Thus, precision and specificity for all tools except Pindel were high.

CNproScan achieved the highest F1 score. The close competitors in this metric were CNOGpro, CNVnator, and DELLY2.

Although DELLY2 and LUMPY are both hybrid triple method combinations, they differ significantly in the detection of CNVs. DELLY2 performed better.

The detection of short CNVs with a low copy number is the most challenging task. For 100× coverage, we can conclude that CNproScan detected duplicated CNVs longer than 37 bp. Two duplicated CNVs of 4bp and 23bp lengths were not detected. The shortest detected deletion was 4bp and then two 17bp deletions.

The performance of the other tools varied. Pindel (90%) followed by CNproScan (86.67%) achieved the highest sensitivity. The third best performing in sensitivity were CNOGpro (73.33), DELLY (73.33), and LUMPY (70.00). CNproScan achieved the highest accuracy (93.33%). CNOGpro (86.67%.), LUMPY (85.00) and DELLY (86.67%.) were close in accuracy.

5.3.3 Results for coverage 10, 20, 200×

In the evaluation of other coverage's effect on the performance, only the best performers from the previous chapter were selected to reduce the complexity of the results. Selected were: CNproScan, CNOGpro, CNVnator, LUMPY, DELLY, and PINDEL.

I benchmarked CNproScan and others at four different coverage values: 10×, 20×, 100×, and 200×. The complete performance metrics are in Table 5.11. The highest values per row are highlighted in bold font type. The CNOGpro was aborted at 200× coverage because of an under-dispersion error, so the results are missing for this coverage.

For 10× coverage, the CNproScan's sensitivity was 66.67%, and 20 out of 30 CNVs were detected. Pindel had the highest TP count of 26, while also having the highest FP rate. The second highest TP count has DELLY and CNproScan. LUMPY has 17 TPs. DELLY and LUMPY had both zero FP. Contrary, there were 19 FP and an additional 20 FP in an empty dataset detected by CNproScan. The combined metric score was the best for LUMPY and DELLY, then CNVnator followed by CNproScan. The hybrid methods LUMPY and DELLY performed very well in the shallow coverage.

For 20× coverage, CNproScan achieved the highest accuracy (86%) and detected 22 TP. CNOGpro also detected 22 TP, LUMPY 21 TP, DELLY 20, and Pindel 27 TP, thus Pindel had the highest sensitivity. There was no FP detected with CNproScan. There is a visible step in detection quality from increasing coverage from 10× to 20×. The combined metric score was the best for CNproScan followed by LUMPY and DELLY.

100× coverage was discussed in the previous chapter, the highest combined score was achieved by CNproScan followed by CNOGpro, LUMPY, and DELLY. Only Pindel detected one more TP than CNproScan but suffered from a high false positive rate across the complete artificial dataset.

Table 5.11 – Results of all coverage values

10×						
	CNproScan	**CNOGpro**	**CNVnator**	**LUMPY**	**DELLY**	**PINDEL**
Accuracy	50.5	38.7	62.3	78.3	**83.3**	40.0
Sensitivity	66.7	43.3	26.7	56.7	66.7	**86.7**
Precision	33.9	20.3	88.9	**100.0**	**100.0**	24.5
Specificity	43.5	37.0	96.8	**100.0**	**100.0**	27.3
F1 score	44.9	27.7	41.0	**72.3**	**80.0**	38.2
20×						
	CNproScan	**CNOGpro**	**CNVnator**	**LUMPY**	**DELLY**	**PINDEL**
Accuracy	**86.7**	68.4	68.3	85.0	83.3	25.8
Sensitivity	73.3	73.3	36.7	70.0	66.7	**90.0**
Precision	**100.0**	57.9	**100.0**	**100.0**	**100.0**	14.4
Specificity	**100.0**	65.2	**100.0**	**100.0**	**100.0**	15.7
F1 score	**84.6**	64.7	53.7	82.4	80.0	24.8
100×						
	CNproScan	**CNOGpro**	**CNVnator**	**LUMPY**	**DELLY**	**PINDEL**
Accuracy	**93.3**	86.7	69.4	85.0	86.7	6.7
Sensitivity	86.7	73.3	43.3	70.0	73.3	**90.0**
Precision	**100.0**	**100.0**	86.7	**100.0**	**100.0**	3.3
Specificity	**100.0**	**100.0**	93.8	**100.0**	**100.0**	3.7
F1 score	**92.9**	84.6	57.8	82.4	84.6	6.4
200×						
	CNproScan	**CNOGpro**	**CNVnator**	**LUMPY**	**DELLY**	**PINDEL**
Accuracy	**95.1**	-	70.0	85.0	85.0	2.7
Sensitivity	**93.3**	-	46.7	70.0	70.0	90.0
Precision	96.6	-	87.5	**100.0**	**100.0**	1.3
Specificity	96.8	-	93.3	**100.0**	**100.0**	1.4
F1 score	**94.9**	-	60.9	82.4	82.4	2.5

Doubling the coverage to 200×, CNproScan detected 28 TP and 1 FP. The second closest was Pindel with 27 TP. The accuracy and sensitivity were the highest for CNproScan as the overall combined score.

Beginning with the 20× coverage, the CNproScan had the highest F1 score and Accuracy and kept it to 200×.

There is also Figure 5.10, where precision, recall, and F1 scores are plotted. It is visible how since reaching coverage 20×, the performance metrics for CNproScan are going up to the highest numbers.

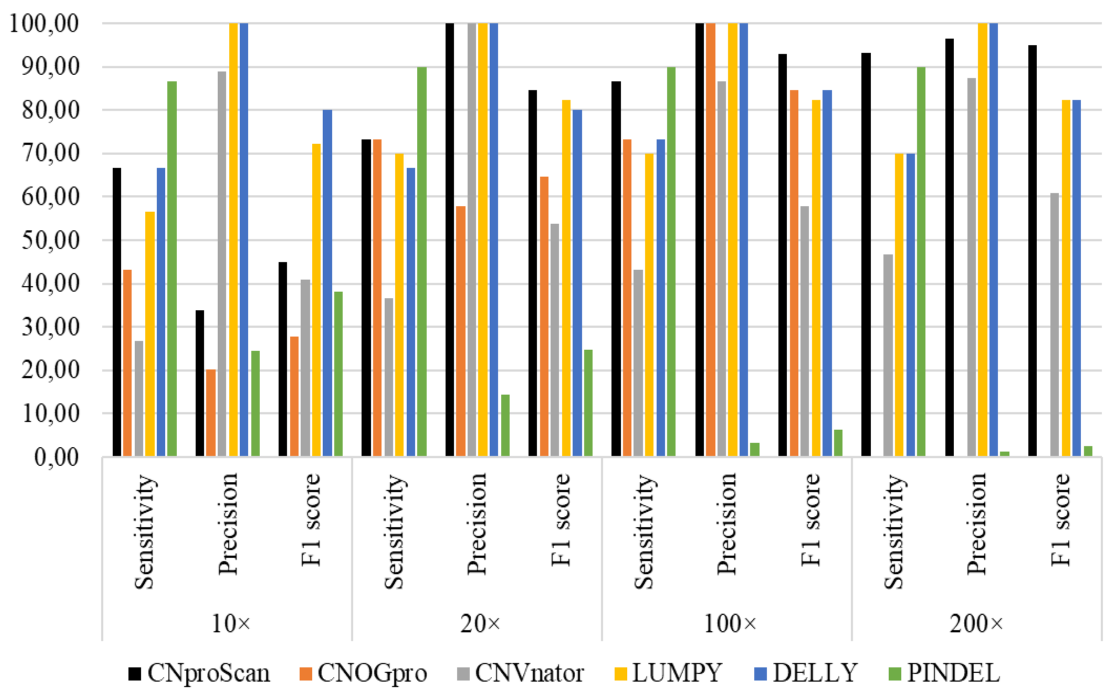


Figure 5.10 – Sensitivity, Precision and F1 scores of the simulated dataset

5.3.4 CNV length analysis

Next, I analyzed how tools dealt with various CNV lengths. The histogram depicting the tools' ability to detect various CNV lengths for 100× coverage is in Figure 5.11. The y-axis shows the count of CNVs detected within four defined bin sizes. The numbers of CNVs in each bin are shown in the brackets in the figure legend. Only CNproScan and Pindel detected the shortest CNVs (blue color). The CNV lengths are categorized into 4 bins: 0-25bp, 26-100bp, 101-1000bp, and 1001-4000bp.

The majority of tools coped perfectly with the longest CNVs (1001-4000bp). Only Pindel and cn.MOPS did not detect a 1302bp duplicated CNV. In the 101-1000bp bin, several tools struggled to detect all CNVs – namely cnv-seq, and cn.MOPS, CNVnator.

On the contrary, only 5 tools detected some CNVs from bin 26-100bp. CNproScan (3 out of 4) and Pindel (4 out of 4) detected the most CNVs. Others were CNOGpro, LUMPY, and DELLY. In the smallest CNVs under 25bp, only CNproScan (3 out of 6) and PINDEL (4 out of 6) detected any CNVs.

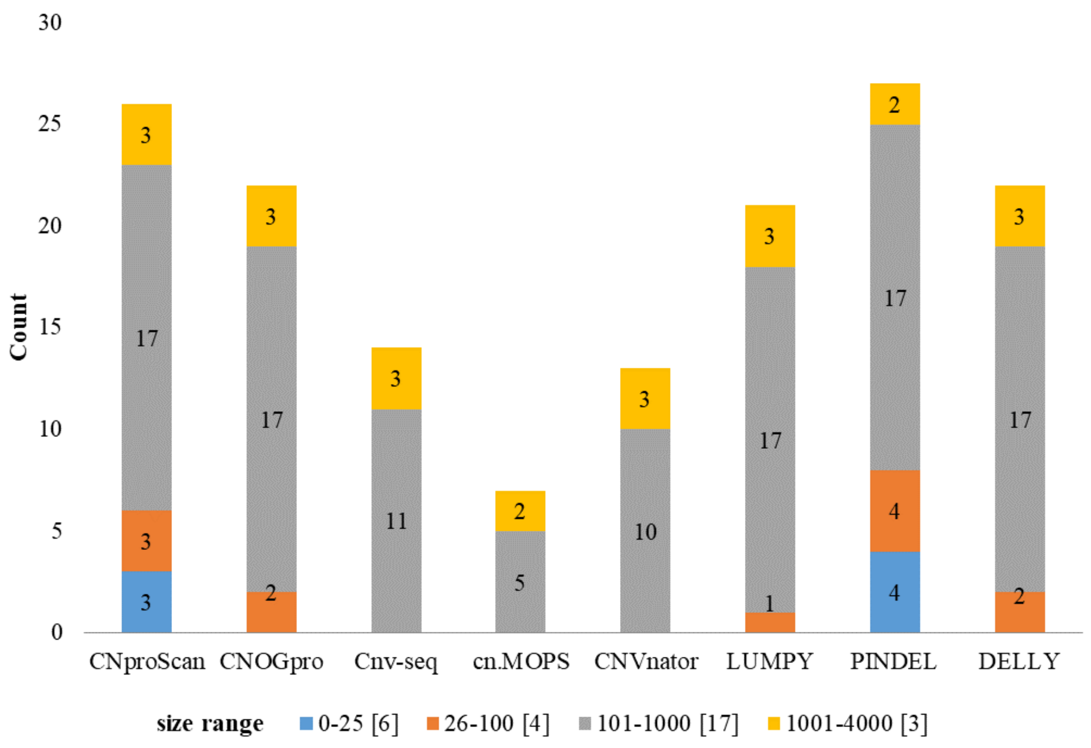


Figure 5.11 – CNV Size Histogram

5.4 R implementation - version 1.0

The initial development was done in MATLAB, but for wider usage, the method was rewritten also into the R with certain modifications. While the main methodology is the same, some changes were made over time that there are not present in the MATLAB version. Some of the changes originated from users posting certain issues with running the CNproScan while others were added to improve the usability, performance, and abilities of the tool. The changes are these:

- Parallelism – the genome is divided into chunks and the outliers are detected in each chunk separately. Using multiple threads parallelly decreased the computation time exponentially (Figure 5.12).
- Multi-chromosome support – samples with more than one chromosome are now supported.
- Removing tails from both ends before outliers detection – initially, only the zero coverage was removed, and then there was outliers detection. For further computation time gain, also the roughly estimated outliers from the upper tail (large peaks) are removed. So, outliers from both tails are removed now before the GESD. This step removes coverage values that would be confirmed by the GESD algorithm and were

just wasting computation space. These outliers are defined as 3 times IQR.

- Extending boundaries – the algorithm of slope changes was tweaked to avoid too extensive prolonging of the boundaries. I added the condition to stop the algorithm when the baseline is reached.
- Estimated CNV number – the copy number is reported just as the ratio between the maximum read-depth value of the CNV and the average coverage. This serves as a rough estimation and is based on the common assumption that copy number is linear with read depth.
- Duplication subtypes – initially, only the tandem or interspersed duplication types were reported but this was upgraded to direct and indirect cases for both duplication types. So, five CNV types can be detected now – deletion, direct and indirect tandem duplication, and direct and indirect interspersed duplication.

Installation is done using the 'devtools' R package from the GitHub repository https://github.com/robinjugas/CNproScan with this command:

```
devtools::install_github("robinjugas/CNproScan")
```

The CNV calling function is then called for example as:

```
DF <- CNproScanCNV(coverage_file, bam_file, fasta_file, GCnorm=TRUE,
MAPnorm=TRUE, ORICnorm=TRUE, bedgraph_file, oriCposition=1, cores=4)
```

5.4.1 Results and Discussion

The updated R version was compared with the original MATLAB version results. The updated R version deflected the sensitivity and specificity tradeoff more to the sensitivity side for the R version, mainly in the lower coverages. However, this is redeemed by lower specificity in all coverage values. This trade-off is present almost in all CNV detection tools when benchmarked [129]. The observed numbers are in Table 5.1. In 10× coverage, 6 more CNVs were detected, while 14 more FP CNVs were retained. For 20× coverage, 4 more CNVs were detected together with 5 more FP CNVs. The TP numbers for 100× and 200× coverages remained the same, however, 4 and 1 more FPs were detected in 100×, and 200× respectively. In 10× and 20× datasets, multiple shorter CNVs spanning the true longer CNV were noticed a few times. These are then counted as a single event if they overlap with any true CNV.

Table 5.12 – Observed results R/MATLAB

	CNproScan MATLAB				**CNproScan R**			
Coverage	**TP**	**TN**	**FP**	**FN**	**TP**	**TN**	**FP**	**FN**
10×	20	30	39	10	26	30	53	4
20×	22	30	0	8	26	30	13	4
100×	26	30	0	4	26	30	4	4
200×	28	30	1	2	28	30	2	2

For the previously published metrics, I added the precision and F1 score results to Table 5.13. The sensitivity is higher for the R version in the low coverages 10× and 20×. But the specificity was negatively affected, the most for the 20×. It seems that this coverage is affected heavily by increasing TP by only 4 while the FP increased by 13. Other higher coverages remain more stable.

Table 5.13 – Performance metrics R/MATLAB

	CNproScan MATLAB				
Coverage	**Sensitivity/ Recall**	**Specificity**	**Accuracy**	**Precision**	**F1 score**
10×	66.67	43.48	50.51	33.90	44.94
20×	73.33	100.00	86.67	100.00	84.61
100×	86.67	100.00	93.33	100.00	92.85
200×	93.33	96.77	95.08	96.55	94.91
	CNproScan R				
10×	86.66	36.14	49.55	32.91	47.70
20×	86.66	69.76	76.71	66.66	75.36
100×	86.66	88.23	87.50	86.66	86.66
200×	93.33	93.75	93.54	93.33	93.33

5.4.2 Overlap analysis

I evaluated the overlaps between the dataset CNV start and stop coordinates and the detected ones. Both the inside overlap and the number of bases that are out of the borders are evaluated in percentage points (see Table 5.14)

The smallest percentage of overlap is 24% for CNV number 19. In the majority of cases, the overlap is over 90%. The outside columns in Table 5.14 represent the percent of bases relative to the CNV length outside of the true borders. For CNV number 1, which is only a small 4 bp deletion, a larger segment of approximately 25 bp was detected. Also, a few times, shorter CNVs spanning the true CNV were captured. These are then merged and overlap as merged. This is more common in the lower coverage and absent in 100× coverage and higher. In the majority of cases, the outside portion is small. Only in 4 cases is the outside portion multiple times higher. See the CNVs number 1, 19, 22, 28. Three of them are deletions. The only duplication is only one time of the original length.

Table 5.14 – Boundaries analysis

			10× [%]		**20× [%]**		**100× [%]**		**200× [%]**	
#	**Length**	**Copy number**	**Inside**	**Outside**	**Inside**	**Outside**	**Inside**	**Outside**	**Inside**	**Outside**
1	4	0	100	550	100	550	100	550	100	550
2	23	2	-	-	-	-	-	-	48	100
3	8	0		-	-	-	-	-	100	275
4	63	4	100	38	100	38	100	38	100	38
5	270	3	100	6	100	6	100	8	100	8
6	828	0	100	3	100	3	100	4	100	7
7	81	4	100	32	100	32	100	32	100	32
8	276	2	44	0	45	6	100	8	100	8
9	210	3	100	8	99	10	100	10	100	10
10	612	0	100	4	99	4	100	6	100	8
11	257	2	50	0	80	0	99	8	100	9
12	216	3	100	12	100	12	100	12	100	12
13	134	0	96	19	96	19	96	28	100	29
14	354	4	98	8	99	8	99	8	99	8
15	1302	2	75	0	98	0	100	2	100	2
16	522	0	98	5	98	5	98	7	100	6
17	3776	3	100	1	100	1	100	1	100	1
18	264	2	50	0	69	0	96	13	96	13
19	17	0	24	194	24	194	24	206	24	206
20	4	2	-	-	-	-	-	-	-	-
21	17	0	-	-	-	-	-	-	-	-
22	26	0	42	142	42	142	42	185	85	162
23	1677	0	99	2	99	2	100	3	100	3
24	516	0	97	7	97	7	99	9	100	8
25	654	0	98	6	98	6	98	8	98	8
26	893	2	95	4	95	4	98	4	98	4
27	516	3	97	8	97	8	97	8	97	8
28	37	3	49	111	49	111	49	111	49	111
29	501	2	36	0	35	0	96	8	96	9
30	660	2	95	6	97	6	97	7	97	7

5.4.3 Copy-number analysis

The detected copy numbers were compared to the true copy numbers in the artificial genomes. The results are in Table 5.15, with false detections highlighted by shadow. In two cases of deletions, number 1 and 19, which are both very short deletions, are these detected correctly as deletions but with a false copy number of 1. This was likely caused by coverage spanning these CNVs not low enough to be later rounded to zero. In duplicated CNVs number 4, 5, 7, 14, and 28, are all their copy numbers underestimated by 1. The average accuracy of correct copy number estimation for all coverages is about 75 %.

Table 5.15 – Detected copy numbers

#	Length	Copy number	10×	20×	100×	200×
1	4	0	1	1	1	1
2	23	2	-	-	-	1
3	8	0	-	-	-	1
4	63	4	3	3	3	3
5	270	3	2	2	2	3
6	828	0	0	0	0	0
7	81	4	3	3	3	3
8	276	2	2	2	2	2
9	210	3	3	3	3	3
10	612	0	0	0	0	0
11	257	2	2	2	2	2
12	216	3	3	3	3	3
13	134	0	0	0	0	0
14	354	4	3	3	4	4
15	1302	2	2	2	2	2
16	522	0	0	0	0	0
17	3776	3	3	3	3	3
18	264	2	2	2	2	2
19	17	0	0	1	1	0
20	4	2	-	-	-	-
21	17	0	-	-	-	-
22	26	0	0	0	0	0
23	1677	0	0	0	0	0
24	516	0	0	0	0	0
25	654	0	0	0	0	0
26	893	2	2	2	2	2
27	516	3	3	3	3	3
28	37	3	2	2	2	2
29	501	2	2	2	2	2
30	660	2	3	2	2	2
Valid estimation			19	19	20	22
Accuracy			73.07	73.07	76.92	78.57

5.4.4 Runtime analysis

I analyze the runtime of the main function with all normalizations turned on and off. The function was tested on AMD Ryzen 5600G (3,9 GHz) and 64GB RAM running Ubuntu 22.04. A single sample of *K. pneumoniae* was tested with a different number of cores. Regarding the RAM usage, the memory consumption didn't exceed 5 GB of RAM throughout the testing measured as the whole R session memory consumption.

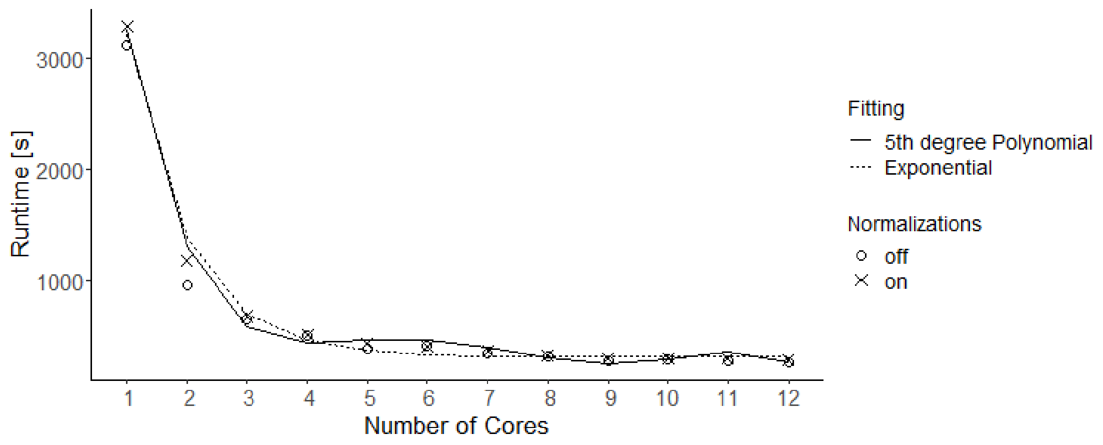


Figure 5.12 – CNproScan runtime analysis

I tested 1 to 12 cores and plot them in Figure 5.12. Observing the runtime analysis, the runtime decreases approximately exponentially with a number of cores, which is also proven by the exponential curve fitting (R^2 error 0.9921). Alternatively, a fifth-degree polynomial was fitted with R^2 error 0.9944.

Importantly, increasing the number of cores beyond 4 does not provide further runtime improvement reaching the running time of approximately 8 minutes. On the other side, unintended performance drawbacks could happen with increasing core numbers based on the fact that dividing the genome into smaller and smaller chunks can have an impact on outlier detection relying on descriptive statistics. However, this impact was present only at a differing number of detected CNVs but the difference was no larger than a few CNVs.

Also, the normalization part of CNV detection does not have a significant impact on running time and the observed difference is not larger than three minutes at maximum. It is necessary to mention, that runtime is dependent on the overall read-depth signal and the number of outliers and peaks. More flat read-depth signals will have reduced computation times and also a low amount of detected CNVs.

5.5 Summary

The chapter presented a developed algorithm called CNproScan for CNV detection in prokaryotic genomes. It is a hybrid method combining the read depth signal with support from pair read features. The method achieved overall better performance compared to other presented tools. It had the highest accuracy and F1 scores beginning with coverage 20×. CNproScan detected very small CNVs and belong with Pindel to the only two tools able to capture them. Also, contrary to the expectation stated in the literature, the achieved resolution of boundaries detection from the read-depth profile was acceptable. Furthermore, it brings the whole spectrum of auxiliary normalizations, such as replication origin bias and circular genome correction, for handling the task specifically for bacterial genomes.

So far, the algorithm was tested only on the artificial dataset. The results of real sequencing data are part of the next chapter.

Also, the updated R version was presented reaching higher sensitivity but slightly decreased specificity. However, this tradeoff is common in many CNV detection tools. With this sensitivity bonus, it was able to perform very well in the low coverage of 10×. The algorithm is available as the R package with documentation.

6 PROCARYASV

6.1 Merging of the detection tools

Such as the hybrid method removes the limits of a single approach, the integration of multiple detection tools limits their weakness and improves performance. This issue is tied to the topic of merging structural variants.

The topic of merging variant callers is more advanced in the field of SNP or SNV variant calling, where various tools are already being successfully merged based on their performance, e.g., using machine learning methods [192].

In the fields of SVs or CNVs, the problem of overlaps arises. The parameters of the minimal overlap and the type of overlap (equal, within, etc.) can be both user-defined or hard-coded. However, these parameters are usually defined somehow arbitrarily.

What reliable results are is the question to ask. Generally, the union or the intersection of results is the most common approach. It depends on the preference for higher sensitivity or specificity. Most effortlessly, the reliable results are those given by the most tools. Then, the threshold of how many tools are the most has to be set. On the other hand, rare events could be omitted. Alternatively, the union approach likely produces a high rate of false positives. A weighted approach can be applied if performance metrics are known. But for accurate performance metrics, you need a valid ground truth set, ideally validated by sequencing methods. [141]

6.2 Pipeline Design

I decided to create a CNV/SV calling pipeline based on the Snakemake framework [257]. It is a Python-based workflow management system for reproducible and scalable analysis. It consists of so-called Snakemake rules which define the inputs and outputs of a given rule. The rule serves to call a certain function, tool, package, etc. Parameters that are necessary for the called tool can be specified too inside the rule or can be adopted from the external configuration file. Scalability parameters can be defined too, such as the number of threads or memory requirements. The possibilities are multiple.

I called the pipeline ProcaryaSV denoting the focus on prokaryotic genomes. It is based on commonly used CNV and SV detection tools and state-of-the-art processes for manipulating sequencing data. All the necessary specifical inputs for each SV/CNV caller are processed as described by the caller's manuals. In some cases, I used or further modify the Snakemake Wrappers repository where the finished easy-to-use rules and wrappers (small Python scripts calling the tools) are available.

The overall simplified workflow is in Figure 6.1. Only the basic tools are pictured, without raw reads quality check or the optional trimming parts.

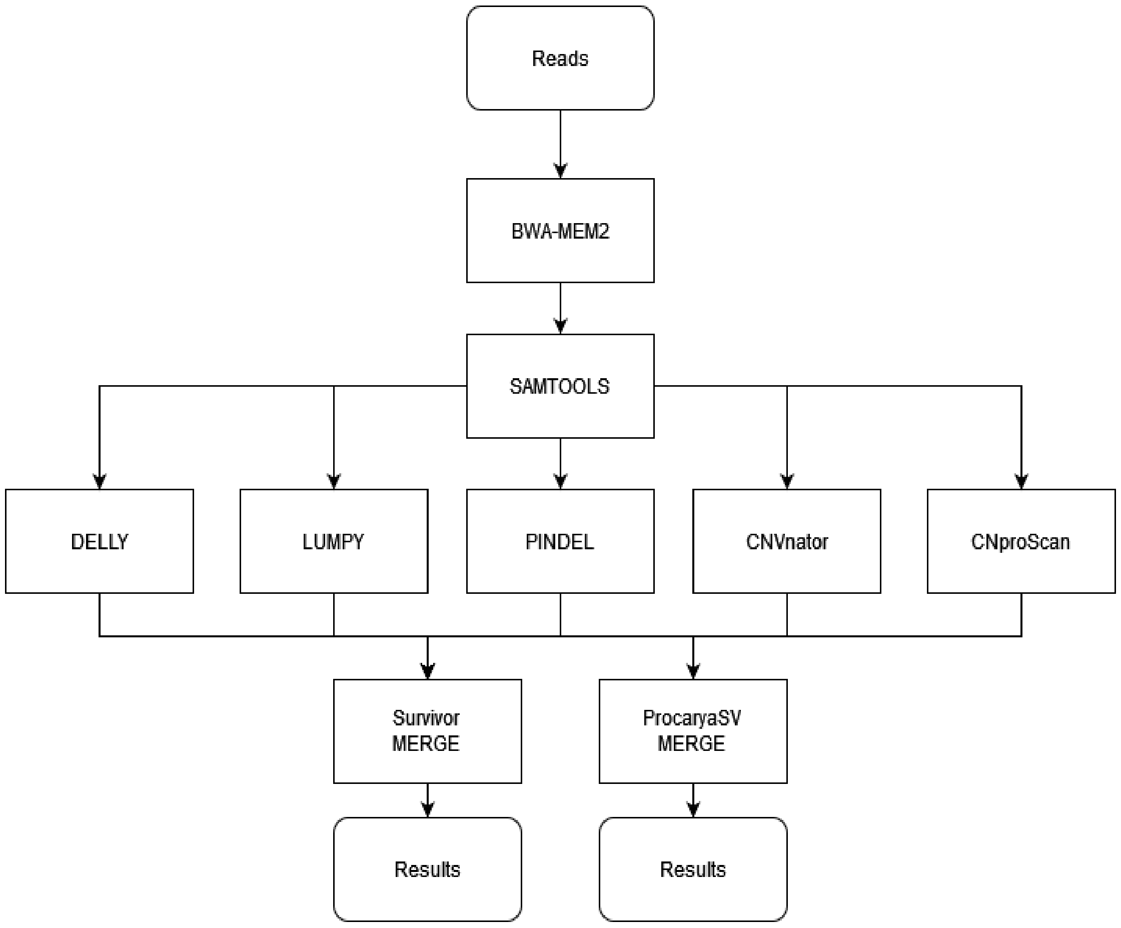


Figure 6.1 – ProcaryaSV workflow

Two CNV callers and three SV callers were used. One of the requirements was that the tools has to be placed in some conda repository to be easily installed by Snakemake. The only exception is the CNproScan which is available only in GitHub so far. Also, tools mustn't require too obscure inputs, e.g. from some deprecated packages. Most importantly, tools have to be suitable for haploid prokaryotic genomes. Thus, the CNproScan, CNVnator, LUMPY, DELLY2, and Pindel were selected. The overview of all the tools used (excluding the tool's dependencies) is in Supplementary Table 5.

The ProcaryaSV is available from the GitHub repository and was tested on the version mentioned in Table 6.1.

Table 6.1 – ProcaryaSV GitHub repository

Repository	**Version**
https://github.com/robinjugas/ProcaryaSV	1.0

It is necessary to fill correctly the YAML configuration file and run the pipeline with the snakemake commands inside bash:

```
WORK_DIR=" path to the working directory"
WORKFLOW_DIR=" path to the Snakefile folder"

snakemake --cores INT --snakefile $WORKFLOW_DIR/Snakefile --directory
$WORK_DIR --configfile config.yaml --use-conda
```

6.3 Merging algorithm

The two main inputs to merging SVs are merging BED files, using the bedtools [196], or merging the VCF files, using, for example, SURVIVOR (StructURal Variant majorIty VOte) [194] or SVDB [195]. The VCF files are not fully suitable for recording large-scale SVs. Foremost, there is only one genomic coordinate column POS, which is used to store SV's start coordinate, but the ending coordinate has to be written into the INFO column. Also, the field name is tool specific. This requires parsing of the column. Furthermore, some callers write the DNA sequence of a given variant into the ALT column, making it hard to read.

I tried both SURVIVOR and SVDB to merge the resulting VCF files. The SVDB failed completely resulting in non-readable files because it writes the genomic sequence of the event into the file. The SURVIVOR did better compared to SVDB and thus was kept in the pipeline for user comparison.

The merging algorithm of ProcaryaSV parses the VCF outputs for all callers and respects their specifics. It separates four categories of SV calls: deletions, duplications, inversions, and insertions, and merges them separately. Insertions and inversions are called only by Pindel and Delly2, while deletions and duplications are called by all of them.

Here, I present my own approach to merging SVs based on cumulating binary vectors and then thresholding them. The user can define the value of the threshold by his or her preference. The input is VCF files from callers. The results are formatted as a TSV (tab-separated values) file, which can be imported into any spreadsheet application. The parameters of minimum and maximum SV length are to be set. All SVs not fitting into these are deleted.

For every type of SV detected (DEL, DUP, INV, INS) the simple binary vector is created for each caller separately and then these are summed up (see illustration in Figure 6.2). This means that a region called by two callers will have a value of two spanning the region where these callers overlap. This summed vector is processed so that small gaps are filled and merged together. This gap is an optional parameter, but a value of 100 bp is the default.

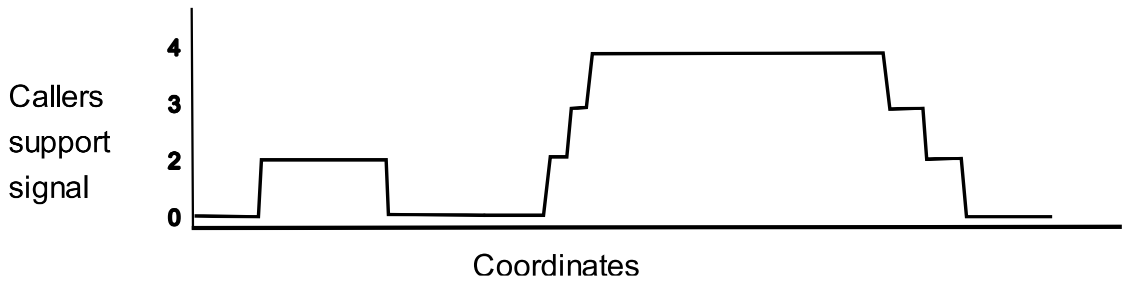


Figure 6.2 – Caller's support signal of two structural variations. First one with a support of two callers, second one with a support of four callers. The steps are created by reported different start and stop coordinates.

In the first iteration, all levels of callers support are outputted except those under the value of the user-defined caller's threshold. The regions called the most times are outputted first and then it proceeds to a lower number of callers. This also means that certain regions can be reported multiple times, once as a shorter region of higher support, and later as a longer region of lower support. The number of callers that called the regions is reported.

In the second iteration, the events are searched for overlaps with the use of the Iranges package function findOverlaps. The important parameter here is the maxgap, meaning the maximum allowed distance between the start and end coordinates. This step is to collapse overlapping events into one. The one event is reported with corresponding values and coordinates with maximum support are saved. The diagram of merging is in Figure 6.3. The threshold represents the maxgap parameter and is applied to all firstly reported events. If the condition was not met, ID #3 would be reported separately.

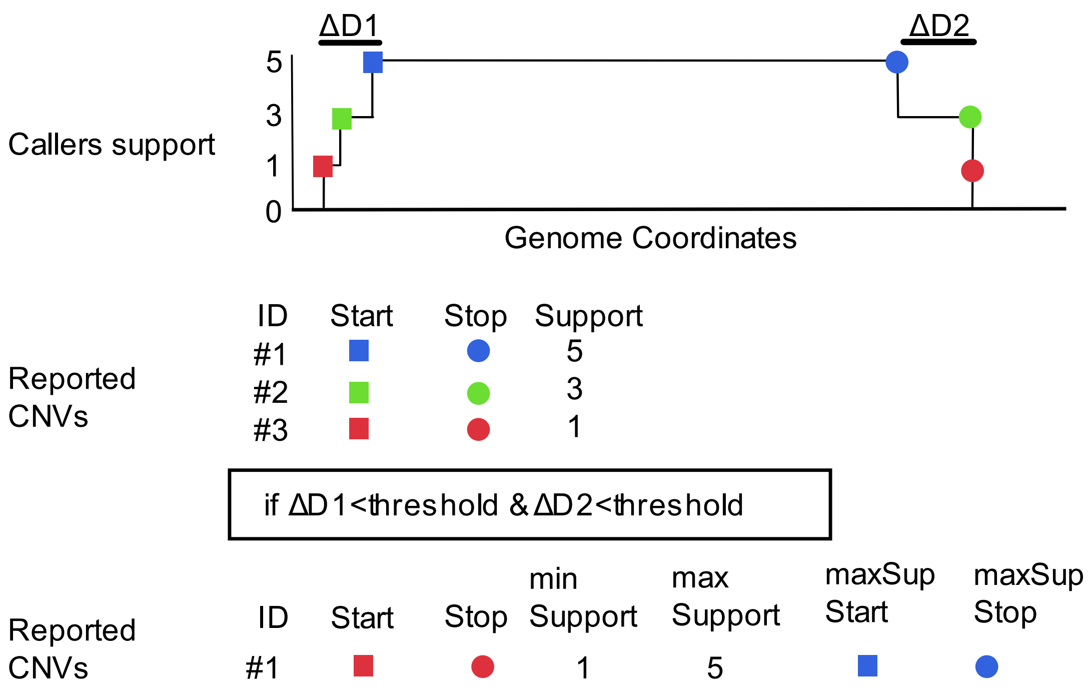


Figure 6.3 – SV merging diagram. The colored objects refer to detected SV’s coordinates.

After this, the reported CNVs are searched for overlaps again, to report potential cases where multiple shorter CNVs overlap a single long CNV. This information is stored in the output file.

In the last iteration, all events are backtracked. The information about the number of underlying events and percentual coverage by each caller is recorded and saved. A region can be supported by multiple callers, but they can contribute as multiple separately reported events merged because of the merging step.

The tabular separated file (.tsv) is the output together with informative graphs. These are the Venn diagram of callers and pie plot of different SV types' abundances.

6.4 Benchmarking on the simulated data

The merging algorithm of ProcaryaSV was benchmarked on the previous artificial dataset. The various values of minimum callers support (MinCallers) were used to decide the optimal value. The threshold is inclusive, the operator ‘>=’ is used.

Table 6.2 – ProcaryaSV merging performance metrics

10×					
MinCallers threshold	**1**	**2**	**3**	**4**	**5**
Accuracy	36.3	**90.0**	85.0	73.3	56.7
Sensitivity	90.0	80.0	70.0	46.7	13.3
Precision	21.8	100.0	100.0	100.0	100.0
Specificity	23.6	100.0	100.0	100.0	100.0
F1 score	35.1	**88.9**	82.4	63.6	23.5
20×					
MinCallers threshold	**1**	**2**	**3**	**4**	**5**
Accuracy	37.0	**90.0**	83.3	83.3	58.3
Sensitivity	90.0	80.0	66.7	66.7	16.7
Precision	22.3	100.0	100.0	100.0	100.0
Specificity	24.2	100.0	100.0	100.0	100.0
F1 score	35.8	**88.9**	80.0	80.0	28.6
100×					
MinCallers threshold	**1**	**2**	**3**	**4**	**5**
Accuracy	13.1	**87.3**	85.0	85.0	61.7
Sensitivity	90.0	83.3	70.0	70.0	23.3
Precision	6.7	89.3	100.0	100.0	100.0
Specificity	7.4	90.9	100.0	100.0	100.0
F1 score	12.5	**86.2**	82.4	82.4	37.8
200×					
MinCallers threshold	**1**	**2**	**3**	**4**	**5**
Accuracy	7.0	**88.9**	85.0	85.0	61.7
Sensitivity	93.3	86.7	70.0	70.0	23.3
Precision	3.5	89.7	100.0	100.0	100.0
Specificity	3.8	90.9	100.0	100.0	100.0
F1 score	6.8	**88.1**	82.4	82.4	37.8

All performance metrics are in Table 6.2, the maximum F1 scores are in bold. The maximum F1 scores are all achieved when the MinCallers threshold is set to 2. The precision-recall curves for all coverage levels are in Figure 6.4. The precision remains the same from the MinCallers threshold set to 2 (higher coverage) or 3 (lower coverage), while a threshold lower than 2 brings a lot of false positives. Recall (sensitivity) decreases to very low numbers, omitting many true positives. Following previous results, setting the minimal callers threshold to 2 is the optimal setting to balance precision and recall.

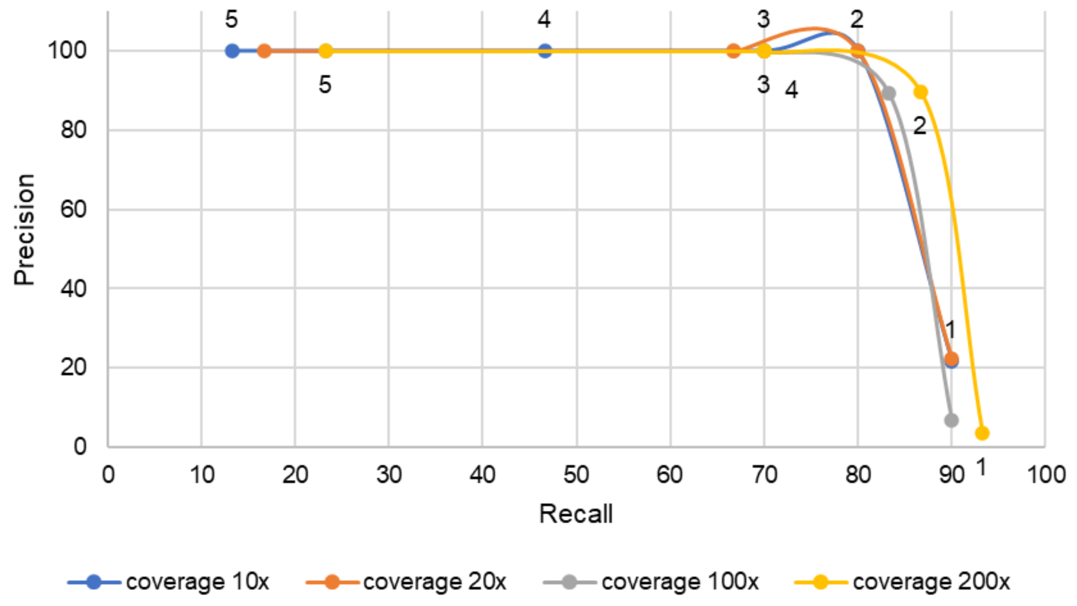


Figure 6.4 – Precision-recall curve for MinCallers threshold parameter. Digits near lines denote the parameter value

For comparison, in Table 6.3 there is an evaluation of the SURVIVOR merging algorithm. The numbers from the previous table are added for easier comparison. The SURVIVOR merge command settings were set to minimal callers 2, maximum allowed distance 1000, and minimal considered SV length 1. Looking at the results, they are almost the same looking at the same settings of minimal callers 2 for both algorithms. The ProcaryaSV had higher F1 scores for 20× and higher coverages by a few points.

This is expected based on the description of the SURVIVOR merging method. In SURVIVOR, two SVs are defined as overlapping if their start and stop coordinates are within 1 kb and of the same SV class.

Table 6.3 – SURVIVOR and ProcaryaSV performance metrics for minCallers of 2

	Coverage	Accuracy	Sensitivity	Precision	Specificity	F1 score
SURVIVOR	**10×**	90.0	80.0	100.0	100.0	88.9
	20×	85.0	70.0	100.0	100.0	82.4
	100×	87.1	80.0	92.3	93.8	85.7
	200×	85.5	76.7	92.0	93.8	83.6
ProcaryaSV	**10×**	90.0	80.0	100.0	100.0	88.9
	20×	**90.0**	80.0	100.0	100.0	**88.9**
	100×	**87.3**	83.3	89.3	90.9	**86.2**
	200×	**88.9**	86.7	89.7	90.9	**88.1**

6.5 Benchmarking on the real data

While the CNproScan chapter included only the results for the simulated dataset, the results for the real dataset are put here as they are obtained with the use of the ProcaryaSV pipeline. The real dataset consists of multiple samples from four Bacteria organisms as already reported previously in the chapter on CNproScan design.

Table 6.4 – Overview of used real datasets, reference genomes used for alignment, and average coverage across the samples

Organism	**Accessory ID**	**No. of samples**	**Reference Accession**	**DoriC ID & position**	**Average coverage**
Staphylococcus aureus	PRJNA497094	92	NC_007793	ORI10010181 1906 nt	350.2×
Escherichia coli	DRA005229	58	AP012306	ORI10020002 3923657 nt	330.3×
Lactobacillus casei	PRJNA342061	50	NC_014334	ORI94010712 1351 nt	189.7×
Klebsiella pneumoniae	PRJNA515630	48	NC_012731	ORI93020089 5248419 nt	67.9×

These data were downloaded from public repositories and quality checked. Only the reads of *Klebsiella pneumoniae* were taken from previous cooperation with University Hospital Brno. The other three datasets were already trimmed from adapters and low-quality 3' ends.

The results mentioned here are described differently compared to the artificial dataset since no apriori known CNVs are known in these samples. Thus, the results are described rather descriptively. Although the pipeline enables to detect the insertions and inversions, only CNVs were evaluated in the results as all five tools can detect them. Looking at all SV types, no insertions were generally called. A portion of inversions was called overwhelmingly by Pindel. The multitudes of them are smaller than 100 bp.

Most importantly, the results are already processed by the ProcaryaSV merge algorithm, so CNVs are overlapped across the callers and merged. The following term CNVs then should be understood as CNV regions with various levels of support by callers. The CNV amounts do not represent the raw outputs of the callers, but the post-processed regions. In each subchapter, I evaluate the CNproScan preference regarding CNV size and CNV type. Also, the amount of overlap with other tools.

6.5.1 Results - *K. pneumoniae*

The results of all 48 *Klebsiella* samples are merged and analyzed together into over 15000 CNVs. The average coverage of *Klebsiella* samples was the lowest across the real datasets. In Figure 6.5 there is a mean read-depth signal calculated from all the samples. Notice that there are a lot of deviations from the baseline, with multiple deletions reaching the zero level. This points out the presence of some CNVs or potential false positives. Furthermore, it denotes there are many shared CNVs across the samples.

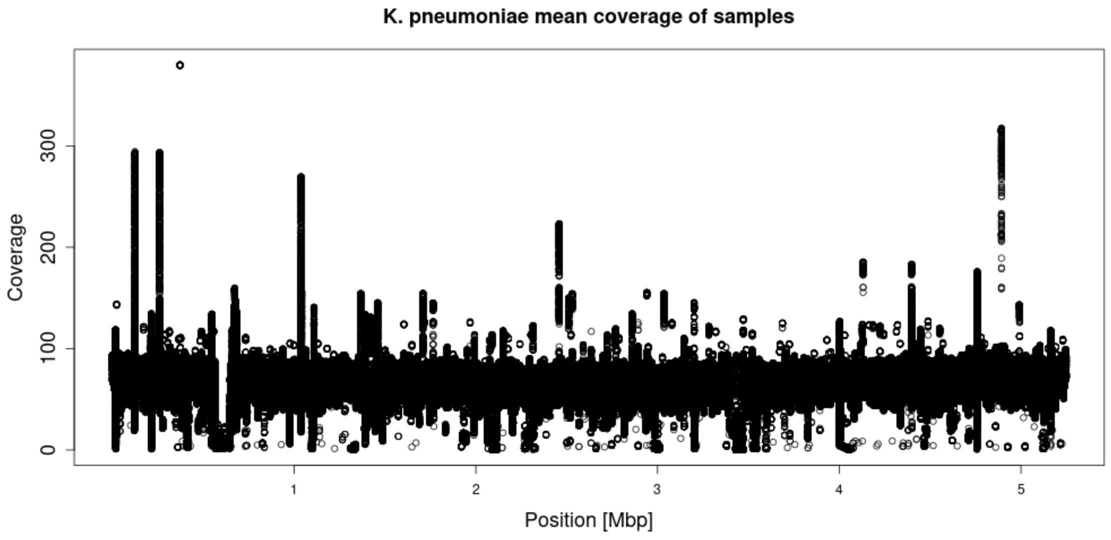


Figure 6.5 – Mean read-depth signal across all samples for *K. pneumoniae*

The frequent way to display overlaps is a Venn diagram, constructed in Figure 6.6, to show callers' representation in the final results. The highest number of unique events were detected by CNproScan (2159 CNVs) and Pindel (1928 CNVs). Venn diagram shows that CNproScan partakes in the majority of events. It shares a high amount of events with read-depth based CNVnator as with hybrid DELLY2 and LUMPY. 207 CNVs were called by all five tools. Approximately 2300 CNVs were called by a combination of four tools, 4700 CNVs were called by three tools, and 3900 by two tools.

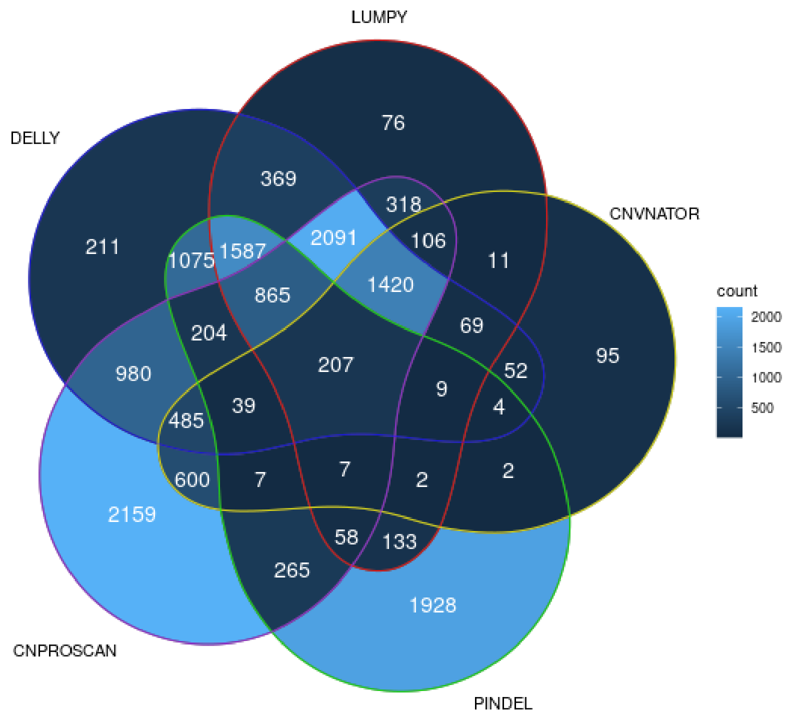


Figure 6.6 – Venn diagram of detected CNVs for *K. pneumonaie* dataset

Next, I evaluated the representation of SV sizes, SV types, and callers that detected them. These figures are displayed as so-called treemaps representing the proportions by the size of each box.

The representation of CNV types by size is in Figure 6.7. The short CNVs under 100 bp are easier to detect as deletions, thus, they are more abundant. Contrary, the duplications are present predominantly in the shorter events up to 1000 bp. Other than that, there is a domination of deletions with increasing CNV length.

The caller's preference for CNV lengths is in Figure 6.8. The abbreviations for boxes with less than 200 CNVs are: L is for LUMPY, CS – CNproScan, CN – CNVnator, P – Pindel, and D – DELLY2. The boxes are sorted by their share. The callers are represented rather equally except for Pindel, which is represented visibly less in other than the 100bp size range.

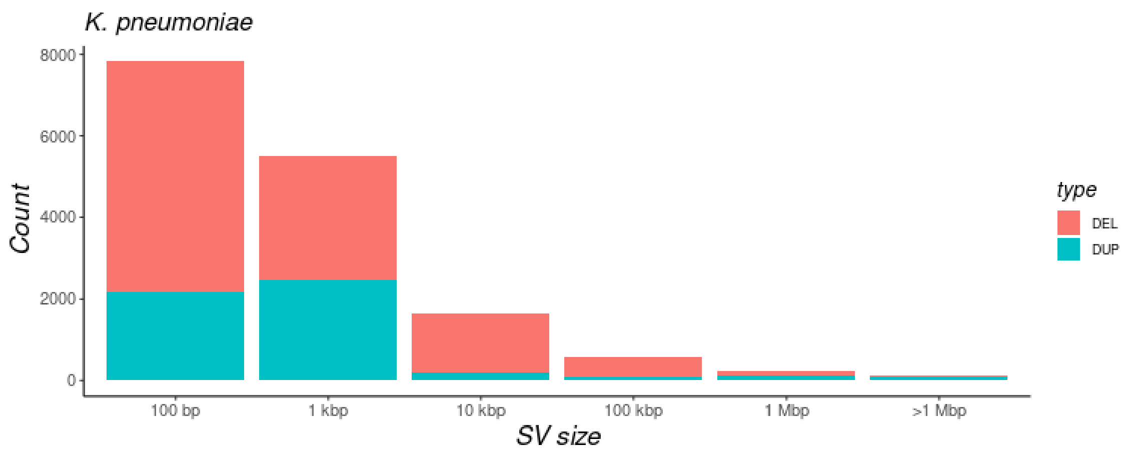


Figure 6.7 – CNV sizes by type for *K. pneumoniae* dataset

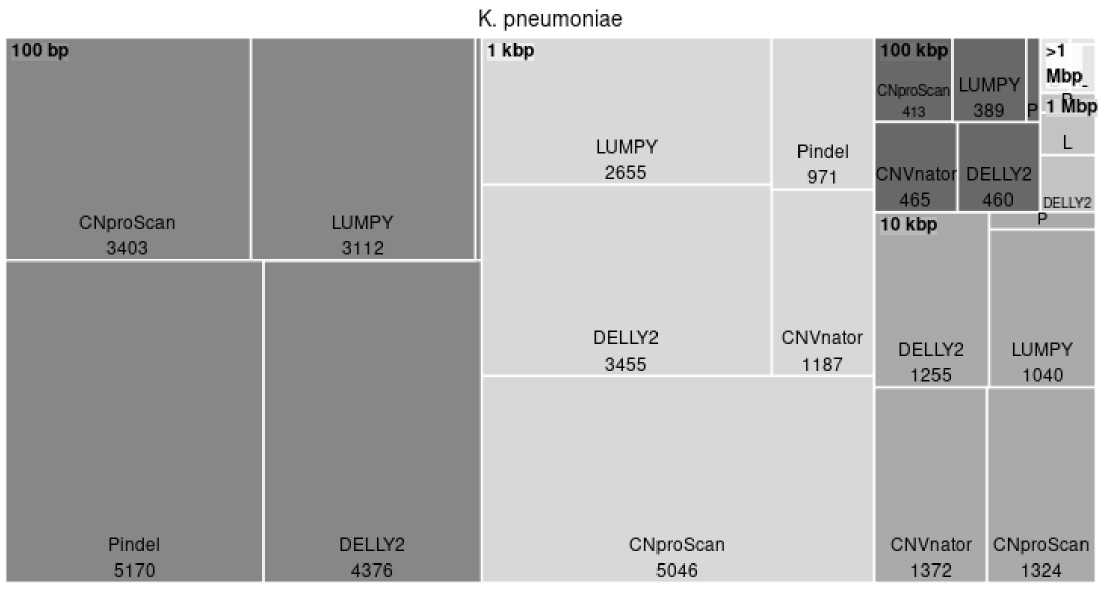


Figure 6.8 – CNV sizes by callers for *K. pneumoniae* dataset

That the larger part of CNVs is deletions can be seen again in Figure 6.9. In deletions, all callers are represented approximately equally, while for duplications the CNVnator is almost absent. CNproScan detected a large portion of both deletions and duplications.

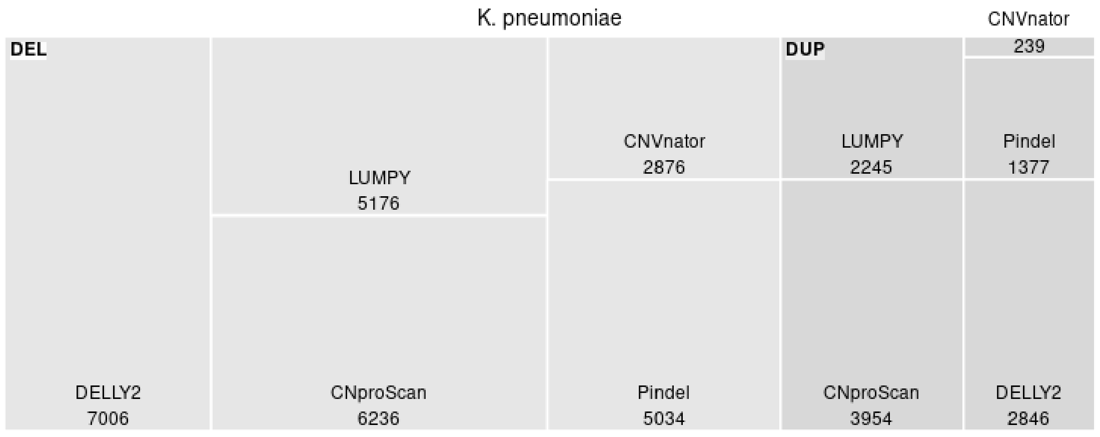


Figure 6.9 – CNV types by caller for *K. pneumoniae* dataset

6.5.2 Results - *E. coli*

The average coverage of *E. coli* samples is 330× and is much higher than for Klebsiella samples. Also, the read-depth signal is rather noisy, as can be seen in averaged read-depth signal in Figure 6.10. Only two short deletions touch the zero-coverage level. This could be caused by the averaging of the signals, and it would denote that the *E. coli* samples are very different from each other having often unique CNVs. Also, the *E. coli* had high levels of replication origin bias (Spearman's Rho -0.97).

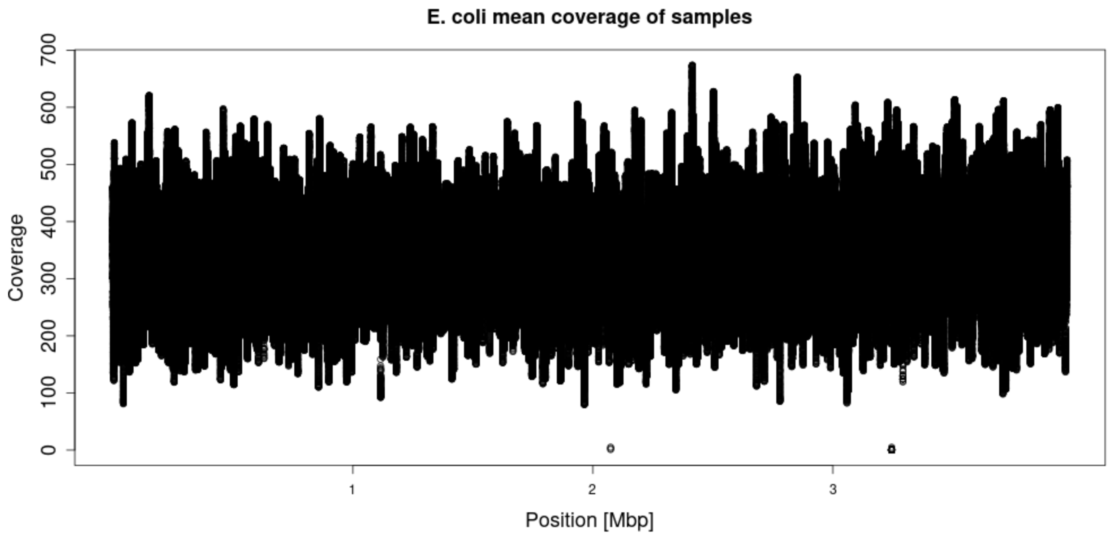


Figure 6.10 – Mean read-depth signal across all samples for *E. coli*

The Venn diagram is in Figure 6.11. For *E. coli* samples, Pindel detected a large number of unique CNVs. There is a very large overlap between Pindel and DELLY, 12000 CNVs. Regarding overlapping events, only 4 CNVs were called by all tools, over 800 were called by four tools, 700 by three tools, and over 14000 by two tools.

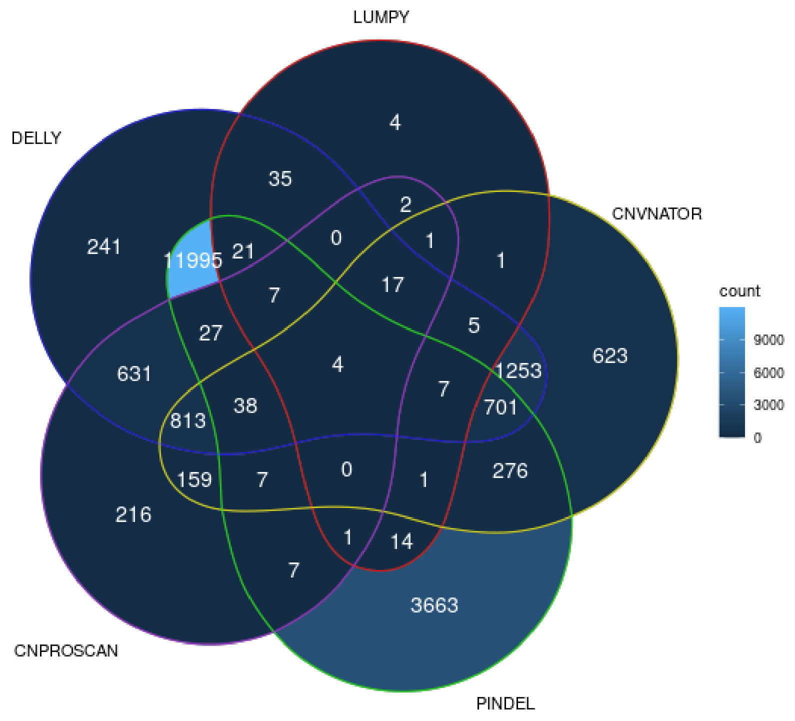


Figure 6.11 – Venn diagram for *E. coli* dataset

A slight majority of cases under 100 bp were deletions, but with the increasing CNV lengths up to 1 kbp and 10 kbp, duplications were more prevalent. Combined, the duplications and deletions are equal in numbers as can be seen in Figure 6.12. Interestingly, the CNproScan captured only a smaller portion of deletions across all CNV lengths. This shows that the majority of deletion was detected by read-pair and other than read-depth approaches.

See Supplementary Figures F1-F2 for plots of SV lengths on CNV types and CNV size preference by callers. Regarding the CNV size, the majority of CNVs under 100 bp were called by Pindel and DELLY2. The rates were more equal for longer CNV ranges.

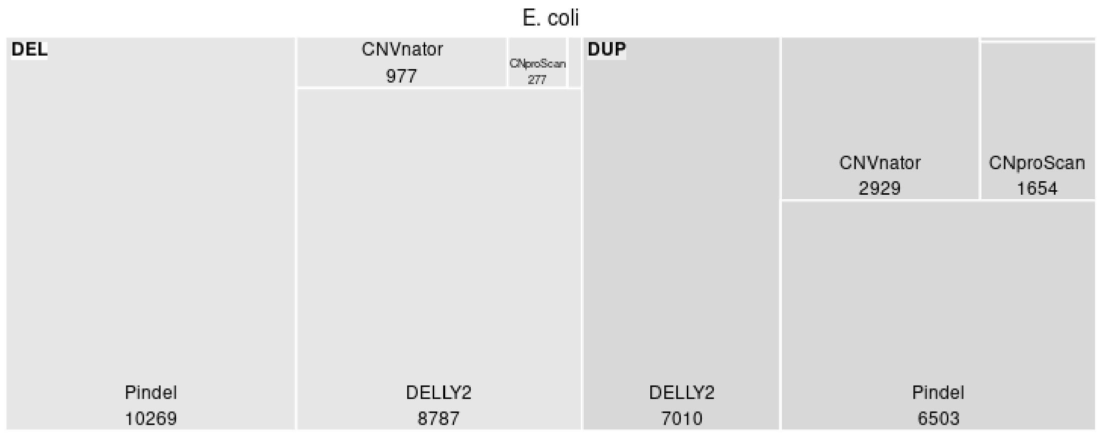


Figure 6.12 – CNV types by callers for *E. coli* dataset

6.5.3 Results - *S. aureus*

The read-depth profile of *S. aureus* in Figure 6.13 was completely different from the previous ones. The baseline is clear without the 'oscillations' visible in *E. coli*. The CNVs are visible. Generally, we would not anticipate a high amount of CNVs based on a read-depth profile.

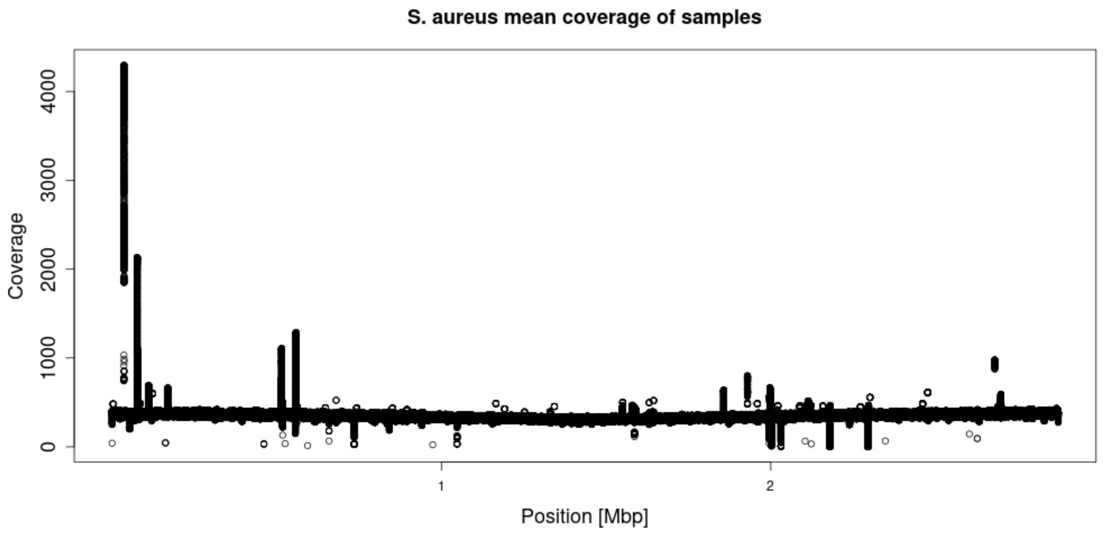


Figure 6.13 – Mean read-depth signal across all samples for *S. aureus*

The Venn diagram is in Figure 6.14. In the amount of the total CNVs detected, Pindel uniquely detected highly more CNVs than the rest combined. The second was DELLY2 which detected over 6000 unique SVs. Regarding overlapping events, 304 CNVs were called by a combination of five tools, about 2000 by four tools, over 5000 by three tools, and around 18000 by two tools.

Deletions were the dominant of all CNV sizes. DELLY2, LUMPY, and Pindel detected very high numbers of deletions. The numbers of duplications were more comparable except for Pindel. The proportions of callers are in Figure 6.15.

See Supplementary Figures G1-G2 for plots of CNV lengths on CNV types and CNV size preference by callers. The majority of events fit into the smallest up to 100bp or up to 1000 bp size range. Pindel dominated the 100 bp range with DELLY2 being the second one.

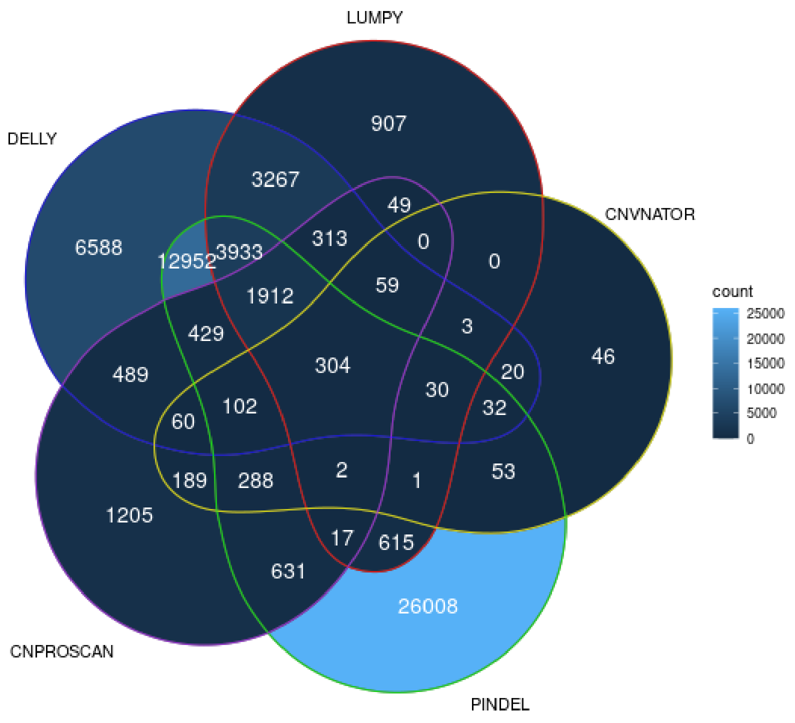


Figure 6.14 – Venn diagram for *S. aureus* dataset

Contrary to the expectation of not many CNVs being present, there was the highest number of CNVs detected. Predominantly detected by Pindel and DELLY. Similarly to previous *E. coli*, the non-read-depth methods detected the most events. However, there might be a large number of false positives.

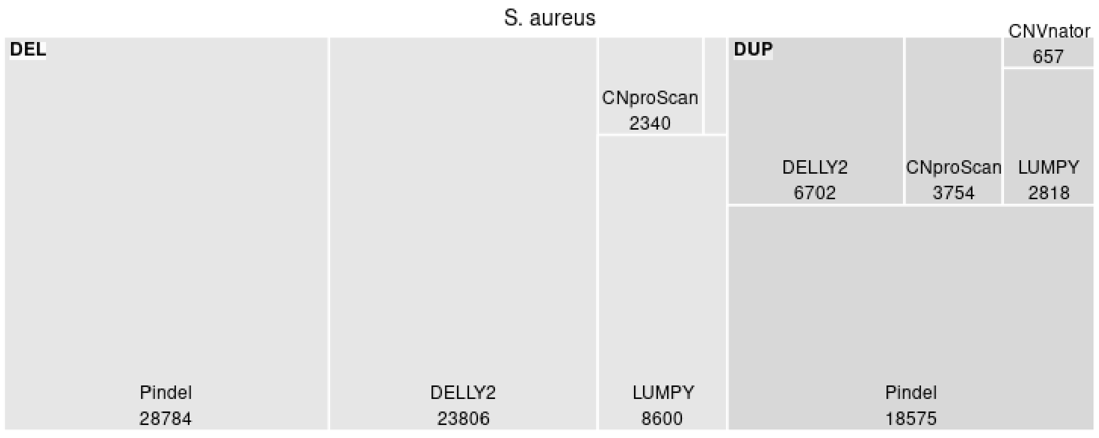


Figure 6.15 – CNV types by caller for *S. aureus* dataset

6.5.4 Results - *L. casei*

The read-depth signal of *L. casei* samples is very distinctive (Figure 6.16). The origin of replication bias can be easily spotted as the V-shaped trend. The Spearman's Rho was -0.93. The read-depth signal is then more similar to *E. coli*.

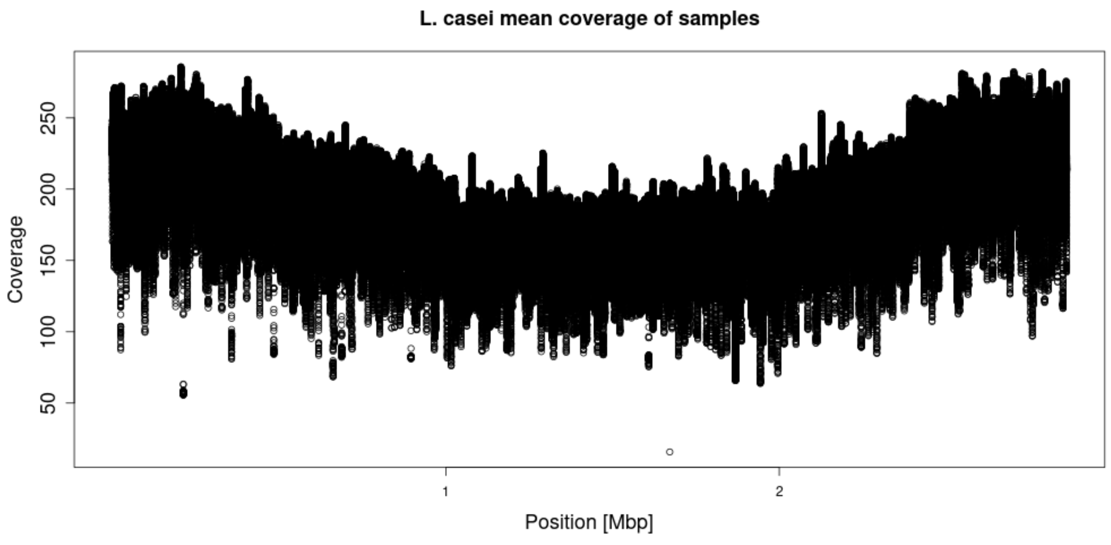


Figure 6.16 – Mean read-depth signal across all samples for *L. casei*

Regarding the top values, *L.casei* results were similar to *K. pneumoniae* results, meaning that CNproScan and Pindel detected the most events. However, generally, *L. casei* had the lowest number of detected CNVs, 6164. Also, it had the lowest number of overlapping events. Only around 500 CNVs were detected by any combination of tools, compared to 5700 CNVs detected uniquely. Also, no CNV was detected by all tools. See the Venn diagram in Figure 6.17.

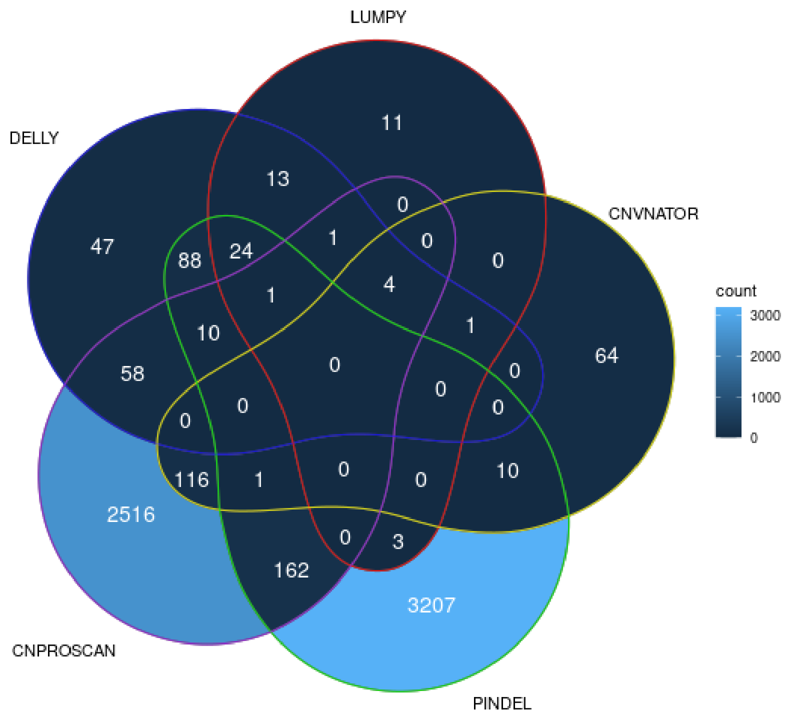


Figure 6.17 – Venn diagram for *L. casei* dataset

Those which were detected are deletions predominantly (Figure 6.18). CNproScan detected the second-highest number of both deletions and duplications. Most events were small, in size up to 1000 bp. Again, CNproScan was the second in the number of CNVs detected in both size categories. See Supplementary Figures H1-H2 for plots of CNV lengths on CNV types and CNV size preference by callers.

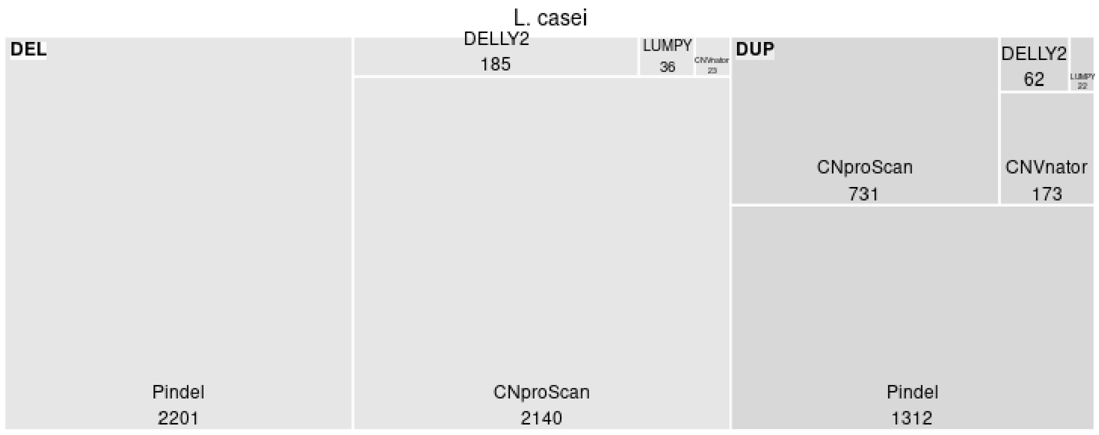


Figure 6.18 – CNV types by caller for *L. casei* dataset

6.6 Runtime analysis

The Snakemake optionally outputs the runtime statistics for all rules. All normalizations for CNproScan were turned on. The running times are higher than the numbers observed in the CNproScan runtime analysis. This can be caused by two reasons. First, only a single sample was tested in the previous runtime analysis. Second, the CPU frequencies are lower for the multi-threaded load over all cores. Additionally, the I/O overhead can play a role. In Table 6.5 there are runtimes for detection tools and merging methods.

Table 6.5 – Runtime for *K. pneumoniae* samples

Tool	**Mean runtime [s]**	**Min runtime [s]**	**Max runtime [s]**
BWA-MEM2	148	75	208
CNproScan	1017	629	141
CNVnator	28	16	35
DELLY2	479	351	580
LUMPY	21	7	32
Pindel	769	448	1025
ProcaryaSV merge	50	38	66
SURVIVOR merge	0.49	0.33	0.65

6.7 Summary

In this chapter, I presented the ProcaryaSV pipeline and its novel merging algorithm. It provided a reproducible framework for the evaluation CNproScan's performance on the real dataset. The pipeline can detect CNVs and also inversions and insertions.

In the field of genome rearrangements detection, merging multiple detection tools seems unavoidable. By using a single tool, we usually achieve either high sensitivity or high specificity. By merging multiple tools we can resolve more true positives and filter out many false positives. Integrating multiple tools would not be possible without robust merging.

In the proposed merging algorithm, multiple callers can be efficiently merged. The performance of merging was tested on the previous artificial dataset. The results are comparable with the SURVIVOR merging method. The ProcaryaSV merging performed better in coverage values of 20×, 100×, and 200×. It was also calculated that the optimal parameter of minimal callers support is 2 to 4, depending on the stringency for filtering false positives. The additional bonus is reporting the coordinates of the region with the highest support, listing all callers involved and their representation in the overall length of detected SV.

The analysis of real sequencing data was done on five bacteria organisms and 248 genomes. I analyzed mainly the overlaps between the tools and the preference of detection tools for CNV types and lengths.

CONCLUSION

The main topic of the thesis is the detection of copy number variations specifically in the prokaryotic genomes. While there is a tremendous and still increasing number of papers focused on the topic of SVs and CNVs, the resources related to bacteria are much less frequent. The work presented in the thesis aims to partially fill this gap.

In two practical chapters, I described the CNproScan algorithm and pipeline implementing it called ProcaryaSV. Both tools might be useful for microbiology research.

The CNproScan was published two years ago and was minorly updated a few times, thus I presented the original results with the updated ones. The methodology is based on read-depth navigated CNV detection combined with read-pair-based categorization. The read-pair approach is based on recent knowledge and enables the categorization of CNVs into known duplication types. The CNproScan does not consist only of detection methods. It handles the GC and mappability biases and bacteria-only related replication origin bias.

The CNproScan was tested on the artificial dataset of various coverages (10×, 20×, 100×, 200×) and compared with other seven CNV detection tools. CNproScan had the highest accuracy and F1 score for 20×, 100×, and 200× coverage. The accuracy for 100× was 93.3 % and the F1 score was 84.6 %. That is about 10 % higher than the closest competition. Also, it proved to be useful for detecting short CNVs under 25bp. The reported CNV boundaries are accurately corresponding to the specified boundaries in the majority of test cases. The accuracy of reporting a valid copy number is about 75 %.

Integration of multiple detection tools has already been done in the past. Merging two methods can easily be done, but the scalability decreases with adding more tools. Thus, I used a signal representation of genome rearrangements and summed the signals of individual detection tools. The merging algorithm was tested on the previous artificial dataset and compared with the SURVIVOR merging algorithm. Generally, the two methods are comparable, yet for coverages starting at 20×, the ProcaryaSV's merging algorithm performed slightly better. Both accuracy and F1 score are about 90 %. The parameter of minimal callers support, denoting how many callers have to detect an SV, was calculated from the precision-recall graph to be ideally 2 or higher. The ProcaryaSV pipeline employs five state-of-the-art detection tools and provides all necessary inputs and outputs for them. The pipeline enables reproducibility and is coded in the Snakemake.

Regarding the limitations of the presented methods. The CNproScan's running time is higher than the competition. This was mitigated as much as possible by implementing parallelization. Furthermore, the algorithm is based on the read-depth approach and requires a certain level of coverage, which is 20× based on results. However, coverage higher than 15× is a common requirement for the detection of any genome rearrangements. Generally, higher coverage leads to higher accuracy.

Nowadays, the topic of bacteria drug resistance is an urgent task. Genome rearrangements, including copy number variations, play a role in this issue. Other than that, genome rearrangements participate in the evolution and specialization of bacteria. Next-generation sequencing is still widely used and brings the high throughput necessary for accurate CNV detection. A reliable tool that is directly designed to detect CNVs in bacterial genomes (like the CNproScan), unlike tools designed for eukaryotic genomes, is essential. In turn, the proposed ProcaryaSV pipeline will enable CNV and SV analysis with maximum support for clinically relevant results.

REFERENCES

[1] Y. Li *et al.*, “Patterns of somatic structural variation in human cancer genomes,” *Nature*, vol. 578, no. 7793, pp. 112–121, Feb. 2020, doi: 10.1038/s41586-019-1913-9.

[2] J. K. Sehn, “Chapter 9 - Insertions and Deletions (Indels),” in *Clinical Genomics*, S. Kulkarni and J. Pfeifer, Eds., Boston: Academic Press, 2015, pp. 129–150. doi: 10.1016/B978-0-12-404748-8.00009-5.

[3] S. Kosugi, Y. Momozawa, X. Liu, C. Terao, M. Kubo, and Y. Kamatani, “Comprehensive evaluation of structural variation detection algorithms for whole genome sequencing,” *Genome Biol.*, vol. 20, p. 117, Jun. 2019, doi: 10.1186/s13059-019-1720-5.

[4] A. R. Quinlan and I. M. Hall, “Characterizing complex structural variation in germline and somatic genomes,” *Trends Genet. TIG*, vol. 28, no. 1, pp. 43–53, Jan. 2012, doi: 10.1016/j.tig.2011.10.002.

[5] A. Ameur *et al.*, “SweGen: a whole-genome data resource of genetic variability in a cross-section of the Swedish population,” *Eur. J. Hum. Genet.*, vol. 25, no. 11, Art. no. 11, Nov. 2017, doi: 10.1038/ejhg.2017.130.

[6] A. J. Sharp, Z. Cheng, and E. E. Eichler, “Structural variation of the human genome,” *Annu. Rev. Genomics Hum. Genet.*, vol. 7, pp. 407–442, 2006, doi: 10.1146/annurev.genom.7.080505.115618.

[7] J. M. Kidd *et al.*, “A Human Genome Structural Variation Sequencing Resource Reveals Insights into Mutational Mechanisms,” *Cell*, vol. 143, no. 5, pp. 837–847, Nov. 2010, doi: 10.1016/j.cell.2010.10.027.

[8] B. Liu *et al.*, “Structural variation discovery in the cancer genome using next generation sequencing: computational solutions and perspectives,” *Oncotarget*, vol. 6, no. 8, pp. 5477–5489, Mar. 2015, doi: 10.18632/oncotarget.3491.

[9] D. P. Clark, N. J. Pazdernik, and M. R. McGehee, *Molecular Biology*, 3rd ed., vol. 2019. Elsevier Inc. [Online]. Available: https://doi.org/10.1016/C2015-0-06229-3

[10] A. Sulovari *et al.*, “Human-specific tandem repeat expansion and differential gene expression during primate evolution,” *Proc. Natl. Acad. Sci.*, vol. 116, no. 46, pp. 23243–23253, Nov. 2019, doi: 10.1073/pnas.1912175116.

[11] T. J. Treangen and S. L. Salzberg, “Repetitive DNA and next-generation sequencing: Computational challenges and solutions,” *Nat. Rev. Genet.*, vol. 13, no. 1, pp. 36–46, 2012, doi: 10.1038/nrg3117.

[12] G. Escaramís, E. Docampo, and R. Rabionet, “A decade of structural variants: description, history and methods to detect structural variation,” *Brief. Funct. Genomics*, vol. 14, no. 5, pp. 305–314, Sep. 2015, doi: 10.1093/bfgp/elv014.

[13] R. L. Collins *et al.*, “Defining the diverse spectrum of inversions, complex structural variation, and chromothripsis in the morbid human genome,” *Genome Biol.*, vol. 18, no. 1, p. 36, Mar. 2017, doi: 10.1186/s13059-017-1158-6.

[14] A. J. Holland and D. W. Cleveland, “Chromoanagenesis and cancer: mechanisms and consequences of localized, complex chromosomal rearrangements,” *Nat. Med.*, vol. 18, no. 11, pp. 1630–1638, Nov. 2012, doi: 10.1038/nm.2988.

[15] P. J. Stephens *et al.*, “Massive genomic rearrangement acquired in a single catastrophic event during cancer development,” *Cell*, vol. 144, no. 1, pp. 27–40, Jan. 2011, doi: 10.1016/j.cell.2010.11.055.

[16] P. Liu *et al.*, “Chromosome catastrophes involve replication mechanisms generating complex genomic rearrangements,” *Cell*, vol. 146, no. 6, pp. 889–903, Sep. 2011, doi: 10.1016/j.cell.2011.07.042.

[17] S. C. Baca *et al.*, “Punctuated evolution of prostate cancer genomes,” *Cell*, vol. 153, no. 3, pp. 666–677, Apr. 2013, doi: 10.1016/j.cell.2013.03.021.

[18] A. Sanchis-Juan *et al.*, “Complex structural variants in Mendelian disorders: identification and breakpoint resolution using short- and long-read genome sequencing,” *Genome Med.*, vol. 10, p. 95, Dec. 2018, doi: 10.1186/s13073-018-0606-6.

[19] O. Pös *et al.*, “DNA copy number variation: Main characteristics, evolutionary significance, and pathological aspects,” *Biomed. J.*, vol. 44, no. 5, pp. 548–559, Oct. 2021, doi: 10.1016/j.bj.2021.02.003.

[20] D. F. Conrad *et al.*, “Origins and functional impact of copy number variation in the human genome,” *Nature*, vol. 464, no. 7289, pp. 704–712, Apr. 2010, doi: 10.1038/nature08516.

[21] A. J. Sharp *et al.*, “Segmental Duplications and Copy-Number Variation in the Human Genome,” *Am. J. Hum. Genet.*, vol. 77, no. 1, pp. 78–88, Jul. 2005, doi: 10.1086/431652.

[22] P. Stankiewicz and J. R. Lupski, “Structural variation in the human genome and its role in disease,” *Annu. Rev. Med.*, vol. 61, pp. 437–455, 2010, doi: 10.1146/annurev-med-100708-204735.

[23] A. W. C. Pang, O. Migita, J. R. Macdonald, L. Feuk, and S. W. Scherer, “Mechanisms of formation of structural variation in a fully sequenced human genome,” *Hum. Mutat.*, vol. 34, no. 2, pp. 345–354, Feb. 2013, doi: 10.1002/humu.22240.

[24] C. M. B. Carvalho and J. R. Lupski, “Mechanisms underlying structural variant formation in genomic disorders,” *Nat. Rev. Genet.*, vol. 17, no. 4, pp. 224–238, Apr. 2016, doi: 10.1038/nrg.2015.25.

[25] J. R. Chapman, M. R. G. Taylor, and S. J. Boulton, “Playing the End Game: DNA Double-Strand Break Repair Pathway Choice,” *Mol. Cell*, vol. 47, no. 4, pp. 497–510, Aug. 2012, doi: 10.1016/j.molcel.2012.07.029.

[26] A. Mehta and J. E. Haber, “Sources of DNA Double-Strand Breaks and Models of Recombinational DNA Repair,” *Cold Spring Harb. Perspect. Biol.*, vol. 6, no. 9, p. a016428, Sep. 2014, doi: 10.1101/cshperspect.a016428.

[27] B. B. Currall, C. Chiangmai, M. E. Talkowski, and C. C. Morton, “Mechanisms for Structural Variation in the Human Genome,” *Curr. Genet. Med. Rep.*, vol. 1, no. 2, pp. 81–90, Jun. 2013, doi: 10.1007/s40142-013-0012-8.

[28] P. J. Hastings, J. R. Lupski, S. M. Rosenberg, and G. Ira, “Mechanisms of change in gene copy number,” *Nat. Rev. Genet.*, vol. 10, no. 8, pp. 551–564, Aug. 2009, doi: 10.1038/nrg2593.

[29] A. C. Vítor, P. Huertas, G. Legube, and S. F. de Almeida, “Studying DNA Double-Strand Break Repair: An Ever-Growing Toolbox,” *Front. Mol. Biosci.*, vol. 7, 2020, Accessed: Mar. 29, 2022. [Online]. Available: https://www.frontiersin.org/article/10.3389/fmolb.2020.00024

[30] P. J. Hastings, G. Ira, and J. R. Lupski, “A microhomology-mediated break-induced replication model for the origin of human copy number variation,” *PLoS Genet.*, vol. 5, no. 1, p. e1000327, Jan. 2009, doi: 10.1371/journal.pgen.1000327.

[31] C. J. Sakofsky, S. Ayyar, A. K. Deem, W.-H. Chung, G. Ira, and A. Malkova, “Translesion Polymerases Drive Microhomology-Mediated Break-Induced Replication Leading to Complex Chromosomal Rearrangements,” *Mol. Cell*, vol. 60, no. 6, pp. 860–872, Dec. 2015, doi: 10.1016/j.molcel.2015.10.041.

[32] J. Weischenfeldt, O. Symmons, F. Spitz, and J. O. Korbel, “Phenotypic impact of genomic structural variation: insights from and for human disease,” *Nat. Rev. Genet.*, vol. 14, no. 2, pp. 125–138, Feb. 2013, doi: 10.1038/nrg3373.

[33] M. Chowdhury, B. S. Pedersen, F. J. Sedlazeck, A. R. Quinlan, and R. M. Layer, “Searching thousands of genomes to classify somatic and novel structural variants using STIX,” *Nat. Methods*, vol. 19, no. 4, pp. 445–448, Apr. 2022, doi: 10.1038/s41592-022-01423-4.

[34] L. Zhang, R. Reifová, Z. Halenková, and Z. Gompert, “How Important Are Structural Variants for Speciation?,” *Genes*, vol. 12, no. 7, p. 1084, Jul. 2021, doi: 10.3390/genes12071084.

[35] X. Dai, R. Theobard, H. Cheng, M. Xing, and J. Zhang, “Fusion genes: A promising tool combating against cancer,” *Biochim. Biophys. Acta BBA - Rev. Cancer*, vol. 1869, no. 2, pp. 149–160, Apr. 2018, doi: 10.1016/j.bbcan.2017.12.003.

[36] T. Lappalainen, A. J. Scott, M. Brandt, and I. M. Hall, “Genomic Analysis in the Age of Human Genome Sequencing,” *Cell*, vol. 177, no. 1, pp. 70–84, Mar. 2019, doi: 10.1016/j.cell.2019.02.032.

[37] L. Yang *et al.*, “Diverse mechanisms of somatic structural variations in human cancer genomes,” *Cell*, vol. 153, no. 4, pp. 919–929, May 2013, doi: 10.1016/j.cell.2013.04.010.

[38] International Human Genome Sequencing Consortium, “Finishing the euchromatic sequence of the human genome,” *Nature*, vol. 431, no. 7011, Art. no. 7011, Oct. 2004, doi: 10.1038/nature03001.

[39] “A second generation human haplotype map of over 3.1 million SNPs,” *Nature*, vol. 449, no. 7164, pp. 851–861, Oct. 2007, doi: 10.1038/nature06258.

[40] R. M. Durbin *et al.*, “A map of human genome variation from population-scale sequencing,” *Nature*, vol. 467, no. 7319, pp. 1061–1073, Oct. 2010, doi: 10.1038/nature09534.

[41] R. E. Mills *et al.*, “Mapping copy number variation by population-scale genome sequencing,” *Nature*, vol. 470, no. 7332, pp. 59–65, Feb. 2011, doi: 10.1038/nature09708.

[42] G. A. McVean *et al.*, “An integrated map of genetic variation from 1,092 human genomes,” *Nature*, vol. 491, no. 7422, Art. no. 7422, Nov. 2012, doi: 10.1038/nature11632.

[43] P. H. Sudmant *et al.*, “An integrated map of structural variation in 2,504 human genomes,” *Nature*, vol. 526, no. 7571, pp. 75–81, Oct. 2015, doi: 10.1038/nature15394.

[44] A. Telenti *et al.*, “Deep sequencing of 10,000 human genomes,” *Proc. Natl. Acad. Sci. U. S. A.*, vol. 113, no. 42, pp. 11901–11906, Oct. 2016, doi: 10.1073/pnas.1613365113.

[45] H. J. Abel *et al.*, “Mapping and characterization of structural variation in 17,795 human genomes,” *Nature*, vol. 583, no. 7814, pp. 83–89, Jul. 2020, doi: 10.1038/s41586-020-2371-0.

[46] D. E. Larson *et al.*, “svtools: population-scale analysis of structural variation,” *Bioinformatics*, vol. 35, no. 22, pp. 4782–4787, Nov. 2019, doi: 10.1093/bioinformatics/btz492.

[47] L. C. Francioli *et al.*, “Whole-genome sequence variation, population structure and demographic history of the Dutch population,” *Nat. Genet.*, vol. 46, no. 8, pp. 818–825, Aug. 2014, doi: 10.1038/ng.3021.

[48] M. Nagasaki *et al.*, “Rare variant discovery by deep whole-genome sequencing of 1,070 Japanese individuals,” *Nat. Commun.*, vol. 6, no. 1, Art. no. 1, Aug. 2015, doi: 10.1038/ncomms9018.

[49] S. Besenbacher *et al.*, “Novel variation and de novo mutation rates in population-wide de novo assembled Danish trios,” *Nat. Commun.*, vol. 6, no. 1, Art. no. 1, Jan. 2015, doi: 10.1038/ncomms6969.

[50] J. Kim *et al.*, “KoVariome: Korean National Standard Reference Variome database of whole genomes with comprehensive SNV, indel, CNV, and SV analyses,” *Sci. Rep.*, vol. 8, no. 1, Art. no. 1, Apr. 2018, doi: 10.1038/s41598-018-23837-x.

[51] S. S. Ho, A. E. Urban, and R. E. Mills, “Structural variation in the sequencing era,” *Nat. Rev. Genet.*, vol. 21, no. 3, Art. no. 3, Mar. 2020, doi: 10.1038/s41576-019-0180-9.

[52] E. Tuzun *et al.*, “Fine-scale structural variation of the human genome,” *Nat. Genet.*, vol. 37, no. 7, pp. 727–732, Jul. 2005, doi: 10.1038/ng1562.

[53] J. Zhang, L. Feuk, G. E. Duggan, R. Khaja, and S. W. Scherer, “Development of bioinformatics resources for display and analysis of copy number and other structural variants in the human genome,” *Cytogenet. Genome Res.*, vol. 115, no. 3–4, pp. 205–214, 2006, doi: 10.1159/000095916.

[54] J. R. MacDonald, R. Ziman, R. K. C. Yuen, L. Feuk, and S. W. Scherer, “The Database of Genomic Variants: a curated collection of structural variation in the human genome,” *Nucleic Acids Res.*, vol. 42, no. Database issue, pp. D986-992, Jan. 2014, doi: 10.1093/nar/gkt958.

[55] I. Lappalainen *et al.*, “DbVar and DGVa: public archives for genomic structural variation,” *Nucleic Acids Res.*, vol. 41, no. Database issue, pp. D936-941, Jan. 2013, doi: 10.1093/nar/gks1213.

[56] K. J. Karczewski *et al.*, “The mutational constraint spectrum quantified from variation in 141,456 humans,” *Nature*, vol. 581, no. 7809, Art. no. 7809, May 2020, doi: 10.1038/s41586-020-2308-7.

[57] R. L. Collins *et al.*, “A structural variation reference for medical and population genetics,” *Nature*, vol. 581, no. 7809, pp. 444–451, May 2020, doi: 10.1038/s41586-020-2287-8.

[58] J. G. Tate *et al.*, “COSMIC: the Catalogue Of Somatic Mutations In Cancer,” *Nucleic Acids Res.*, vol. 47, no. D1, pp. D941–D947, Jan. 2019, doi: 10.1093/nar/gky1015.

[59] T. Horiuchi, S. Horiuchi, and A. Novick, “The Genetic Basis of Hyper-Synthesis of β-Galactosidase,” *Genetics*, vol. 48, no. 2, pp. 157–169, Feb. 1963.

[60] E. Darmon and D. R. F. Leach, “Bacterial Genome Instability,” *Microbiol. Mol. Biol. Rev.*, vol. 78, no. 1, pp. 1–39, 2014, doi: 10.1128/mmbr.00035-13.

[61] P. T. West, R. B. Chanin, and A. S. Bhatt, “From genome structure to function: insights into structural variation in microbiology,” *Curr. Opin. Microbiol.*, vol. 69, p. 102192, Oct. 2022, doi: 10.1016/j.mib.2022.102192.

[62] Z. Seferbekova *et al.*, “High Rates of Genome Rearrangements and Pathogenicity of Shigella spp.,” *Front. Microbiol.*, vol. 12, 2021, Accessed: Feb. 03, 2023. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fmicb.2021.628622

[63] P. Hu *et al.*, “Comparative Genomics Study of Multi-Drug-Resistance Mechanisms in the Antibiotic-Resistant Streptococcus suis R61 Strain,” *PLoS ONE*, vol. 6, no. 9, p. e24988, Sep. 2011, doi: 10.1371/journal.pone.0024988.

[64] T. A. Brown, *Genomes*, 2nd ed. Oxford: Wiley-Liss, 2002. Accessed: Sep. 13, 2022. [Online]. Available: http://www.ncbi.nlm.nih.gov/books/NBK21128/

[65] V. Periwal and V. Scaria, “Insights into structural variations and genome rearrangements in prokaryotic genomes,” *Bioinformatics*, vol. 31, no. 1, pp. 1–9, 2015, doi: 10.1093/bioinformatics/btu600.

[66] E. P. C. Rocha, “The Organization of the Bacterial Genome,” *Annu. Rev. Genet.*, vol. 42, no. 1, pp. 211–233, 2008, doi: 10.1146/annurev.genet.42.110807.091653.

[67] S. F. Fitzgerald *et al.*, “Genome structural variation in Escherichia coli O157:H7,” *Microb. Genomics*, vol. 7, no. 11, p. 000682, doi: 10.1099/mgen.0.000682.

[68] A. Slack, P. C. Thornton, D. B. Magner, S. M. Rosenberg, and P. J. Hastings, “On the Mechanism of Gene Amplification Induced under Stress in Escherichia coli,” *PLoS Genet.*, vol. 2, no. 4, p. e48, Apr. 2006, doi: 10.1371/journal.pgen.0020048.

[69] A. E. Darling, I. Miklós, and M. A. Ragan, “Dynamics of Genome Rearrangement in Bacterial Populations,” *PLOS Genet.*, vol. 4, no. 7, p. e1000128, 7 2008, doi: 10.1371/journal.pgen.1000128.

[70] D. Hughes, “Evaluating genome dynamics: the constraints on rearrangements within bacterial genomes,” *Genome Biol.*, vol. 1, no. 6, p. reviews0006.1, Dec. 2000, doi: 10.1186/gb-2000-1-6-reviews0006.

[71] J. Repar and T. Warnecke, “Non-Random Inversion Landscapes in Prokaryotic Genomes Are Shaped by Heterogeneous Selection Pressures,” *Mol. Biol. Evol.*, vol. 34, no. 8, pp. 1902–1911, Aug. 2017, doi: 10.1093/molbev/msx127.

[72] P. J. Hastings and S. M. Rosenberg, “In pursuit of a molecular mechanism for adaptive gene amplification,” *DNA Repair*, vol. 1, no. 2, pp. 111–123, Feb. 2002, doi: 10.1016/s1568-7864(01)00011-8.

[73] R. J. Bengtsson *et al.*, “Accessory Genome Dynamics and Structural Variation of Shigella from Persistent Infections,” *mBio*, vol. 12, no. 2, pp. e00254-21, doi: 10.1128/mBio.00254-21.

[74] G. Firrao *et al.*, “Genomic Structural Variations Affecting Virulence During Clonal Expansion of Pseudomonas syringae pv. actinidiae Biovar 3 in Europe,” *Front. Microbiol.*, vol. 9, 2018, Accessed: Mar. 06, 2022. [Online]. Available: https://www.frontiersin.org/article/10.3389/fmicb.2018.00656

[75] D. Zeevi *et al.*, “Structural variation in the gut microbiome associates with host health,” *Nature*, vol. 568, no. 7750, Art. no. 7750, Apr. 2019, doi: 10.1038/s41586-019-1065-y.

[76] P. Balachandran and C. R. Beck, “Structural variant identification and characterization,” *Chromosome Res.*, vol. 28, no. 1, pp. 31–47, Mar. 2020, doi: 10.1007/s10577-019-09623-z.

[77] S. Chan *et al.*, “Structural Variation Detection and Analysis Using Bionano Optical Mapping,” *Methods Mol. Biol. Clifton NJ*, vol. 1833, pp. 193–203, 2018, doi: 10.1007/978-1-4939-8666-8_16.

[78] N. P. Carter, “Methods and strategies for analyzing copy number variation using DNA microarrays,” *Nat. Genet.*, vol. 39, no. 7 Suppl, pp. S16-21, Jul. 2007, doi: 10.1038/ng2028.

[79] D. Pinkel *et al.*, “High resolution analysis of DNA copy number variation using comparative genomic hybridization to microarrays,” *Nat. Genet.*, vol. 20, no. 2, pp. 207–211, Oct. 1998, doi: 10.1038/2524.

[80] A. Kallioniemi *et al.*, “Comparative Genomic Hybridization for Molecular Cytogenetic Analysis of Solid Tumors,” *Science*, vol. 258, no. 5083, pp. 818–821, Oct. 1992, doi: 10.1126/science.1359641.

[81] O.-P. Kallioniemi *et al.*, “Optimizing comparative genomic hybridization for analysis of DNA sequence copy number changes in solid tumors,” *Genes. Chromosomes Cancer*, vol. 10, no. 4, pp. 231–243, 1994, doi: 10.1002/gcc.2870100403.

[82] D. Pinkel and D. G. Albertson, “Comparative Genomic Hybridization,” *Annu. Rev. Genomics Hum. Genet.*, vol. 6, no. 1, pp. 331–354, 2005, doi: 10.1146/annurev.genom.6.080604.162140.

[83] C. Alkan, B. P. Coe, and E. E. Eichler, “Genome structural variation discovery and genotyping,” *Nat. Rev. Genet.*, vol. 12, no. 5, pp. 363–376, 2011, doi: 10.1038/nrg2958.

[84] B. A. Bejjani and L. G. Shaffer, “Application of Array-Based Comparative Genomic Hybridization to Clinical Diagnostics,” *J. Mol. Diagn. JMD*, vol. 8, no. 5, pp. 528–533, Nov. 2006, doi: 10.2353/jmoldx.2006.060029.

[85] B. Carvalho, E. Ouwerkerk, G. A. Meijer, and B. Ylstra, “High resolution microarray comparative genomic hybridisation analysis using spotted oligonucleotides,” *J. Clin. Pathol.*, vol. 57, no. 6, pp. 644–646, Jun. 2004, doi: 10.1136/jcp.2003.013029.

[86] W. R. Lai, M. D. Johnson, R. Kucherlapati, and P. J. Park, “Comparative analysis of algorithms for identifying amplifications and deletions in array CGH data,” *Bioinforma. Oxf. Engl.*, vol. 21, no. 19, pp. 3763–3770, Oct. 2005, doi: 10.1093/bioinformatics/bti611.

[87] A. B. Olshen, E. S. Venkatraman, R. Lucito, and M. Wigler, “Circular binary segmentation for the analysis of array-based DNA copy number data,” *Biostat. Oxf. Engl.*, vol. 5, no. 4, pp. 557–572, Oct. 2004, doi: 10.1093/biostatistics/kxh008.

[88] J. O. Korbel *et al.*, “Systematic prediction and validation of breakpoints associated with copy-number variants in the human genome,” *Proc. Natl. Acad. Sci.*, vol. 104, no. 24, pp. 10110–10115, Jun. 2007, doi: 10.1073/pnas.0703834104.

[89] “Molecular Analysis and Genome Discovery, 2nd Edition | Wiley,” *Wiley.com*. https://www.wiley.com/en-ai/Molecular+Analysis+and+Genome+Discovery%2C+2nd+Edition-p-9780470758779 (accessed Dec. 02, 2021).

[90] T. LaFramboise, “Single nucleotide polymorphism arrays: a decade of biological, computational and technological advances,” *Nucleic Acids Res.*, vol. 37, no. 13, pp. 4181–4193, Jul. 2009, doi: 10.1093/nar/gkp552.

[91] S. Colella *et al.*, “QuantiSNP: an Objective Bayes Hidden-Markov Model to detect and accurately map copy number variation using SNP genotyping data,” *Nucleic Acids Res.*, vol. 35, no. 6, pp. 2013–2025, 2007, doi: 10.1093/nar/gkm076.

[92] K. Wang *et al.*, “PennCNV: An integrated hidden Markov model designed for high-resolution copy number variation detection in whole-genome SNP genotyping data,” *Genome Res.*, vol. 17, no. 11, pp. 1665–1674, Nov. 2007, doi: 10.1101/gr.6861907.

[93] L. Winchester, C. Yau, and J. Ragoussis, “Comparing CNV detection methods for SNP arrays,” *Brief. Funct. Genomics*, vol. 8, no. 5, pp. 353–366, Sep. 2009, doi: 10.1093/bfgp/elp017.

[94] X. Zhang, R. Du, S. Li, F. Zhang, L. Jin, and H. Wang, “Evaluation of copy number variation detection for a SNP array platform,” *BMC Bioinformatics*, vol. 15, no. 1, p. 50, Feb. 2014, doi: 10.1186/1471-2105-15-50.

[95] S. Li, X. Dou, R. Gao, X. Ge, M. Qian, and L. Wan, “A remark on copy number variation detection methods,” *PLOS ONE*, vol. 13, no. 4, p. e0196226, Winter 2018, doi: 10.1371/journal.pone.0196226.

[96] J. Hodzic, L. Gurbeta, E. Omanovic-Miklicanin, and A. Badnjevic, “Overview of Next-generation Sequencing Platforms Used in Published Draft Plant Genomes in Light of Genotypization of Immortelle Plant (Helichrysium Arenarium),” *Med. Arch.*, vol. 71, no. 4, pp. 288–292, Aug. 2017, doi: 10.5455/medarh.2017.71.288-292.

[97] J. M. Heather and B. Chain, “The sequence of sequencers: The history of sequencing DNA,” *Genomics*, vol. 107, no. 1, pp. 1–8, Jan. 2016, doi: 10.1016/j.ygeno.2015.11.003.

[98] J. Shendure and H. Ji, “Next-generation DNA sequencing,” *Nat. Biotechnol.*, vol. 26, no. 10, pp. 1135–1145, Oct. 2008, doi: 10.1038/nbt1486.

[99] J. Shendure, R. D. Mitra, C. Varma, and G. M. Church, “Advanced sequencing technologies: methods and goals,” *Nat. Rev. Genet.*, vol. 5, no. 5, pp. 335–344, May 2004, doi: 10.1038/nrg1325.

[100] M. Kircher and J. Kelso, “High-throughput DNA sequencing – concepts and limitations,” *BioEssays*, vol. 32, no. 6, pp. 524–536, 2010, doi: 10.1002/bies.200900181.

[101] H. P. J. Buermans and J. T. den Dunnen, “Next generation sequencing technology: Advances and applications,” *Biochim. Biophys. Acta BBA - Mol. Basis Dis.*, vol. 1842, no. 10, pp. 1932–1941, 2014, doi: 10.1016/j.bbadis.2014.06.015.

[102] J. C. Dohm, C. Lottaz, T. Borodina, and H. Himmelbauer, “Substantial biases in ultra-short read data sets from high-throughput DNA sequencing,” *Nucleic Acids Res.*, vol. 36, no. 16, 2008, doi: 10.1093/nar/gkn425.

[103] A. Masoudi-Nejad, Z. Narimani, and N. Hosseinkhan, *Next Generation Sequencing and Sequence Assembly*, vol. 4. in SpringerBriefs in Systems Biology, vol. 4. New York, NY: Springer, 2013. doi: 10.1007/978-1-4614-7726-6.

[104] E. S. Lander *et al.*, “Initial sequencing and analysis of the human genome,” *Nature*, vol. 409, no. 6822, pp. 860–921, Feb. 2001, doi: 10.1038/35057062.

[105] S. Anderson, “Shotgun DNA sequencing using cloned DNase I-generated fragments.,” *Nucleic Acids Res.*, vol. 9, no. 13, pp. 3015–3027, Jul. 1981.

[106] A. Edwards *et al.*, “Automated DNA sequencing of the human HPRT locus,” *Genomics*, vol. 6, no. 4, pp. 593–608, Apr. 1990, doi: 10.1016/0888-7543(90)90493-E.

[107] J. Commins, C. Toft, and M. A. Fares, “Computational Biology Methods and Their Application to the Comparative Genomics of Endocellular Symbiotic Bacteria of Insects,” *Biol. Proced. Online*, vol. 11, pp. 52–78, Mar. 2009, doi: 10.1007/s12575-009-9004-1.

[108] J. Zhang, K. Kobert, T. Flouri, and A. Stamatakis, “PEAR: a fast and accurate Illumina Paired-End reAd mergeR,” *Bioinformatics*, vol. 30, no. 5, pp. 614–620, Mar. 2014, doi: 10.1093/bioinformatics/btt593.

[109] O. A. Hampton *et al.*, “SVachra: a tool to identify genomic structural variation in mate pair sequencing data containing inward and outward facing reads,” *BMC Genomics*, vol. 18, no. Suppl 6, p. 691, Oct. 2017, doi: 10.1186/s12864-017-4021-y.

[110] M. Zorc, J. Ogorevc, and P. Dovč, *Mining for Structural Variations in Next-Generation Sequencing Data*. IntechOpen, 2018. doi: 10.5772/intechopen.76568.

[111] P. Medvedev, M. Stanciu, and M. Brudno, “Computational methods for discovering structural variation with next-generation sequencing,” *Nat. Methods*, vol. 6, no. 11S, pp. S13–S13, 2009, doi: 10.1038/nmeth.1374.

[112] J. O. Korbel *et al.*, “Paired-End Mapping Reveals Extensive Structural Variation in the Human Genome,” *Science*, vol. 318, no. 5849, pp. 420–426, Oct. 2007, doi: 10.1126/science.1149504.

[113] K. Ye, M. H. Schulz, Q. Long, R. Apweiler, and Z. Ning, “Pindel: a pattern growth approach to detect break points of large deletions and medium sized insertions from

paired-end short reads," *Bioinformatics*, vol. 25, no. 21, pp. 2865–2871, Nov. 2009, doi: 10.1093/bioinformatics/btp394.

[114] H. J. Abel, E. J. Duncavage, N. Becker, J. R. Armstrong, V. J. Magrini, and J. D. Pfeifer, "SLOPE: a quick and accurate method for locating non-SNP structural variation from targeted next-generation sequence data," *Bioinforma. Oxf. Engl.*, vol. 26, no. 21, pp. 2684–2688, Nov. 2010, doi: 10.1093/bioinformatics/btq528.

[115] A. Soylev, T. M. Le, H. Amini, C. Alkan, and F. Hormozdiari, "Discovery of tandem and interspersed segmental duplications using high-throughput sequencing," *Bioinformatics*, vol. 35, no. 20, pp. 3923–3930, Oct. 2019, doi: 10.1093/bioinformatics/btz237.

[116] S. Lee, E. Cheran, and M. Brudno, "A robust framework for detecting structural variations in a genome," *Bioinformatics*, vol. 24, no. 13, pp. i59–i67, Jul. 2008, doi: 10.1093/bioinformatics/btn176.

[117] J. Qi and F. Zhao, "inGAP-sv: a novel scheme to identify and visualize structural variation from paired end mapping data," *Nucleic Acids Res.*, vol. 39, no. Web Server issue, pp. W567–W575, Jul. 2011, doi: 10.1093/nar/gkr506.

[118] T. Rausch, T. Zichner, A. Schlattl, A. M. Stütz, V. Benes, and J. O. Korbel, "DELLY: structural variant discovery by integrated paired-end and split-read analysis," *Bioinformatics*, vol. 28, no. 18, pp. i333–i339, Sep. 2012, doi: 10.1093/bioinformatics/bts378.

[119] R. E. Mills *et al.*, "An initial map of insertion and deletion (INDEL) variation in the human genome," *Genome Res.*, vol. 16, no. 9, pp. 1182–1190, Sep. 2006, doi: 10.1101/gr.4565806.

[120] J. Schröder *et al.*, "Socrates: identification of genomic rearrangements in tumour genomes by re-aligning soft clipped reads," *Bioinformatics*, vol. 30, no. 8, pp. 1064–1072, Apr. 2014, doi: 10.1093/bioinformatics/btt767.

[121] S. Suzuki, T. Yasuda, Y. Shiraishi, S. Miyano, and M. Nagasaki, "ClipCrop: a tool for detecting structural variations with single-base resolution using soft-clipping information," *BMC Bioinformatics*, vol. 12 Suppl 14, p. S7, Dec. 2011, doi: 10.1186/1471-2105-12-S14-S7.

[122] H. Li, "Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM," *ArXiv13033997 Q-Bio*, May 2013, Accessed: Apr. 15, 2022. [Online]. Available: http://arxiv.org/abs/1303.3997

[123] B. Langmead and S. L. Salzberg, "Fast gapped-read alignment with Bowtie 2," *Nat Methods*, vol. 9, no. 4, pp. 357–359, 2012, doi: 10.1038/nmeth.1923.

[124] P. Danecek *et al.*, "Twelve years of SAMtools and BCFtools," *GigaScience*, vol. 10, no. 2, p. giab008, Feb. 2021, doi: 10.1093/gigascience/giab008.

[125] R. M. Layer, C. Chiang, A. R. Quinlan, and I. M. Hall, "LUMPY: a probabilistic framework for structural variant discovery," *Genome Biol.*, vol. 15, no. 6, p. R84, Jun. 2014, doi: 10.1186/gb-2014-15-6-r84.

[126] D. R. Smith *et al.*, "Rapid whole-genome mutational profiling using next-generation sequencing technologies," *Genome Res.*, vol. 18, no. 10, pp. 1638–1642, Oct. 2008, doi: 10.1101/gr.077776.108.

[127] A. Magi, L. Tattini, T. Pippucci, F. Torricelli, and M. Benelli, "Read count approach for DNA copy number variants detection," *Bioinformatics*, vol. 28, no. 4, pp. 470–478, 2012, doi: 10.1093/bioinformatics/btr707.

[128] L. Tattini, R. D'Aurizio, and A. Magi, "Detection of Genomic Structural Variants from Next-Generation Sequencing Data," *Front. Bioeng. Biotechnol.*, vol. 3, p. 92, Jun. 2015, doi: 10.3389/fbioe.2015.00092.

[129] L. Zhang, W. Bai, N. Yuan, and Z. Du, "Comprehensively benchmarking applications for detecting copy number variation," *PLoS Comput. Biol.*, vol. 15, no. 5, pp. 1–12, 2019, doi: 10.1371/journal.pcbi.1007069.

[130] J. Duan, J.-G. Zhang, H.-W. Deng, and Y.-P. Wang, "Comparative Studies of Copy Number Variation Detection Methods for Next-Generation Sequencing Technologies," *PLoS ONE*, vol. 8, no. 3, pp. e59128–e59128, 2013, doi: 10.1371/journal.pone.0059128.

[131] S. Yoon, Z. Xuan, V. Makarov, K. Ye, and J. Sebat, "Sensitive and accurate detection of copy number variants using read depth of coverage," *Genome Res.*, vol. 19, no. 9, pp. 1586–1592, 2009, doi: 10.1101/gr.092981.109.

[132] M. C. Wendl, "A general coverage theory for shotgun DNA sequencing.," *J. Comput. Biol. J. Comput. Mol. Cell Biol.*, vol. 13, no. 6, pp. 1177–1196, 2006, doi: 10.1089/cmb.2006.13.1177.

[133] M. C. Wendl and W. B. Barbazuk, "Extension of Lander-Waterman theory for sequencing filtered DNA libraries," *BMC Bioinformatics*, vol. 6, pp. 1–12, 2005, doi: 10.1186/1471-2105-6-245.

[134] E. S. Lander and M. S. Waterman, "Genomic mapping by fingerprinting random clones: a mathematical analysis," *Genomics*, vol. 2, no. 3, pp. 231–239, Apr. 1988, doi: 10.1016/0888-7543(88)90007-9.

[135] M. Zhao, Q. Wang, Q. Wang, P. Jia, and Z. Zhao, "Computational tools for copy number variation (CNV) detection using next-generation sequencing data: features and perspectives - Springer," *BMC Bioinformatics*, vol. 14 Suppl 1, no. Suppl 11, pp. S1–S1, 2013, doi: 10.1186/1471-2105-14-S11-S1.

[136] Y.-C. Wei and G.-H. Huang, "CONY: A Bayesian procedure for detecting copy number variations from sequencing read depths," *Sci. Rep.*, vol. 10, no. 1, p. 10493, Jun. 2020, doi: 10.1038/s41598-020-64353-1.

[137] L. Zhao, H. Liu, X. Yuan, K. Gao, and J. Duan, "Comparative study of whole exome sequencing-based copy number variation detection tools," *BMC Bioinformatics*, vol. 21, no. 1, p. 97, Mar. 2020, doi: 10.1186/s12859-020-3421-1.

[138] J. M. Kidd *et al.*, "Mapping and sequencing of structural variation from eight human genomes," *Nature*, vol. 453, no. 7191, pp. 56–64, May 2008, doi: 10.1038/nature06862.

[139] M. Mahmoud, N. Gobet, D. I. Cruz-Dávalos, N. Mounier, C. Dessimoz, and F. J. Sedlazeck, "Structural variant calling: the long and the short of it," *Genome Biol.*, vol. 20, no. 1, p. 246, Nov. 2019, doi: 10.1186/s13059-019-1828-7.

[140] J.-Y. Zhang *et al.*, “Using de novo assembly to identify structural variation of eight complex immune system gene regions,” *PLOS Comput. Biol.*, vol. 17, no. 8, p. e1009254, 8 2021, doi: 10.1371/journal.pcbi.1009254.

[141] K. Lin, S. Smit, G. Bonnema, G. Sanchez-Perez, and D. de Ridder, “Making the difference: integrating structural variation detection tools,” *Brief. Bioinform.*, vol. 16, no. 5, pp. 852–864, Sep. 2015, doi: 10.1093/bib/bbu047.

[142] F. J. Sedlazeck, H. Lee, C. A. Darby, and M. C. Schatz, “Piercing the dark matter: bioinformatics of long-range sequencing and mapping,” *Nat. Rev. Genet.*, vol. 19, no. 6, pp. 329–346, Jun. 2018, doi: 10.1038/s41576-018-0003-4.

[143] M. Hayes, Y. S. Pyon, and J. Li, “A Model-Based Clustering Method for Genomic Structural Variant Prediction and Genotyping Using Paired-End Sequencing Data,” *PLOS ONE*, vol. 7, no. 12, p. e52881, 12 2012, doi: 10.1371/journal.pone.0052881.

[144] F. Hormozdiari, C. Alkan, E. E. Eichler, and S. C. Sahinalp, “Combinatorial algorithms for structural variation detection in high-throughput sequenced genomes,” *Genome Res.*, vol. 19, no. 7, pp. 1270–1278, Jan. 2009, doi: 10.1101/gr.088633.108.

[145] S. Lee, F. Hormozdiari, C. Alkan, and M. Brudno, “MoDIL: detecting small indels from clone-end sequencing with mixtures of distributions,” *Nat. Methods*, vol. 6, no. 7, pp. 473–474, Jul. 2009, doi: 10.1038/nmeth.f.256.

[146] S. M. Teo, Y. Pawitan, C. S. Ku, K. S. Chia, and A. Salim, “Statistical challenges associated with detecting copy number variations with next-generation sequencing,” *Bioinformatics*, vol. 28, no. 21, pp. 2711–2718, Nov. 2012, doi: 10.1093/bioinformatics/bts535.

[147] K. Wong, T. M. Keane, J. Stalker, and D. J. Adams, “Enhanced structural variant and breakpoint detection using SVMerge by integration of multiple detection methods and local assembly,” *Genome Biol.*, vol. 11, no. 12, p. R128, 2010, doi: 10.1186/gb-2010-11-12-r128.

[148] P. Guan and W.-K. Sung, “Structural variation detection using next-generation sequencing data: A comparative technical review,” *Methods San Diego Calif*, vol. 102, pp. 36–49, Jun. 2016, doi: 10.1016/j.ymeth.2016.01.020.

[149] J. Geryk, A. Zinkova, I. Zedníková, H. Simková, V. Stenzl, and M. Korabecna, “Improving structural variant clustering to reduce the negative effect of the breakpoint uncertainty problem,” *BMC Bioinformatics*, vol. 22, no. 1, p. 464, Sep. 2021, doi: 10.1186/s12859-021-04374-3.

[150] S. Sindi, E. Helman, A. Bashir, and B. J. Raphael, “A geometric approach for classification and comparison of structural variants,” *Bioinforma. Oxf. Engl.*, vol. 25, no. 12, pp. i222-230, Jun. 2009, doi: 10.1093/bioinformatics/btp208.

[151] A. Bashir, S. Volik, C. Collins, V. Bafna, and B. J. Raphael, “Evaluation of paired-end sequencing strategies for detection of genome rearrangements in cancer,” *PLoS Comput. Biol.*, vol. 4, no. 4, p. e1000051, Apr. 2008, doi: 10.1371/journal.pcbi.1000051.

[152] O. Brynildsrud, L. G. Snipen, and J. Bohlin, “CNOGpro: Detection and quantification of CNVs in prokaryotic whole-genome sequencing data,” *Bioinformatics*, vol. 31, no. 11, pp. 1708–1715, 2015, doi: 10.1093/bioinformatics/btv070.

[153] V. Boeva *et al.*, “Multi-factor data normalization enables the detection of copy number aberrations in amplicon sequencing data,” *Bioinformatics*, vol. 30, no. 24, pp. 3443–3450, Dec. 2014, doi: 10.1093/bioinformatics/btu436.

[154] R. Wang, D.-Y. Lin, and Y. Jiang, “SCOPE: A Normalization and Copy-Number Estimation Method for Single-Cell DNA Sequencing,” *Cell Syst.*, vol. 10, no. 5, pp. 445-452.e6, May 2020, doi: 10.1016/j.cels.2020.03.005.

[155] G. Klambauer *et al.*, “cn.MOPS: mixture of Poissons for discovering copy number variations in next-generation sequencing data with a low false discovery rate,” *Nucleic Acids Res.*, vol. 40, no. 9, p. e69, May 2012, doi: 10.1093/nar/gks003.

[156] A. Magi, M. Benelli, S. Yoon, F. Roviello, and F. Torricelli, “Detecting common copy number variants in high-throughput sequencing data by using JointSLM algorithm,” *Nucleic Acids Res.*, vol. 39, no. 10, p. e65, May 2011, doi: 10.1093/nar/gkr068.

[157] A. Magi *et al.*, “EXCAVATOR: detecting copy number variants from whole-exome sequencing data,” *Genome Biol.*, vol. 14, no. 10, p. R120, Dec. 2013, doi: 10.1186/gb-2013-14-10-r120.

[158] C. Xie and M. T. Tammi, “CNV-seq, a new method to detect copy number variation using high-throughput sequencing,” *BMC Bioinformatics*, vol. 10, no. 1, p. 80, Mar. 2009, doi: 10.1186/1471-2105-10-80.

[159] M. Pendleton *et al.*, “Assembly and diploid architecture of an individual human genome via single-molecule technologies,” *Nat. Methods*, vol. 12, no. 8, pp. 780–786, Aug. 2015, doi: 10.1038/nmeth.3454.

[160] M. Pirooznia, F. S. Goes, and P. P. Zandi, “Whole-genome CNV analysis: advances in computational approaches,” *Front. Genet.*, vol. 6, p. 138, 2015, doi: 10.3389/fgene.2015.00138.

[161] S. Volik *et al.*, “End-sequence profiling: Sequence-based analysis of aberrant genomes,” *Proc. Natl. Acad. Sci.*, vol. 100, no. 13, pp. 7696–7701, Jun. 2003, doi: 10.1073/pnas.1232418100.

[162] C. Alkan *et al.*, “Personalized copy number and segmental duplication maps using next-generation sequencing,” *Nat. Genet.*, vol. 41, no. 10, pp. 1061–1067, 2009, doi: 10.1038/ng.437.

[163] D. R. Bentley *et al.*, “Accurate whole human genome sequencing using reversible terminator chemistry,” *Nature*, vol. 456, no. 7218, pp. 53–59, Nov. 2008, doi: 10.1038/nature07517.

[164] P. J. Campbell *et al.*, “Identification of somatically acquired rearrangements in cancer using genome-wide massively parallel paired-end sequencing,” *Nat. Genet.*, vol. 40, no. 6, pp. 722–729, Jun. 2008, doi: 10.1038/ng.128.

[165] F. Hormozdiari *et al.*, “Next-generation VariationHunter: combinatorial algorithms for transposon insertion discovery,” *Bioinformatics*, vol. 26, no. 12, pp. i350–i357, Jun. 2010, doi: 10.1093/bioinformatics/btq216.

[166] J. O. Korbel *et al.*, “PEMer: a computational framework with simulation-based error models for inferring genomic structural variants from massive paired-end sequencing data,” *Genome Biol.*, vol. 10, no. 2, p. R23, Feb. 2009, doi: 10.1186/gb-2009-10-2-r23.

[167] K. Chen *et al.*, “BreakDancer: an algorithm for high-resolution mapping of genomic structural variation,” *Nat. Methods*, vol. 6, no. 9, pp. 677–681, Sep. 2009, doi: 10.1038/nmeth.1363.

[168] E. Karakoc *et al.*, “Detection of structural variants and indels within exome data,” *Nat. Methods*, vol. 9, no. 2, pp. 176–178, Dec. 2011, doi: 10.1038/nmeth.1810.

[169] J. Wang *et al.*, “CREST maps somatic structural variation in cancer genomes with base-pair resolution,” *Nat. Methods*, vol. 8, no. 8, pp. 652–654, Aug. 2011, doi: 10.1038/nmeth.1628.

[170] D. Y. Chiang *et al.*, “High-resolution mapping of copy-number alterations with massively parallel sequencing,” *Nat. Methods*, vol. 6, no. 1, pp. 99–103, Jan. 2009, doi: 10.1038/nmeth.1276.

[171] N. Sepúlveda, S. G. Campino, S. A. Assefa, C. J. Sutherland, A. Pain, and T. G. Clark, “A Poisson hierarchical modelling approach to detecting copy number variation in sequence coverage data,” *BMC Genomics*, vol. 14, no. 1, p. 128, Feb. 2013, doi: 10.1186/1471-2164-14-128.

[172] S. Ivakhno, T. Royce, A. J. Cox, D. J. Evers, R. K. Cheetham, and S. Tavaré, “CNAseg--a novel framework for identification of copy number changes in cancer from second-generation sequencing data,” *Bioinforma. Oxf. Engl.*, vol. 26, no. 24, pp. 3051–3058, Dec. 2010, doi: 10.1093/bioinformatics/btq587.

[173] A. Abyzov, A. E. Urban, M. Snyder, and M. Gerstein, “CNVnator: an approach to discover, genotype, and characterize typical and atypical CNVs from family and population genome sequencing,” *Genome Res.*, vol. 21, no. 6, pp. 974–984, Jun. 2011, doi: 10.1101/gr.114876.110.

[174] C. A. Miller, O. Hampton, C. Coarfa, and A. Milosavljevic, “ReadDepth: A Parallel R Package for Detecting Copy Number Alterations from Short Sequencing Reads,” *PLOS ONE*, vol. 6, no. 1, p. e16327, Spring 2011, doi: 10.1371/journal.pone.0016327.

[175] D. Desvillechabrol, C. Bouchier, S. Kennedy, and T. Cokelaer, “Sequana coverage: detection and characterization of genomic variations using running median and mixture models,” *GigaScience*, vol. 7, no. 12, Sep. 2018, doi: 10.1093/gigascience/giy110.

[176] L. Wu, H. Wang, Y. Xia, and R. Xi, “CNV-BAC: Copy number Variation Detection in Bacterial Circular Genome,” *Bioinformatics*, vol. 36, no. 12, pp. 3890–3891, Jun. 2020, doi: 10.1093/bioinformatics/btaa208.

[177] R. Xi, S. Lee, Y. Xia, T.-M. Kim, and P. J. Park, “Copy number analysis of whole-genome data using BIC-seq2 and its application to detection of cancer susceptibility variants,” *Nucleic Acids Res.*, vol. 44, no. 13, pp. 6274–6286, doi: 10.1093/nar/gkw491.

[178] R. Xi, J. Luquette, A. Hadjipanayis, T.-M. Kim, and P. J. Park, “BIC-seq: a fast algorithm for detection of copy number alterations based on high-throughput sequencing data,” *Genome Biol.*, vol. 11, no. Suppl 1, p. O10, 2010, doi: 10.1186/gb-2010-11-s1-o10.

[179] I. Hajirasouliha *et al.*, “Detection and characterization of novel sequence insertions using paired-end next-generation sequencing,” *Bioinforma. Oxf. Engl.*, vol. 26, no. 10, pp. 1277–1283, May 2010, doi: 10.1093/bioinformatics/btq152.

[180] Z. Iqbal, M. Caccamo, I. Turner, P. Flicek, and G. McVean, “De novo assembly and genotyping of variants using colored de Bruijn graphs,” *Nat. Genet.*, vol. 44, no. 2, pp. 226–232, Feb. 2012, doi: 10.1038/ng.1028.

[181] K. Chen, L. Chen, X. Fan, J. Wallis, L. Ding, and G. Weinstock, “TIGRA: A targeted iterative graph routing assembler for breakpoint assembly,” *Genome Res.*, vol. 24, no. 2, pp. 310–317, Feb. 2014, doi: 10.1101/gr.162883.113.

[182] P. Medvedev, M. Fiume, M. Dzamba, T. Smith, and M. Brudno, “Detecting copy number variation with mated short reads,” *Genome Res.*, vol. 20, no. 11, pp. 1613–1622, Nov. 2010, doi: 10.1101/gr.106344.110.

[183] A. R. Quinlan *et al.*, “Genome-wide mapping and assembly of structural variant breakpoints in the mouse genome,” *Genome Res.*, vol. 20, no. 5, pp. 623–635, May 2010, doi: 10.1101/gr.102970.109.

[184] D. He, F. Hormozdiari, N. Furlotte, and E. Eskin, “Efficient algorithms for tandem copy number variation reconstruction in repeat-rich regions,” *Bioinformatics*, vol. 27, no. 11, pp. 1513–1520, Jun. 2011, doi: 10.1093/bioinformatics/btr169.

[185] J. J. Michaelson and J. Sebat, “forestSV: structural variant discovery through statistical learning,” *Nat. Methods*, vol. 9, no. 8, pp. 819–821, Jul. 2012, doi: 10.1038/nmeth.2085.

[186] S. S. Sindi, S. Onal, L. C. Peng, H.-T. Wu, and B. J. Raphael, “An integrative probabilistic model for identification of structural variation in sequencing data,” *Genome Biol.*, vol. 13, no. 3, p. R22, 2012, doi: 10.1186/gb-2012-13-3-r22.

[187] Y. Jiang, Y. Wang, and M. Brudno, “PRISM: Pair-read informed split-read mapping for base-pair level detection of insertion, deletion and structural variants,” *Bioinformatics*, vol. 28, no. 20, pp. 2576–2583, Oct. 2012, doi: 10.1093/bioinformatics/bts484.

[188] G. Escaramís *et al.*, “PeSV-Fisher: Identification of Somatic and Non-Somatic Structural Variants Using Next Generation Sequencing Data,” *PLOS ONE*, vol. 8, no. 5, p. e63377, 5 2013, doi: 10.1371/journal.pone.0063377.

[189] B. J. Walker *et al.*, “Pilon: An integrated tool for comprehensive microbial variant detection and genome assembly improvement,” *PLoS ONE*, vol. 9, no. 11, 2014, doi: 10.1371/journal.pone.0112963.

[190] A. Soylev, C. Kockan, F. Hormozdiari, and C. Alkan, “Toolkit for automated and rapid discovery of structural variants,” *Methods San Diego Calif*, vol. 129, pp. 3–7, Oct. 2017, doi: 10.1016/j.ymeth.2017.05.030.

[191] I. A. E. M. van Belzen, A. Schönhuth, P. Kemmeren, and J. Y. Hehir-Kwa, “Structural variant detection in cancer genomes: computational challenges and perspectives for precision oncology,” *Npj Precis. Oncol.*, vol. 5, no. 1, Art. no. 1, Mar. 2021, doi: 10.1038/s41698-021-00155-6.

[192] L. T. Fang *et al.*, “An ensemble approach to accurately detect somatic mutations using SomaticSeq,” *Genome Biol.*, vol. 16, no. 1, p. 197, Sep. 2015, doi: 10.1186/s13059-015-0758-2.

[193] S. M. E. Sahraeian, R. Liu, B. Lau, K. Podesta, M. Mohiyuddin, and H. Y. K. Lam, "Deep convolutional neural networks for accurate somatic mutation detection," *Nat. Commun.*, vol. 10, no. 1, Art. no. 1, Mar. 2019, doi: 10.1038/s41467-019-09027-x.

[194] D. C. Jeffares *et al.*, "Transient structural variations have strong effects on quantitative traits and reproductive isolation in fission yeast," *Nat. Commun.*, vol. 8, no. 1, Art. no. 1, Jan. 2017, doi: 10.1038/ncomms14061.

[195] J. Eisfeldt, "SVDB." Dec. 30, 2022. Accessed: Jan. 15, 2023. [Online]. Available: https://github.com/J35P312/SVDB

[196] A. R. Quinlan and I. M. Hall, "BEDTools: a flexible suite of utilities for comparing genomic features," *Bioinforma. Oxf. Engl.*, vol. 26, no. 6, pp. 841–842, Mar. 2010, doi: 10.1093/bioinformatics/btq033.

[197] H. Y. K. Lam *et al.*, "Detecting and annotating genetic variations using the HugeSeq pipeline," *Nat. Biotechnol.*, vol. 30, no. 3, pp. 226–229, Mar. 2012, doi: 10.1038/nbt.2134.

[198] A. McKenna *et al.*, "The Genome Analysis Toolkit: a MapReduce framework for analyzing next-generation DNA sequencing data," *Genome Res.*, vol. 20, no. 9, pp. 1297–1303, Sep. 2010, doi: 10.1101/gr.107524.110.

[199] T. Mimori *et al.*, "iSVP: an integrated structural variant calling pipeline from high-throughput sequencing data," *BMC Syst. Biol.*, vol. 7, no. Suppl 6, p. S8, Dec. 2013, doi: 10.1186/1752-0509-7-S6-S8.

[200] L. Jia *et al.*, "intansv: an R package for integrative analysis of structural variations," *PeerJ*, vol. 8, p. e8867, Apr. 2020, doi: 10.7717/peerj.8867.

[201] S. N. Hart *et al.*, "SoftSearch: integration of multiple sequence features to identify breakpoints of structural variations," *PloS One*, vol. 8, no. 12, p. e83356, 2013, doi: 10.1371/journal.pone.0083356.

[202] M. Mohiyuddin *et al.*, "MetaSV: an accurate and integrative structural-variant caller for next generation sequencing," *Bioinformatics*, vol. 31, no. 16, pp. 2741–2744, Aug. 2015, doi: 10.1093/bioinformatics/btv204.

[203] A. Bankevich *et al.*, "SPAdes: A New Genome Assembly Algorithm and Its Applications to Single-Cell Sequencing," *J. Comput. Biol.*, vol. 19, no. 5, pp. 455–477, 2012, doi: 10.1089/cmb.2012.0021.

[204] A. Abyzov and M. Gerstein, "AGE: defining breakpoints of genomic structural variants at single-nucleotide resolution, through optimal alignments with gap excision," *Bioinformatics*, vol. 27, no. 5, pp. 595–603, Mar. 2011, doi: 10.1093/bioinformatics/btq713.

[205] T. Becker *et al.*, "FusorSV: an algorithm for optimally combining data from multiple structural variation detection methods," *Genome Biol.*, vol. 19, p. 38, Mar. 2018, doi: 10.1186/s13059-018-1404-6.

[206] J. Köster and S. Rahmann, "Snakemake—a scalable bioinformatics workflow engine," *Bioinformatics*, vol. 28, no. 19, pp. 2520–2522, Oct. 2012, doi: 10.1093/bioinformatics/bts480.

[207] A. Kuzniar *et al.*, “sv-callers: a highly portable parallel workflow for structural variant detection in whole-genome sequence data,” *PeerJ*, vol. 8, p. e8214, Jan. 2020, doi: 10.7717/peerj.8214.

[208] S. Zarate *et al.*, “Parliament2: Accurate structural variant calling at scale,” *GigaScience*, vol. 9, no. 12, Nov. 2020, doi: 10.1093/gigascience/giaa145.

[209] C. Chiang *et al.*, “SpeedSeq: ultra-fast personal genome analysis and interpretation,” *Nat. Methods*, vol. 12, no. 10, pp. 966–968, Oct. 2015, doi: 10.1038/nmeth.3505.

[210] I. Sugita, S. Matsuyama, H. Dobashi, D. Komura, and S. Ishikawa, “Viola: a structural variant signature extractor with user-defined classifications,” *Bioinformatics*, vol. 38, no. 2, pp. 540–542, Jan. 2022, doi: 10.1093/bioinformatics/btab662.

[211] R. E. Handsaker, J. M. Korn, J. Nemesh, and S. A. McCarroll, “Discovery and genotyping of genome structural polymorphism by sequencing on a population scale,” *Nat. Genet.*, vol. 43, no. 3, pp. 269–276, Mar. 2011, doi: 10.1038/ng.768.

[212] F. Hormozdiari, I. Hajirasouliha, A. McPherson, E. E. Eichler, and S. C. Sahinalp, “Simultaneous structural variation discovery among multiple paired-end sequenced genomes,” *Genome Res.*, vol. 21, no. 12, pp. 2203–2212, Dec. 2011, doi: 10.1101/gr.120501.111.

[213] F. Hach *et al.*, “mrsFAST: a cache-oblivious algorithm for short-read mapping,” *Nat. Methods*, vol. 7, no. 8, Art. no. 8, Aug. 2010, doi: 10.1038/nmeth0810-576.

[214] A. Abyzov *et al.*, “Analysis of deletion breakpoints from 1,092 humans reveals details of mutation mechanisms,” *Nat. Commun.*, vol. 6, p. 7256, Jun. 2015, doi: 10.1038/ncomms8256.

[215] H. Y. K. Lam *et al.*, “Nucleotide-resolution analysis of structural variants using BreakSeq and a breakpoint library,” *Nat. Biotechnol.*, vol. 28, no. 1, Art. no. 1, Jan. 2010, doi: 10.1038/nbt.1600.

[216] S. Pabinger *et al.*, “A survey of tools for variant analysis of next-generation genome sequencing data,” *Brief. Bioinform.*, vol. 15, no. 2, pp. 256–278, Mar. 2014, doi: 10.1093/bib/bbs086.

[217] A. Hatem, D. Bozdağ, A. E. Toland, and Ü. V. Çatalyürek, “Benchmarking short sequence mapping tools,” *BMC Bioinformatics*, vol. 14, no. 1, p. 184, Jun. 2013, doi: 10.1186/1471-2105-14-184.

[218] S. Canzar and S. L. Salzberg, “Short Read Mapping: An Algorithmic Tour,” *Proc. IEEE*, vol. 105, no. 3, pp. 436–458, 2017, doi: 10.1109/JPROC.2015.2455551.

[219] S. Schbath, V. Martin, M. Zytnicki, J. Fayolle, V. Loux, and J.-F. Gibrat, “Mapping Reads on a Genomic Sequence: An Algorithmic Overview and a Practical Comparative Analysis,” *J. Comput. Biol.*, vol. 19, no. 6, pp. 796–813, Jun. 2012, doi: 10.1089/cmb.2012.0022.

[220] H. Li and R. Durbin, “Fast and accurate short read alignment with Burrows-Wheeler transform,” *Bioinformatics*, vol. 25, no. 14, pp. 1754–1760, 2009, doi: 10.1093/bioinformatics/btp324.

[221] H. Li and R. Durbin, "Fast and accurate long-read alignment with Burrows-Wheeler transform," *Bioinformatics*, vol. 26, no. 5, pp. 589–595, 2010, doi: 10.1093/bioinformatics/btp698.

[222] V. Bansal, "A computational method for estimating the PCR duplication rate in DNA and RNA-seq experiments," *BMC Bioinformatics*, vol. 18, no. Suppl 3, p. 43, Mar. 2017, doi: 10.1186/s12859-017-1471-9.

[223] N. Nagarajan and M. Pop, "Sequence assembly demystified.," *Nat. Rev. Genet.*, vol. 14, no. 3, pp. 157–67, 2013, doi: 10.1038/nrg3367.

[224] M. Baker, "De novo genome assembly: what every biologist should know," *Nat. Methods*, vol. 9, no. 4, pp. 333–337, 2012, doi: 10.1038/nmeth.1935.

[225] J. T. Simpson and M. Pop, "The Theory and Practice of Genome Sequence Assembly," *Annu. Rev. Genomics Hum. Genet.*, vol. 16, no. 1, pp. 153–172, 2015, doi: 10.1146/annurev-genom-090314-050032.

[226] D. R. Zerbino and E. Birney, "Velvet: Algorithms for de novo short read assembly using de Bruijn graphs," *Genome Res.*, vol. 18, no. 5, pp. 821–829, 2008, doi: 10.1101/gr.074492.107.

[227] R. Luo *et al.*, "SOAPdenovo2: an empirically improved memory-efficient short-read de novo assembler," *GigaScience*, vol. 1, no. 1, pp. 2047-217X-1–18, Dec. 2012, doi: 10.1186/2047-217X-1-18.

[228] W. Huang, L. Li, J. R. Myers, and G. T. Marth, "ART: A next-generation sequencing read simulator," *Bioinformatics*, vol. 28, no. 4, pp. 593–594, 2012, doi: 10.1093/bioinformatics/btr708.

[229] X. Hu *et al.*, "pIRS: Profile-based Illumina pair-end reads simulator," *Bioinformatics*, vol. 28, no. 11, pp. 1533–1535, Jun. 2012, doi: 10.1093/bioinformatics/bts187.

[230] "SAM/BAM and related specifications." samtools, Dec. 19, 2022. Accessed: Dec. 29, 2022. [Online]. Available: https://github.com/samtools/hts-specs

[231] M. G. Ross *et al.*, "Characterizing and measuring bias in sequence data," *Genome Biol.*, vol. 14, no. 5, p. R51, May 2013, doi: 10.1186/gb-2013-14-5-r51.

[232] D. Sims, I. Sudbery, N. E. Ilott, A. Heger, and C. P. Ponting, "Sequencing depth and coverage: key considerations in genomic analyses," *Nat. Rev. Genet.*, vol. 15, no. 2, pp. 121–132, Feb. 2014, doi: 10.1038/nrg3642.

[233] Y. Benjamini and T. P. Speed, "Summarizing and correcting the GC content bias in high-throughput sequencing," *Nucleic Acids Res.*, vol. 40, no. 10, pp. 1–14, 2012, doi: 10.1093/nar/gks001.

[234] D. Aird *et al.*, "Analyzing and minimizing PCR amplification bias in Illumina sequencing libraries," *Genome Biol.*, vol. 12, no. 2, p. R18, 2011, doi: 10.1186/gb-2011-12-2-r18.

[235] S. Gunasekera *et al.*, "Evaluating coverage bias in next-generation sequencing of Escherichia coli," *PLOS ONE*, vol. 16, no. 6, p. e0253440, 6 2021, doi: 10.1371/journal.pone.0253440.

[236] F. Zare, A. Hosny, and S. Nabavi, “Noise cancellation using total variation for copy number variation detection,” *BMC Bioinformatics*, vol. 19, no. Suppl 11, p. 361, Oct. 2018, doi: 10.1186/s12859-018-2332-x.

[237] M.-S. Cheung, T. A. Down, I. Latorre, and J. Ahringer, “Systematic bias in high-throughput sequencing data and its correction by BEADS,” *Nucleic Acids Res.*, vol. 39, no. 15, p. e103, Aug. 2011, doi: 10.1093/nar/gkr425.

[238] J. D. Wang and P. A. Levin, “Metabolism, cell growth and the bacterial cell cycle,” *Nat. Rev. Microbiol.*, vol. 7, no. 11, pp. 822–827, Nov. 2009, doi: 10.1038/nrmicro2202.

[239] H. Luo and F. Gao, “DoriC 10.0: an updated database of replication origins in prokaryotic genomes including chromosomes and plasmids,” *Nucleic Acids Res.*, vol. 47, no. D1, pp. D74–D77, Jan. 2019, doi: 10.1093/nar/gky1014.

[240] S. M. Lesch and D. R. Jeske, “Some Suggestions for Teaching About Normal Approximations to Poisson and Binomial Distribution Functions,” *Am. Stat.*, vol. 63, no. 3, pp. 274–277, Aug. 2009, doi: 10.1198/tast.2009.08147.

[241] R. Jugas, M. Vitek, D. Maderankova, and H. Skutkova, “Signal Processing Based CNV Detection in Bacterial Genomes,” in *Bioinformatics and Biomedical Engineering*, I. Rojas, O. Valenzuela, F. Rojas, and F. Ortuño, Eds., in Lecture Notes in Computer Science. Cham: Springer International Publishing, 2019, pp. 93–102. doi: 10.1007/978-3-030-17938-0_9.

[242] B. lglewicz and D. C. Hoaglin, *How to Detect and Handle Outliers*, vol. 1993. in The ASQC Basic References in Quality Control: Statistical Techniques, no. Volume 16, vol. 1993. Milwaukee, Wisconsin, USA: American Society for Quality Control (ASQC. Accessed: Dec. 19, 2022. [Online]. Available: https://hwbdocuments.env.nm.gov/Los%20Alamos%20National%20Labs/TA%2054/11587.pdf

[243] D. M. Hawkins, *Identification of Outliers*. Dordrecht: Springer Netherlands, 1980. doi: 10.1007/978-94-015-3994-4.

[244] The National Institute of Standards and Technology (NIST), “NIST/SEMATECH e-Handbook of Statistical Methods,” 2012. https://doi.org/10.18434/M32189 (accessed Dec. 19, 2022).

[245] M. S. Lindner, M. Kollock, F. Zickmann, and B. Y. Renard, “Analyzing genome coverage profiles with applications to quality control in metagenomics,” *Bioinforma. Oxf. Engl.*, vol. 29, no. 10, pp. 1260–1267, May 2013, doi: 10.1093/bioinformatics/btt147.

[246] R. Jugas *et al.*, “CNproScan: Hybrid CNV detection for bacterial genomes,” *Genomics*, vol. 113, no. 5, pp. 3103–3111, Sep. 2021, doi: 10.1016/j.ygeno.2021.06.040.

[247] “Babraham Bioinformatics - FastQC A Quality Control tool for High Throughput Sequence Data.” https://www.bioinformatics.babraham.ac.uk/projects/fastqc/ (accessed Dec. 28, 2022).

[248] S. Chen, Y. Zhou, Y. Chen, and J. Gu, “fastp: an ultra-fast all-in-one FASTQ preprocessor,” *Bioinformatics*, vol. 34, no. 17, pp. i884–i890, Sep. 2018, doi: 10.1093/bioinformatics/bty560.

[249] F. Krueger, “Trim Galore! A wrapper around Cutadapt and FastQC to consistently apply adapter and quality trimming to FastQ files, with extra functionality for RRBS data.,” *https://www.bioinformatics.babraham.ac.uk/projects/trim_galore*. 2012. [Online]. Available: https://github.com/FelixKrueger/TrimGalore

[250] H. Li *et al.*, “The Sequence Alignment/Map format and SAMtools,” *Bioinformatics*, vol. 25, no. 16, pp. 2078–2079, 2009, doi: 10.1093/bioinformatics/btp352.

[251] C. Pockrandt, M. Alzamel, C. S. Iliopoulos, and K. Reinert, “GenMap: ultra-fast computation of genome mappability,” *Bioinforma. Oxf. Engl.*, vol. 36, no. 12, pp. 3687–3692, Jun. 2020, doi: 10.1093/bioinformatics/btaa222.

[252] R. Copin *et al.*, “Sequential evolution of virulence and resistance during clonal spread of community-acquired methicillin-resistant Staphylococcus aureus,” *Proc. Natl. Acad. Sci.*, vol. 116, no. 5, pp. 1745–1754, Jan. 2019, doi: 10.1073/pnas.1814265116.

[253] T. Horinouchi *et al.*, “Prediction of Cross-resistance and Collateral Sensitivity by Gene Expression profiles and Genomic Mutations,” *Sci. Rep.*, vol. 7, no. 1, Art. no. 1, Oct. 2017, doi: 10.1038/s41598-017-14335-7.

[254] J. Wang *et al.*, “Genome adaptive evolution of Lactobacillus casei under long-term antibiotic selection pressures,” *BMC Genomics*, vol. 18, no. 1, p. 320, Apr. 2017, doi: 10.1186/s12864-017-3710-x.

[255] M. Bezdicek *et al.*, “Application of mini-MLST and whole genome sequencing in low diversity hospital extended-spectrum beta-lactamase producing Klebsiella pneumoniae population,” *PLoS ONE*, vol. 14, no. 8, pp. 1–14, 2019, doi: 10.1371/journal.pone.0221187.

[256] T. Derrien *et al.*, “Fast Computation and Applications of Genome Mappability,” *PLOS ONE*, vol. 7, no. 1, p. e30377, Spring 2012, doi: 10.1371/journal.pone.0030377.

[257] F. Mölder *et al.*, “Sustainable data analysis with Snakemake.” F1000Research, Apr. 19, 2021. doi: 10.12688/f1000research.29032.2.

LIST OF ABBREVIATIONS

1KGP	1000 genomes project
ARP	anomalously mapped read pair
AS	assembly approach
BAC	bacterial artificial chromosome
BAF	B allele frequency
BAM	Binary Alignment Map
BED	browser extensible data
BIR	break-induced replication
bp	Base-pair
CBS	circular binary segmentation
cDNA	coding DNA
CGH	comparative genomic hybridization
CNA	copy number alteration
CNV	copy number variation
COSMIC	Catalog Of Somatic Mutations In Cancer
cxSV	complex SV
dbVar	NCBI's database of human genomic Structural Variation
DEL	deletion
DGV	The Database of Genomic Variants
DGVa	Database of Genomic Variants archive
DNA	deoxyribonucleic acid
DSB	double-stranded break
dsDNA	double-stranded DNA
DUP	duplication
EBI	European Bioinformatics Institute
ESD	extreme studentized deviate
ESP	end-sequence profiling
EVA	European Variation Archive
FISH	Fluorescence in situ hybridization
FoSTeS	fork stalling and template switching
GESD	generalized extreme studentized deviate
GIAB	Genome in the Bottle
gnomAD	Genome Aggregation Database
HGT	horizontal gene transfer
HMM	Hidden Markov Model
HR	homologous recombination
indel	insertion-deletion
INS	insertion
INV	inversion
LCR	low-copy repeats
LINE	long interspersed element
MAD	median absolute deviation
mCNV	multiallelic CNV

MEI	mobile element insertion
MGE	Mobile genetic element
MMBIR	microhomology-mediated break-induced replication
MMBIR	microhomology-mediated break-induced replication
NAHR	nonallelic homologous recombination
NCBI	National Center for Biotechnology Information
NGS	next-generation sequencing
NHEJ	non-homologous end joining
NMEJ	non-homologous end joining
nt	nucleotide
NUMT	nuclear insertion of the mitochondrial genome
oriC	replication origin
PCR	polymerase chain reaction
PDF	probability distribution function
RD	read depth approach
RP	read pair approach
SAM	Sequence Alignment Map
SD	segmental duplication
SINE	short interspersed element
SMRT	single-molecule real-time
SNP	Single Nucleotide Polymorphism
SNV	single nucleotide variants
SR	split read approach
ssDNA	single-stranded DNA
STR	short tandem repeats
SV	structural variation
TE	Transposable elements
TGS	third generation sequencing
TLEN	Template length, BAM field
TSV	tab-separated values
VAF	variant allele frequency
VCF	variant call format
VEI	viral insertion
VNTR	variable number tandem repeats
WES	whole exome sequencing
WGS	whole genome sequencing
ZMW	zero-mode waveguides

LIST OF FIGURES

LIST OF TABLES

LIST OF APPENDICES

Appendix A - Curriculum vitae

Personal Information	
Name / Surname/ Title	Ing. Robin Jugas
E-mail	robinjugas@gmail.com
Nationality	Czech
Date of birth	19.03.1991
Education	
Doctoral Study	2016-so far, Biomedical Technology and Bioinformatics
Name of the organization	Brno University of Technology, Department of Biomedical Engineering
Thesis theme	Utilization of Genomic Signal Processing Techniques for De-Novo Assembly and Detection of Structural Variations in Bacterial Genomes Supervisor: Ing. Helena Škutková, Ph.D.
Graduate study	2014-2016, Biomedical Engineering and Bioinformatics, Ing.
Name of the organization	Brno University of Technology, Department of Biomedical Engineering
Thesis theme	Signal Processing Based Methods for Genome Assembly Refinement Supervisor: Mgr. Ing. Karel Sedlář, Ph.D.
Undergraduate study	2011-2014, Biomedical Engineering and Bioinformatics, Bc.
Name of the organization	Brno University of Technology, Department of Biomedical Engineering
Thesis theme	Digital Processing of Plant Genomes Supervisor: Mgr. Ing. Karel Sedlář
Work experience	
Dates, Position	09/2020 – so far, Bioinformatician
Main activities and responsibilities	Bioinformatics analysis of Human methylation data, Human genomes variant, and SV calling, RNA array data analysis
Name of employer	CEITEC MU, Ondřej Slabý Research Group
Dates, Position	07/2014– 08/2015 IT support
Main activities and responsibilities	Linux server administrator, network administrator, HW maintenance
Name of employer	SDB Brno (Youth free-time center)
Training courses	
	2018 NGS School
	2017 Summer school of mathematical biology IBA MU
Academic/scientific activities	
Scientific impact	WoS records: 7, h-index: 2, citations number: 15
Researcher ID	O-4299-2017
Google Scholar	https://scholar.google.com/citations?user=9BPJp_wAAAAJ
ORCID	0000-0003-4675-0985
GitHub	https://github.com/robinjugas

Appendix B - List of own publications

Journal Articles

JUGAS R, SEDLAR K, VITEK M, NYKRYNOVA M, BARTON V, BEZDICEK M, LENGEROVA M, SKUTKOVA H. CNproScan: Hybrid CNV detection for bacterial genomes. Genomics. 2021 Sep;113(5):3103-3111. doi: 10.1016/j.ygeno.2021.06.040. Epub 2021 Jul 3. PMID: 34224809 IF 4.31 Q2 (2021)

MADERANKOVA D, JUGAS R, SEDLAR K, VITEK M, SKUTKOVA H. Rapid Bacterial Species Delineation Based on Parameters Derived From Genome Numerical Representations. Comput Struct Biotechnol J. 2019 Jan 9;17:118-126. doi: 10.1016/j.csbj.2018.12.006. PMID: 30728919; PMCID: PMC6352304. IF 6.18 Q1 (2019)

SKUTKOVA H, MADERANKOVA D, SEDLAR K, JUGAS R, VITEK M. A degeneration-reducing criterion for optimal digital mapping of genetic codes. Comput Struct Biotechnol J. 2019 Mar 19;17:406-414. doi: 10.1016/j.csbj.2019.03.007. PMID: 30984363; PMCID: PMC6444178. IF 6.18 Q1 (2019)

Conference Proceedings

JUGAS, R.; VÍTEK, M.; MADĚRÁNKOVÁ, D.; ŠKUTKOVÁ, H. Signal processing based CNV detection in bacterial genomes. In Bioinformatics and Biomedical Engineering. IWBBIO 2019. Lecture Notes in Computer Science. Granada, Spain: Springer Verlag, 2019. p. 93-102. ISBN: 978-3-030-17937-3. ISSN: 0302-9743.

JUGAS, R. Application of Optimization Algorithms to the Genome Assembly. In Proceedings of the 24th Conference STUDENT EEICT 2018. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních, 2018. p. 595-599. ISBN: 978-80-214-5614-3.

JUGAS, R.; VÍTEK, M.; SEDLÁŘ, K.; ŠKUTKOVÁ, H. Cross-Correlation Based Detection of Contigs Overlaps. In MIPRO 2018 proceedings. New York: IEEE, 2018. p. 155-158. ISBN: 978-953-233-095-3.

JUGAS, R. DNA reads feature extraction analysis. In Proceedings of IEEE Student Branch Conference Mikulov 2017. 2017. p. 33-36. ISBN: 978-80-214-5526-9.

JUGAS, R.; SEDLÁŘ, K.; ŠKUTKOVÁ, H.; VÍTEK, M. Overlap detection for a genome assembly based on genomic signal processing. In 2017 IEEE 30th International Symposium on Computer-Based Medical Systems. Proceedings of the ... IEEE International Symposium on Computer-Based Medical Systems. USA: IEEE Computer Society, 2017. p. 301-305. ISBN: 978-1-5386-1710-6. ISSN: 2372-9198.

JUGAS, R. Číslicové zpracování genomických signálů pro klasifikaci rostlin. In Proceedings of the 20th conference Student EEICT 2014. Brno: LITERA, 2014. p. 132-134. ISBN: 978-80-214-4922-0.

Posters

JUGAS, R.; SEDLÁŘ, K.; VÍTEK, M.; ŠKUTKOVÁ, H. CNV Hybrid Detection in Bacteria Using Signal Processing. International Conference of the Genetics Society of Korea 2019: New Horizons of Genetics. 1. Seoul, Republic of Korea: The Genetics Society of Korea, 2019. p. 101-101.

Appendix C - Supplementary Figures

Coverage Normality

Figure A1 - Q-Q plots of artificial samples

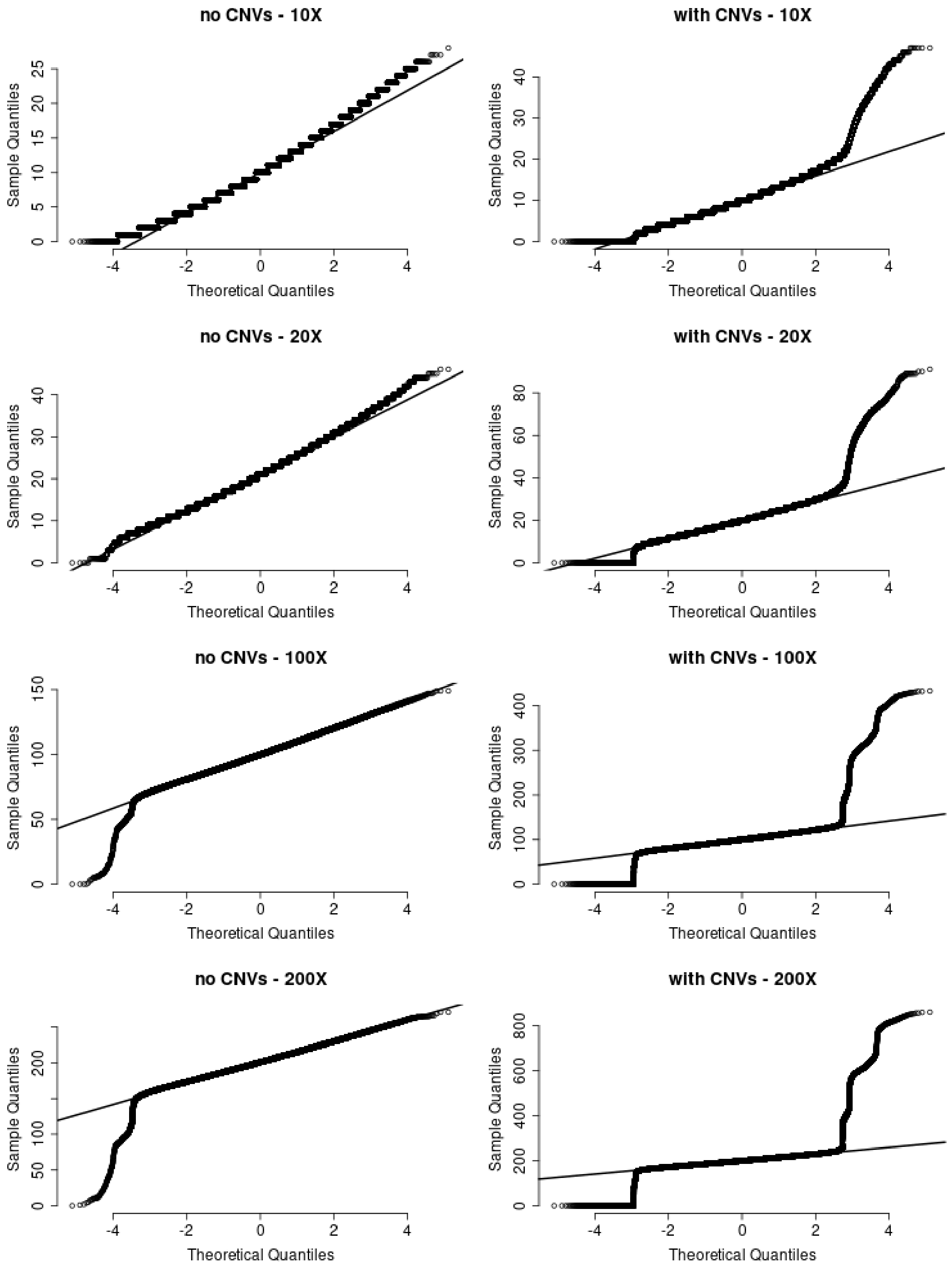


Figure A2 - Histograms of artificial samples

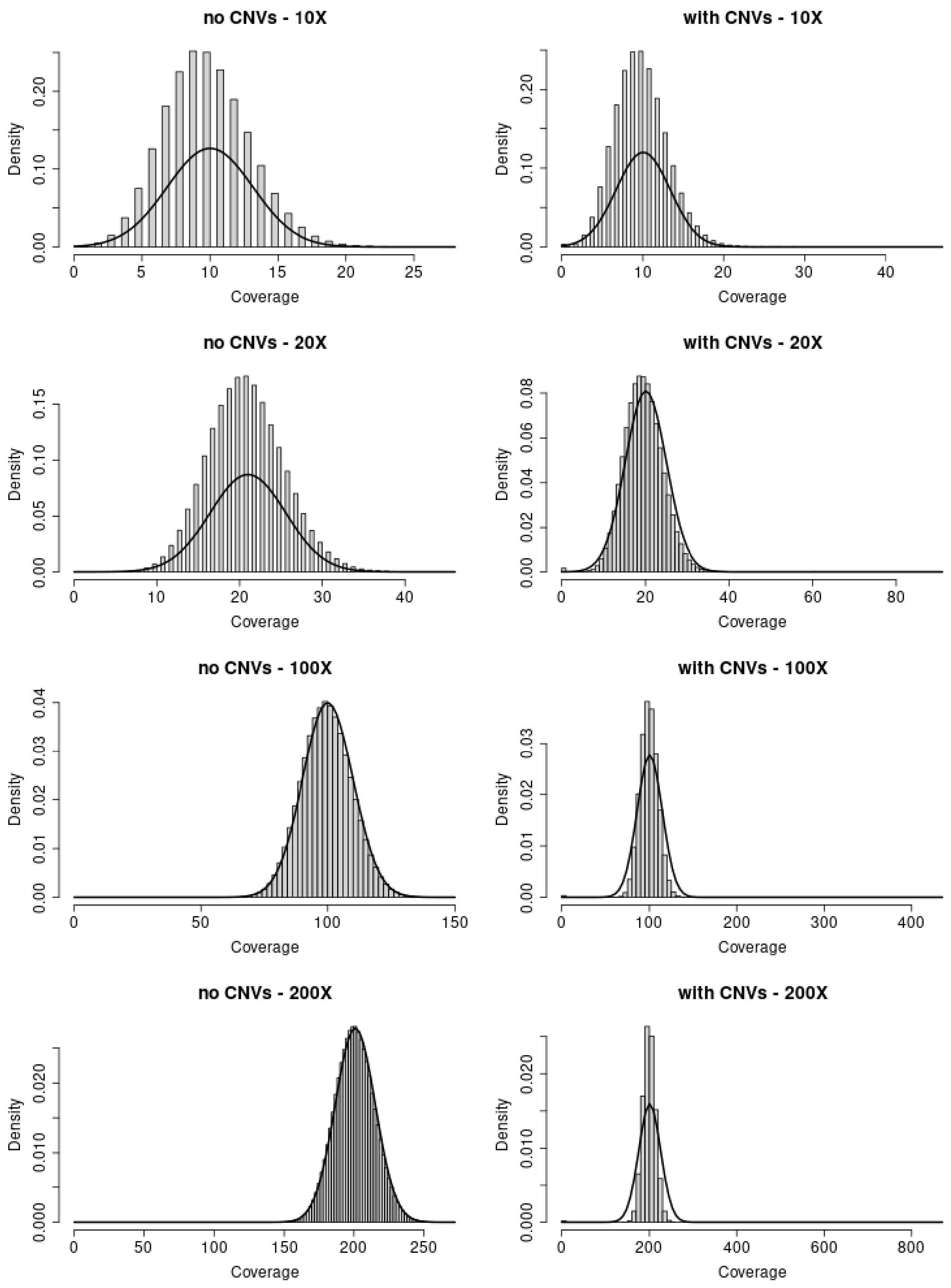


Figure A3 - Q-Q plots of real samples

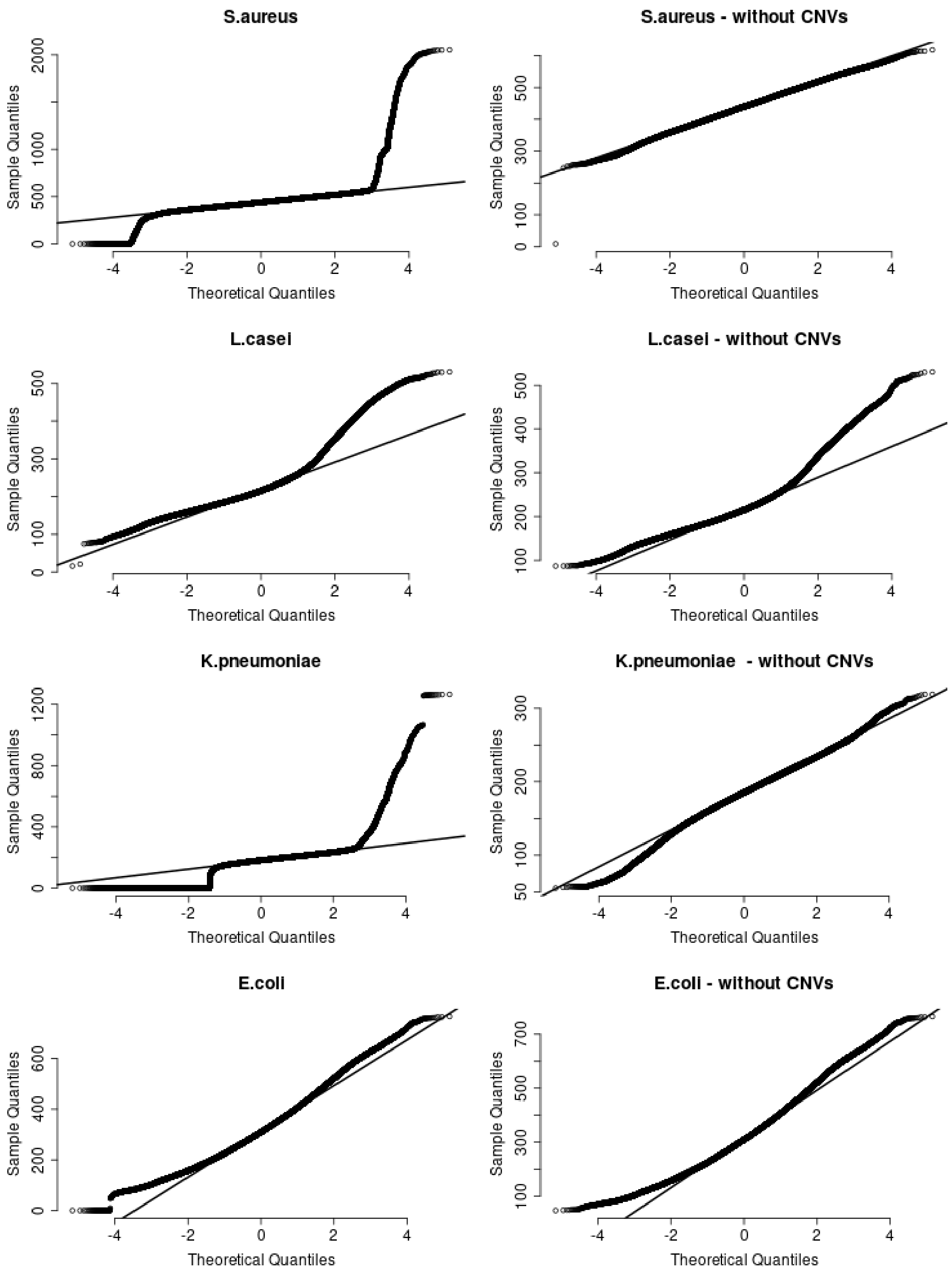
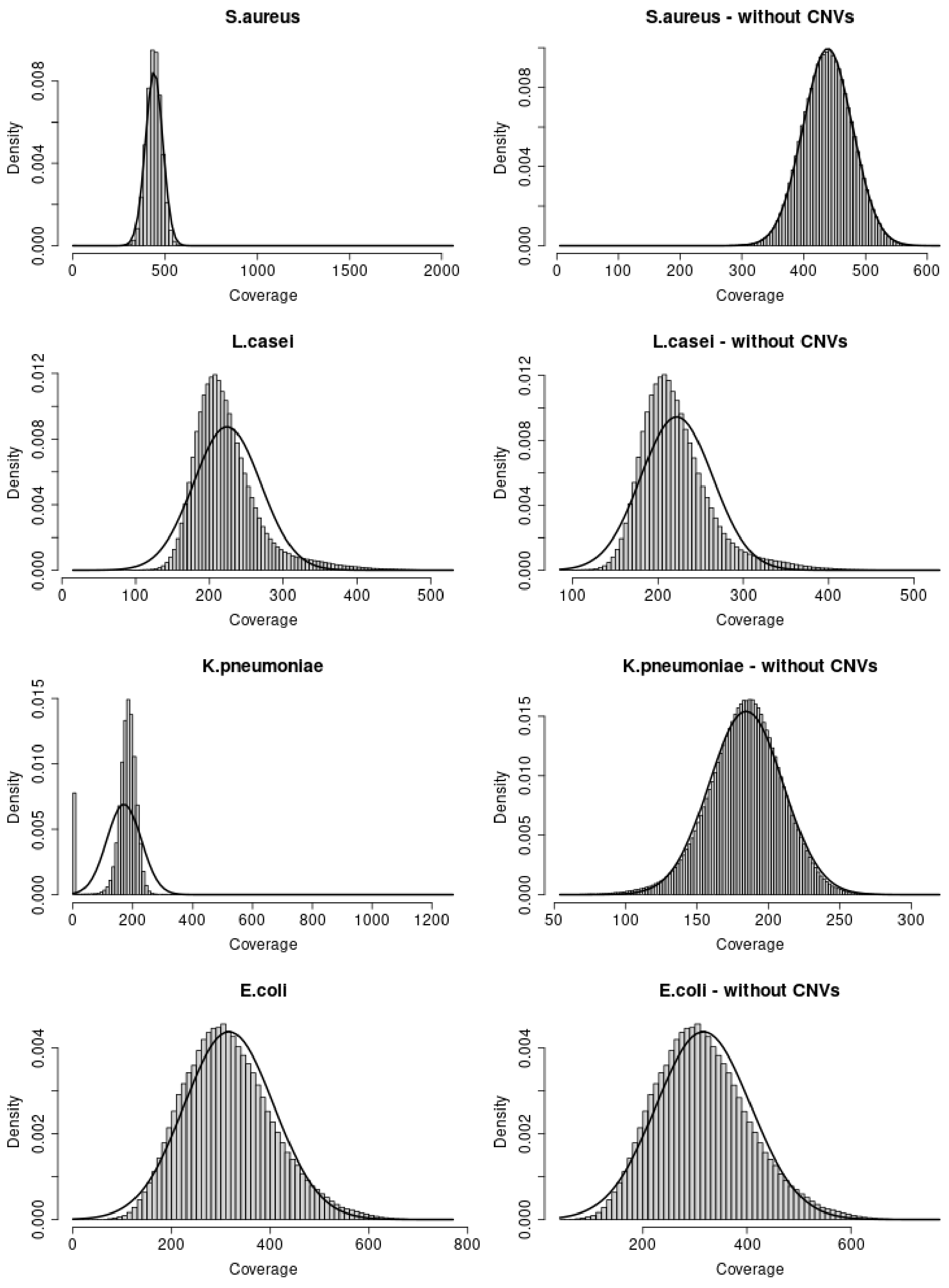


Figure A4 - Histograms of real samples



Normalizations

Figure B1 - GC normalization for *E. coli*

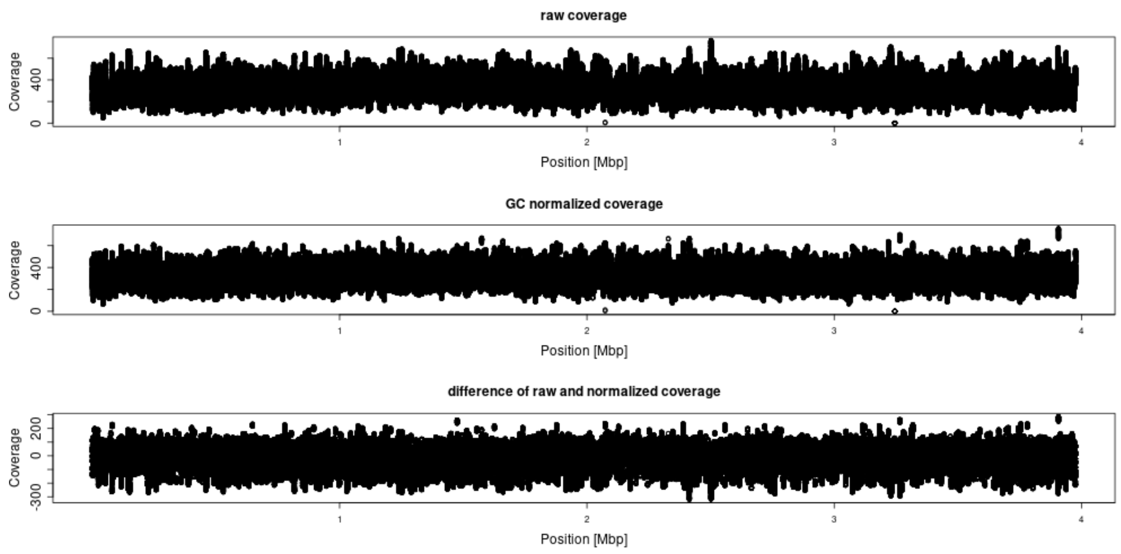


Figure B2 - GC normalization for *K. pneumoniae*

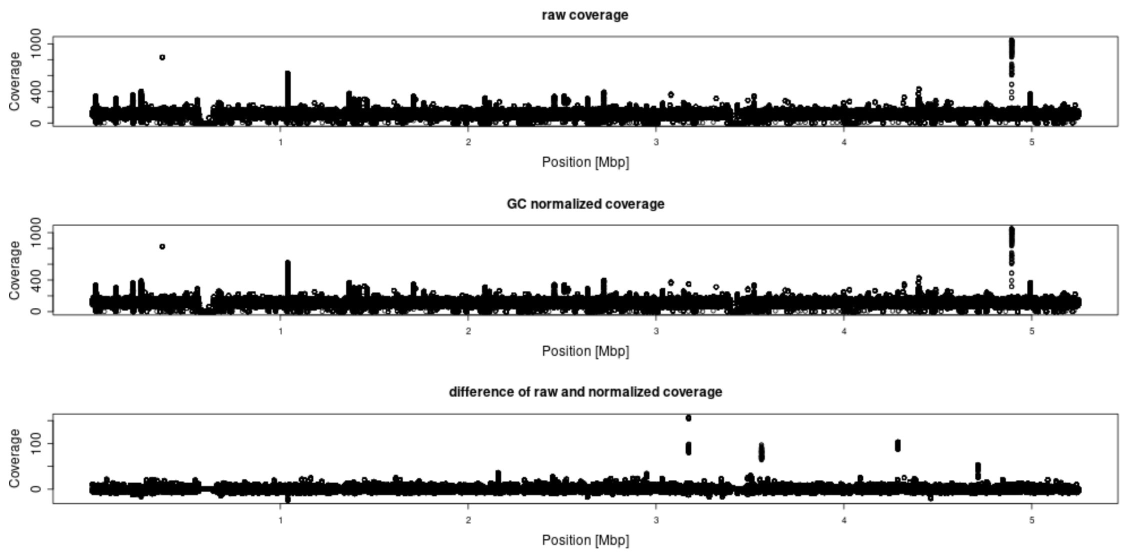


Figure B3 - GC normalization for *L. casei*

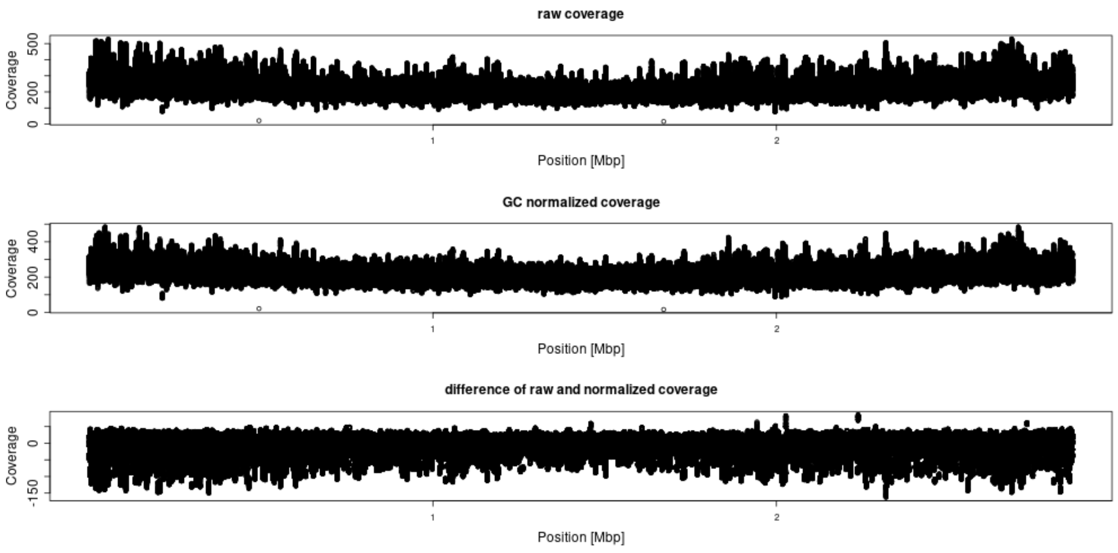


Figure B4 - GC normalization for *S. aureus*

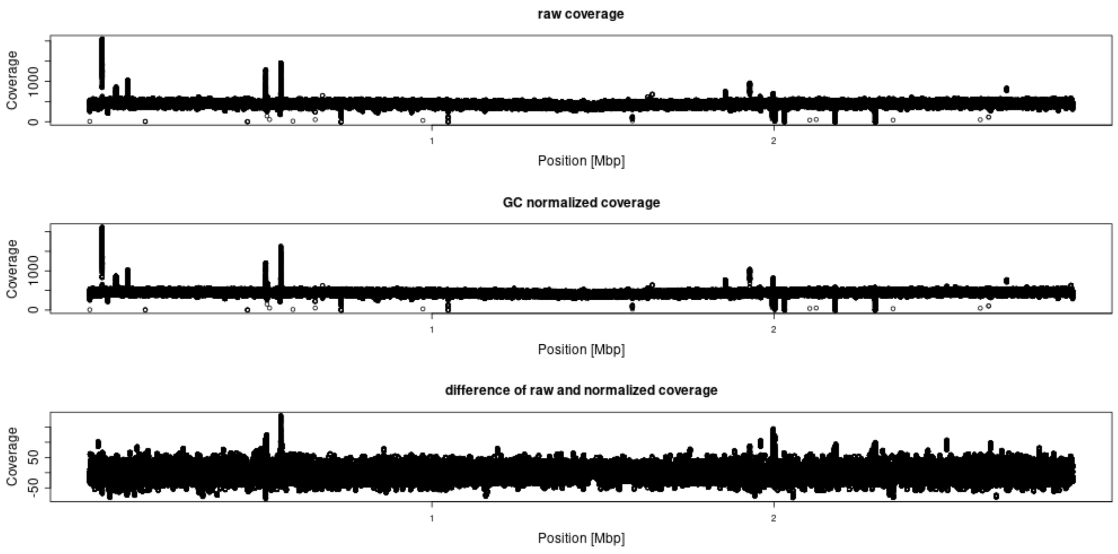


Figure C1 - mappability normalization for *E. coli*

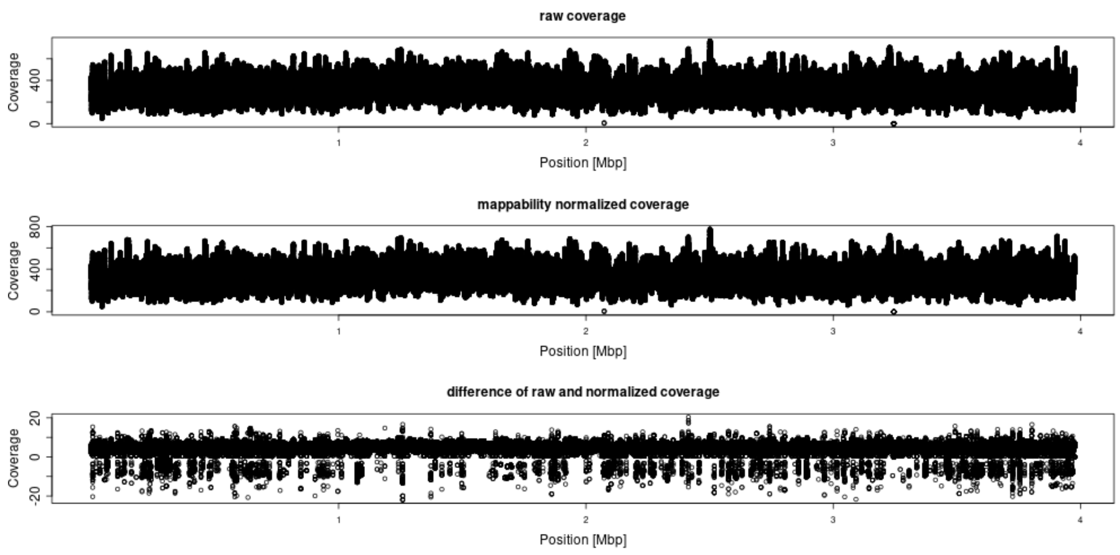


Figure C2 - mappability normalization for *K. pneumoniae*

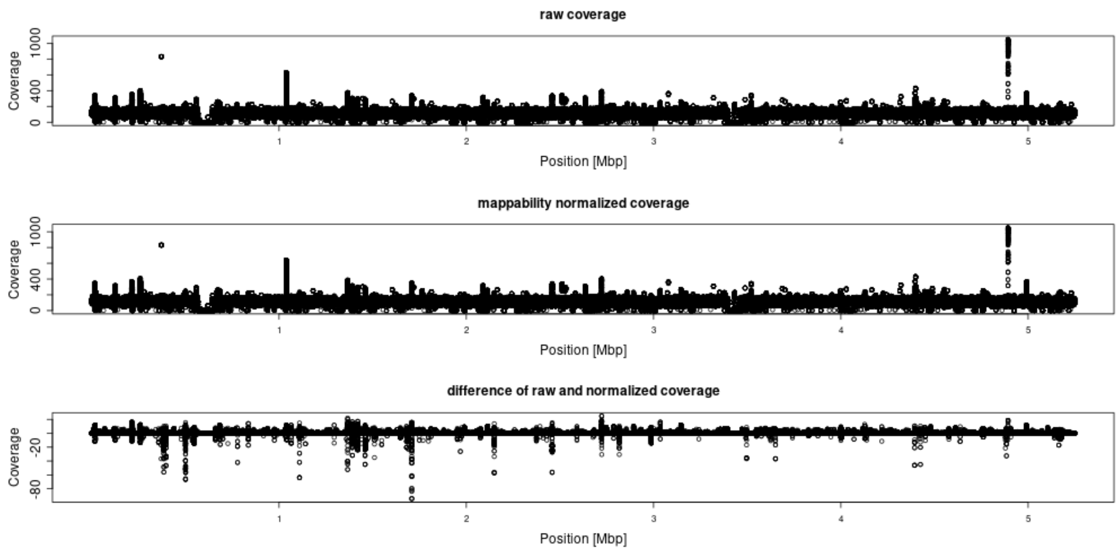


Figure C3 - mappability normalization for *L. casei*

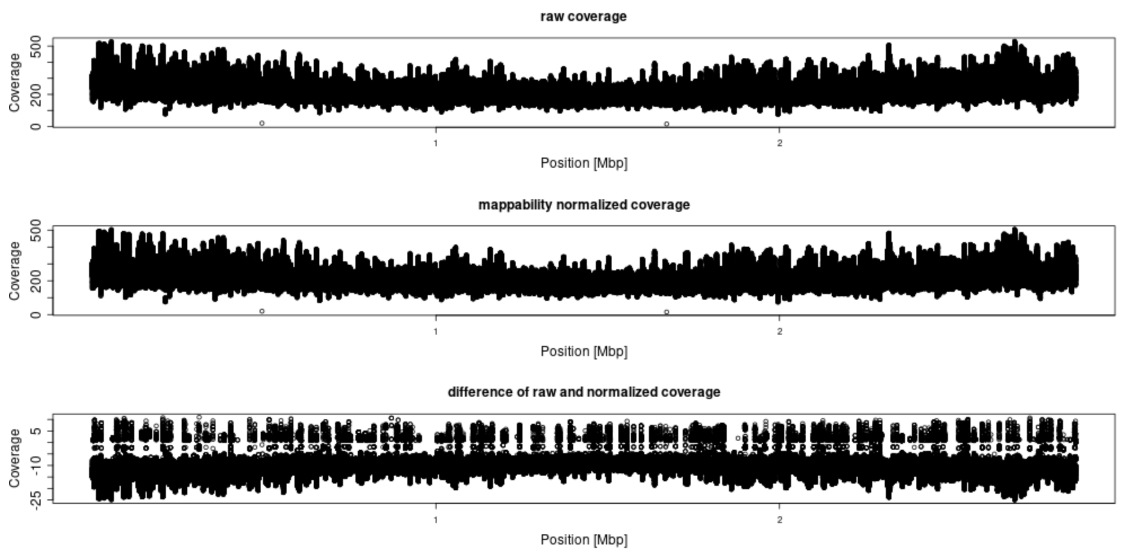


Figure C4 - mappability normalization for *S. aureus*

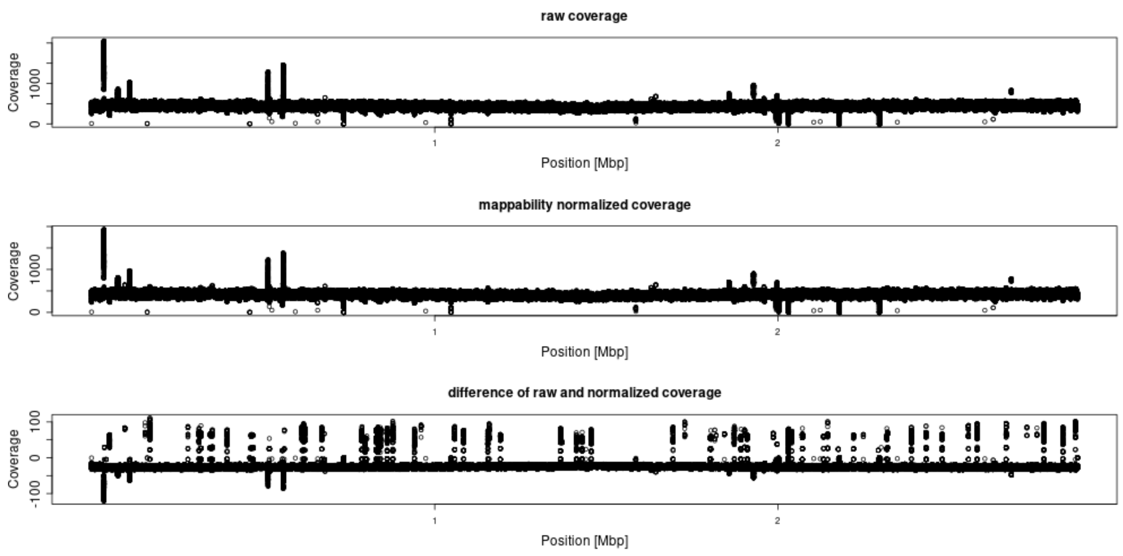


Figure D1 - replication bias normalization for *E. coli*

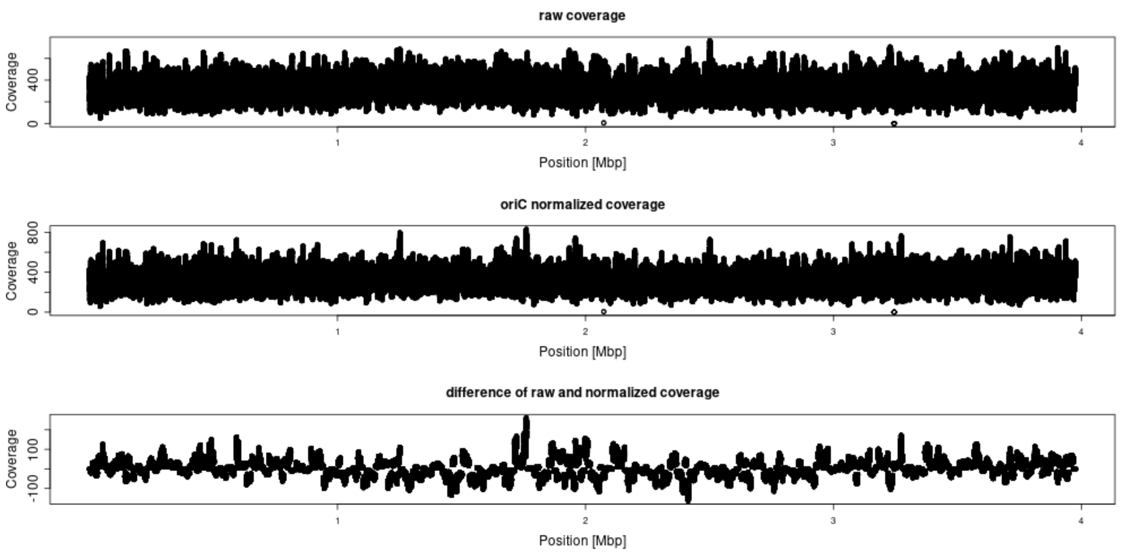


Figure D2 - replication bias normalization for *K. pneumoniae*

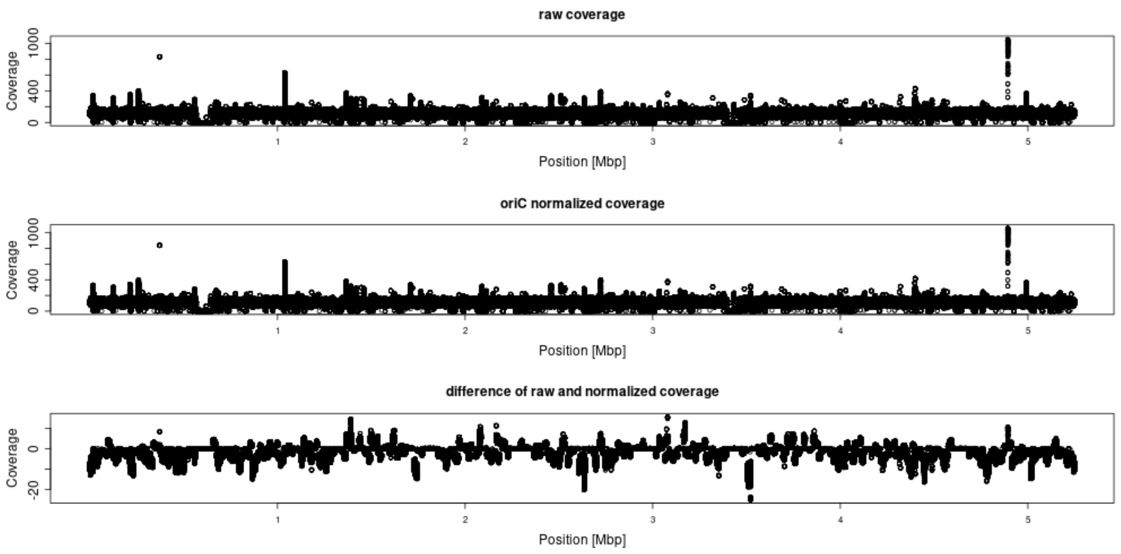


Figure D3 - replication bias normalization for *L. casei*

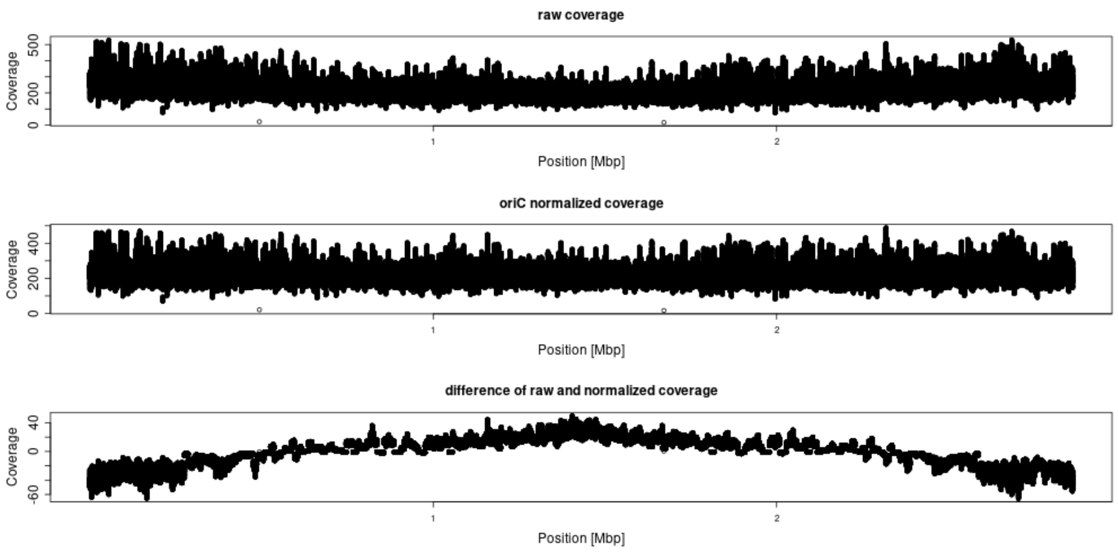
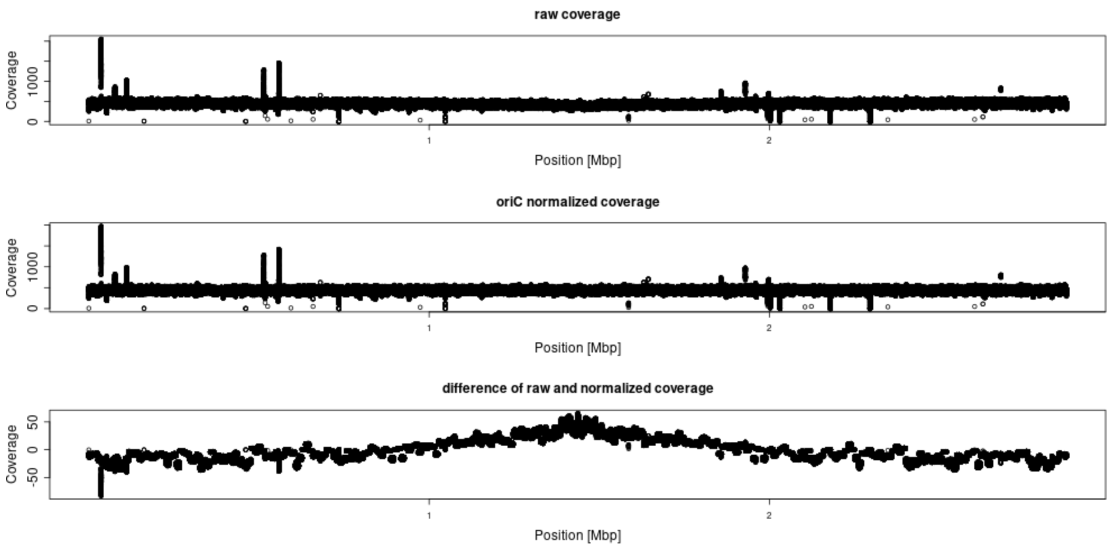


Figure D4 - replication bias normalization for *S. aureus*



Read-pair signatures

Figure E1 - IGV CNV #6 - Deletion of 828 bp has many signature reads of both orientations +/- and -/+

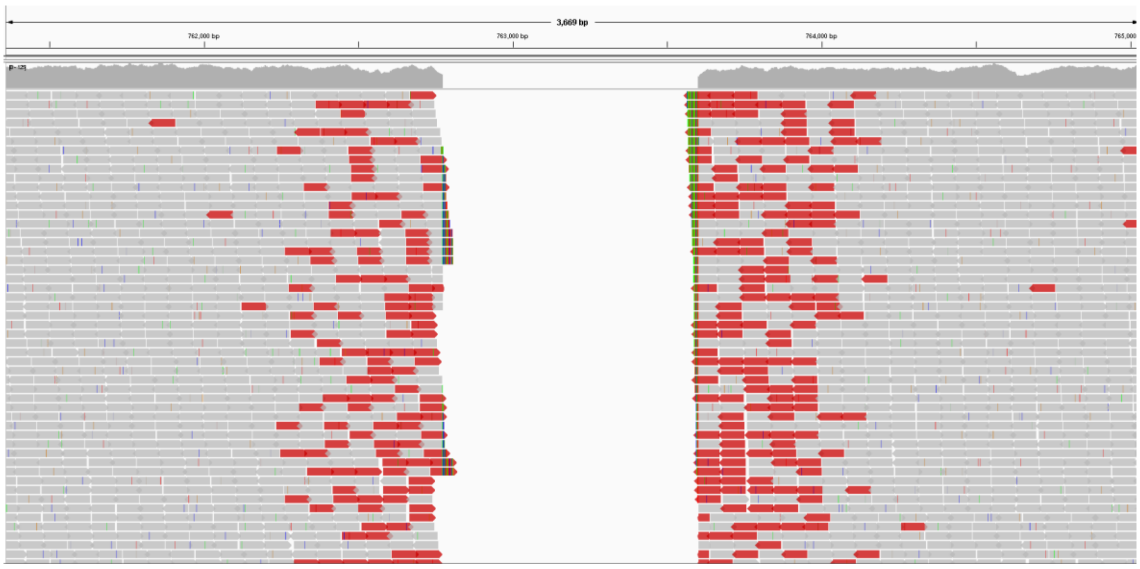
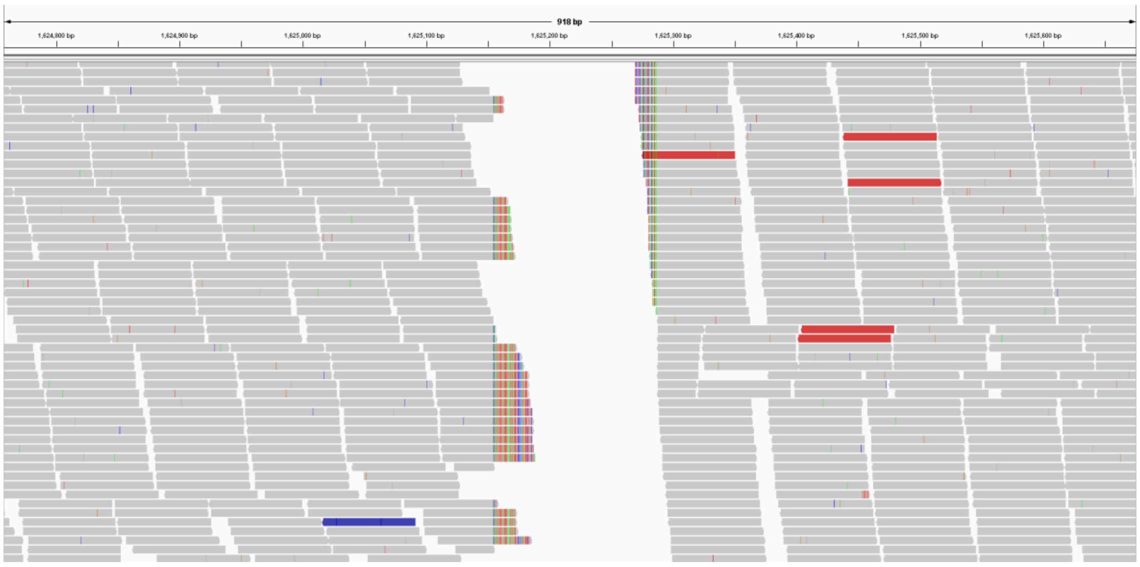


Figure E2 - IGV CNV #13 - Deletion of 134 bp creates only slip-read signatures and almost no read-pair signatures



ProcaryaSV Results

Figure F1 - *E. coli* CNV size by type

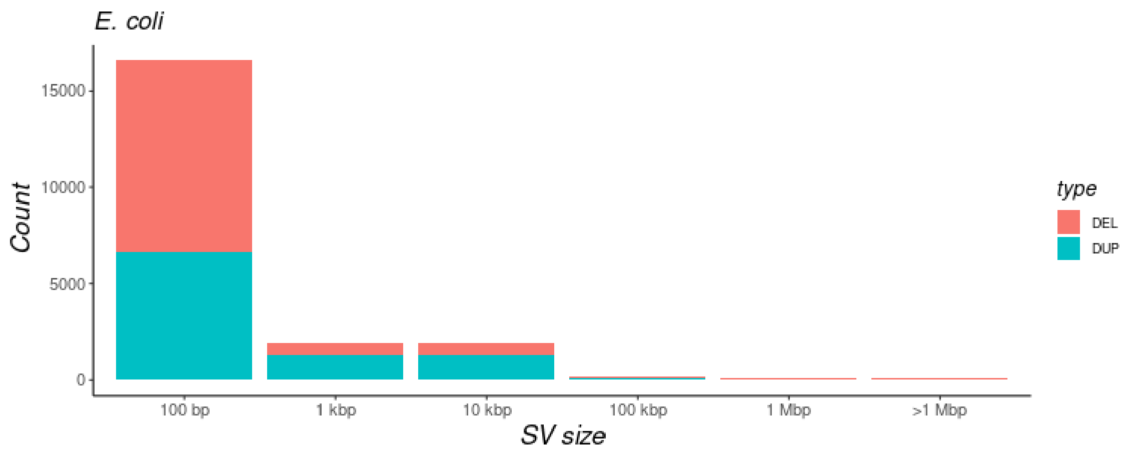


Figure F2 - *E. coli* CNV size by caller

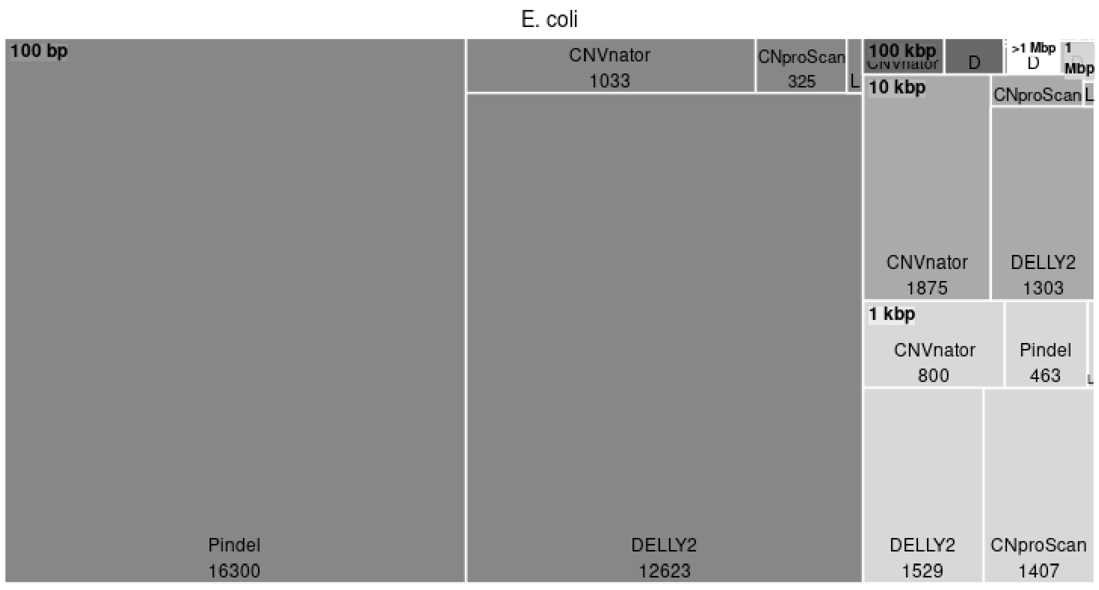


Figure G1 - *S. aureus* CNV size by type

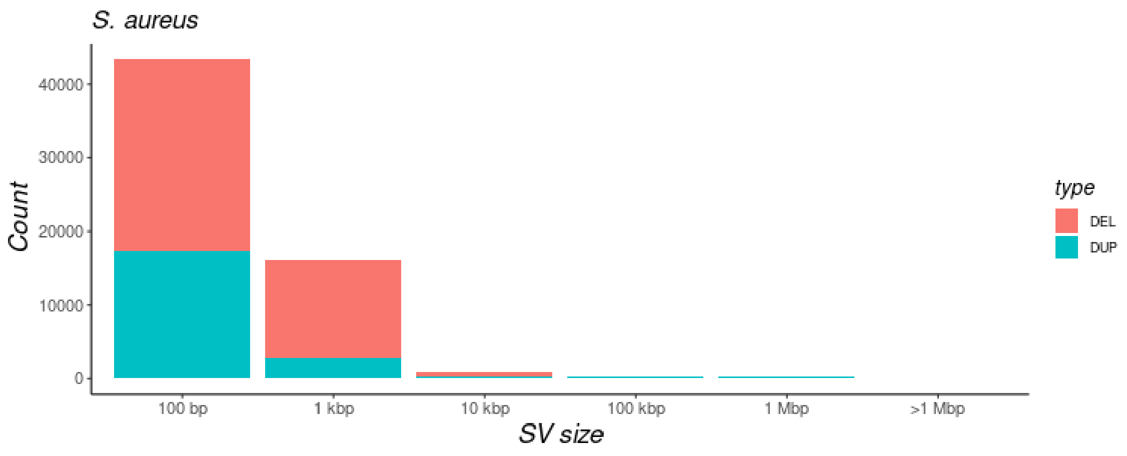


Figure G2 - *S. aureus* CNV size by caller

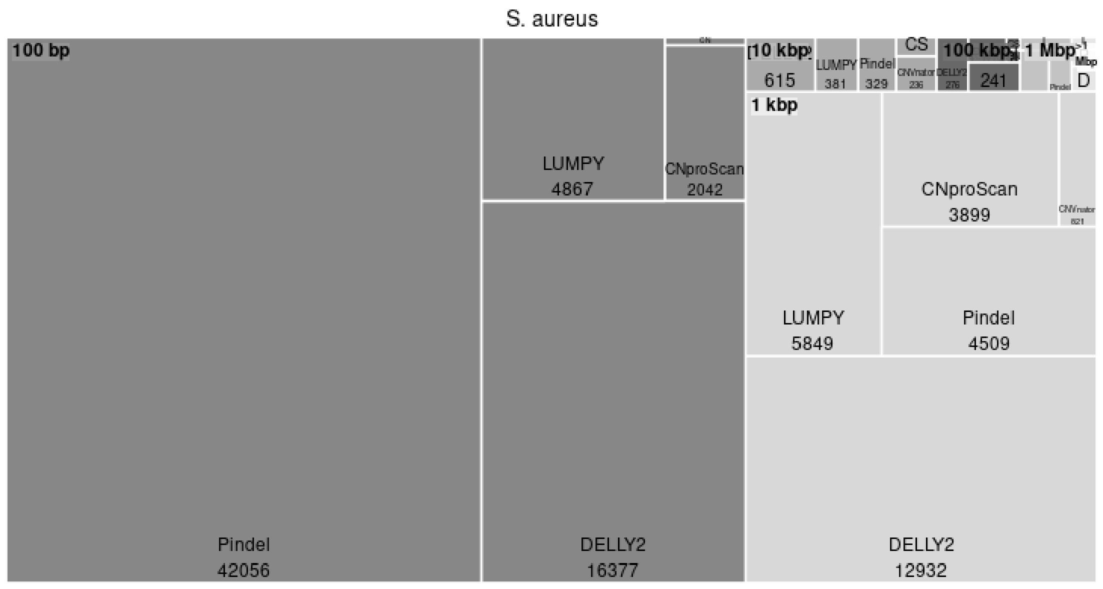


Figure H1 - *L. casei* CNV size by type

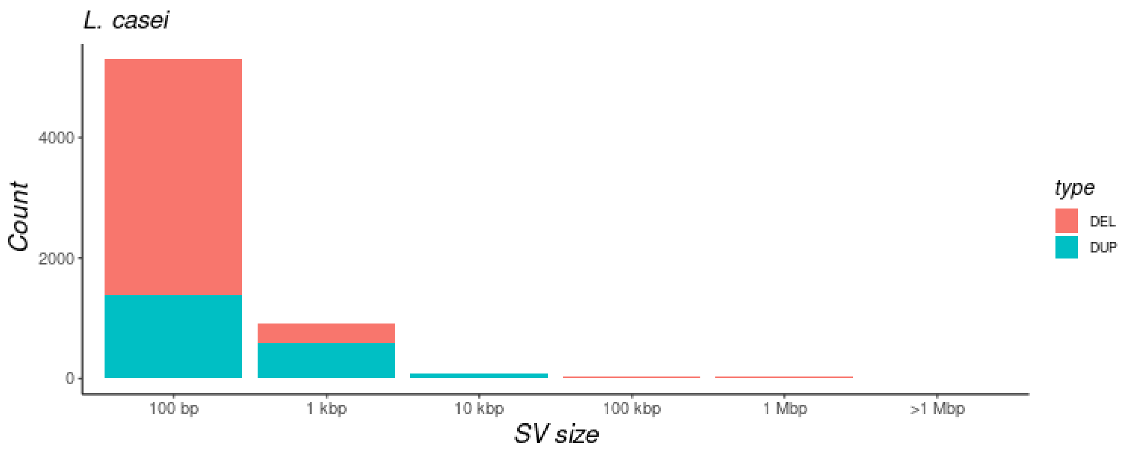
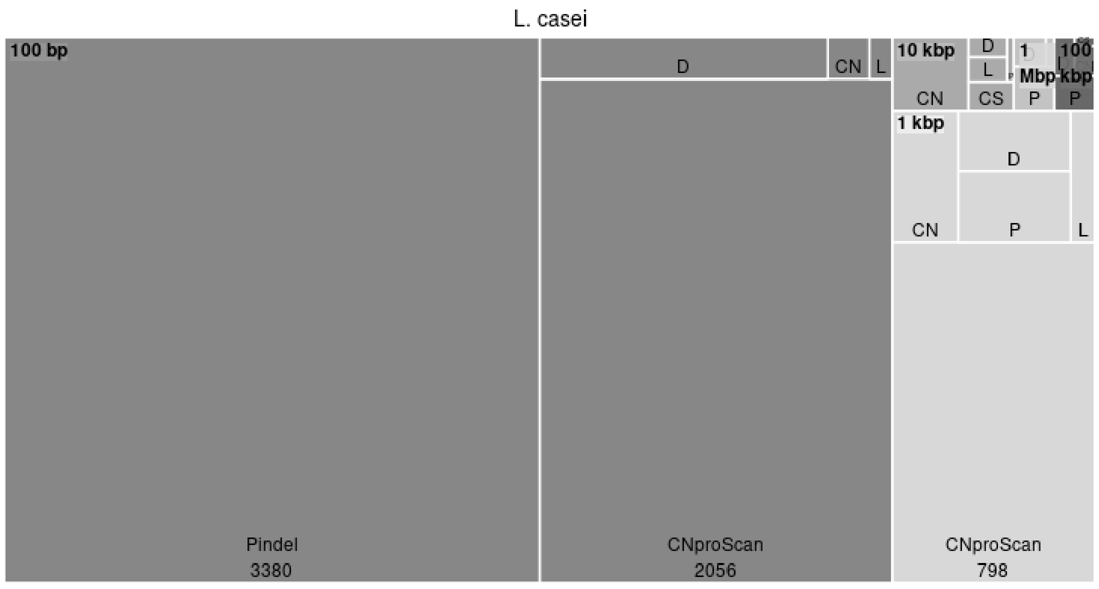


Figure H2 - *L. casei* CNV size by caller



Appendix D - Supplementary Tables

Supplementary Table 1 - Tools Overview		
Tool	**Status**	**Webpage/ Repository**
Read-pair approach tools		
MoDIL	deprecated*	http://compbio.cs.toronto.edu/modil/
VariationHunter	NA, replaced by TARDIS	http://compbio.cs.sfu.ca/strvar.htm
VariationHunter 2	NA, replaced by TARDIS	https://compbio.cs.sfu.ca/strvar.htm
PEMer	deprecated	http://sv.gersteinlab.org/pemer/
BreakDancer	not available	http://genome.wustl.edu/tools/cancer-genomics/
Split-read approach tools		
Pindel	deprecated but actively used	https://gmt.genome.wustl.edu/packages/pindel/ https://github.com/genome/pindel
Splitread	deprecated	http://splitread.sourceforge.net/
ClipCrop	only method	
CREST	deprecated	https://www.stjude.org/research/labs/zhang-lab/crest.html
Socrates	not available	http://bioinf.wehi.edu.au/socrates
SLOPE	not available	http://www-genepi.med.utah.edu/suppl/SLOPE/index.html
Read-depth approach tools		
RDxplorer	deprecated	https://rdxplorer.sourceforge.net/
mrFAST	deprecated	https://github.com/BilkentCompGen/mrfast
CNV-seq	deprecated	https://github.com/hliang/cnv-seq
cn.MOPS	maintained	https://bioconductor.org/packages/release/bioc/html/cn.mops.html
Sepúlveda	method	
CNAseg	not available	http://www.compbio.group.cam.ac.uk/software.html
JointSLM	deprecated	
CNVnator	maintained	https://github.com/abyzovlab/CNVnator
ReadDepth	deprecated	https://github.com/chrisamiller/readDepth
CNOGpro	deprecated	https://github.com/cran/CNOGpro
Sequana	maintained	https://github.com/sequana/sequana
CNV-BAC	deprecated	https://github.com/LinjieWu/CNV-BAC
Assembly approach tools		
NovelSeq	deprecated	https://novelseq.sourceforge.net/Home
Cortex	deprecated	http://cortexassembler.sourceforge.net

TIGRA	deprecated	https://bioinformatics.mdanderson.org/public-software/archive/tigra/
Hybrid / integrating approaches tools		
CNVer	deprecated	http://compbio.cs.toronto.edu/CNVer/
HYDRA	deprecated	https://code.google.com/archive/p/hydra-sv/
He et al	method only	
inGAP-sv	deprecated	https://ingap.sourceforge.net/
forestSV	deprecated	https://sebatlab.org/data-software
GASVpro	deprecated	https://compbio.cs.brown.edu/projects/gasv/
PRISM	deprecated	http://compbio.cs.toronto.edu/prism/
DELLY	maintained	https://github.com/dellytools/delly
PeSV-Fisher	not available	http://gd.crg.eu/tools
LUMPY	maintained	https://github.com/arq5x/lumpy-sv
Pilon	maintained	https://github.com/broadinstitute/pilon/releases/
TARDIS	maintained	https://github.com/BilkentCompGen/tardis
SV / CNV detection pipelines		
SVmerge	deprecated	https://svmerge.sourceforge.net/
HugeSeq	deprecated	https://github.com/StanfordBioinformatics/HugeSeq
iSVP	only method	
intansv	maintained	https://bioconductor.org/packages/release/bioc/html/intansv.html
MetaSV	deprecated	https://github.com/bioinform/metasv
FusorSV	maintained	https://github.com/timothyjamesbecker/FusorSV
sv-callers	maintained	https://github.com/GooglingTheCancerGenome/sv-callers
Parliament2	deprecated	https://github.com/dnanexus/parliament2
Viola	maintained	https://github.com/dermasugita/Viola-SV
* deprecated means that the source code is untouched since the date of publication or updated in a distant time. Some packages are not available at all, some are maintained and used, and some are deprecated yet still used.		

Supplementary Table 2 - GC normalization with Spearman's correlation coefficient

Sample	**Condition**	**Rho**	**P-value**	**Significant (α=0.05)**
E. coli	Raw	-0.971	8.395E-37	Yes
E. coli	Normalized	0.222	0.093	**No**
K. pneumoniae	Raw	-0.301	0.012	Yes
K. pneumoniae	Normalized	0.110	0.371	**No**
L.casei	Raw	-0.928	3.510E-23	Yes
L.casei	Normalized	-0.160	0.254	**No**
S. aureus	Raw	-0.981	4.066E-38	Yes
S. aureus	Normalized	0.178	0.201	**No**

Supplementary Table 3 - mappability normalization with Spearman's correlation coefficient				
Sample	**Condition**	**Rho**	**P-value**	**Significant (α=0.05)**
E. coli	Raw	-0.158	0.517	No
E. coli	Normalized	-0.202	0.406	No
K. pneumoniae	Raw	-0.451	0.059	No
K. pneumoniae	Normalized	S.D. of coverage for mappability score is zero		
L.casei	Raw	0.355	0.147	No
L.casei	Normalized	S.D. of coverage for mappability score is zero		
S. aureus	Raw	0.949	1.698E-09	Yes
S. aureus	Normalized	-0.383	0.116	**No**

Supplementary Table 4 - oriC normalization with Spearman's correlation coefficient				
Sample	**Condition**	**Rho**	**P-value**	**Significant (α=0.05)**
E. coli	Raw	0.235	0.0007	Yes
E. coli	Normalized	0.090	0.203	**No**
K. pneumoniae	Raw	-0.252	3.326E-05	Yes
K. pneumoniae	Normalized	-0.219	0.0003	Yes
L.casei	Raw	-0.933	3.192E-65	Yes
L. casei	Normalized	-0.617	1.611E-16	Yes
S. aureus	Raw	-0.824	3.185E-37	Yes
S. aureus	Normalized	-0.345	2.023E-05	Yes

Supplementary Table 5 - Used tools in the ProcaryaSV pipeline		
Tool/Package	**Version**	**Usage**
fastqc	0.11.9	Raw and trimmed reads QC
trim_galore	0.6.7	Reads trimming
qualimap	2.2.2d	Alignment analytics and report
multiqc	1.13	Merging reports
bwa-mem2	2.2.1	Reads alignemnt
sambamba	0.7.1	SAM/BAM handling
samblaster	0.1.24	Marking PCR duplicates and discordant reads
samtools	1.13	SAM/BAM handling
genmap	1.3.0	Mappability file creation (for CNproScan)
Picard suite	2.21.1	Alignment analytics (for Pindel)
CNproScan	1.0	CNV detection (RD+PR methods)
CNVnator	0.4.1	CNV detection (RD method)
LUMPY	0.3.1	SV detection (RD+PR+SR methods)
DELLY2	0.9.1	SV detection (RD+PR+SR methods)

Pindel	0.2.5b9	SV detection (SR method)
survivor	1.0.7	SV merging
ProcaryaSV	1.0	SV merging